Web Crawlers com Scrapy

Gileno Alves Santa Cruz Filho

Gileno, quem?

Centro de Informática
UFPE

PyCursos

O que são Web Crawlers?

Crawler
X
Scraping

Scraping

- Extrair Dados Estruturados de algo não estruturado
- HTML
- XML
- IMAGENS (OCR)
- PDF

Crawler

- Visitar Páginas Web
- Seguir regras de visitação
- Não necessariamente precisa extrair informação estruturada (fazer scraping)

Porque criá-los?

Informação == dinheiro


Google

E-Commerce

- Qual o valor que meus concorrentes estão cobrando?
- Quais produtos estão sendo vendidos na loja X?
- Qual a variação de preço?
- Produto está disponível?

Imóveis

- Qual preço praticado na região X?
- O apartamento X está abaixo do preço praticado no mercado?
- Que imóveis estão disponíveis na cidade Y?

Cotação Dolar

Dólar Hoje

US$ 1,00

DÓLAR COMERCIAL

R$ 4,85

DÓLAR TURISMO

SELECIONE A CIDADE...

Compare as cotações do **Dólar** em sua cidade

Quero comprar | Quero vender

Remessa online

Código Fonte

```
348 <input type="hidden" id="nome-moeda" value="Dólar" />
349 <input type="hidden" id="artigo" value="do" />
350 <input type="hidden" id="id-moeda" value="8" />
351 <input type="hidden" id="toggle" value="0" />
352 <input type="hidden" value="4,85" id="taxa-comercial">
353 <input type="hidden" id="taxa-turismo" value="0" />
354 <input type="hidden" id="input-alterado" value="" />
```

urllib

```
import urllib.request
import re
url = 'https://www.melhorcambio.com/dolar-hoje'
headers = {
        'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/63.0.3239.132 Safari/5’
}
req = urllib.request.Request(url, headers=headers)
with urllib.request.urlopen(r) as response:
        html = response.read().decode(’utf-8’)
preco = re.findall(r'<input type="hidden" value="(.*)" id="taxa-comercial">', html)[0]
print(preco)
```

Usando Requests

```
import requests
import re
url = 'https://www.melhorcambio.com/dolar-hoje'
headers = {
'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.132 Safari/5’
}
req = requests.get(url, headers=headers)
html = req.text
preco = re.findall(r'<input type="hidden" value="(.*)" id="taxa-comercial">', html)[0]
print(preco)
```

Scrapy

Spiders

```html
<table data-apvhdrid="512">
<tbody>
    <tr data-id="1">
        <td data-itemid="001">Beer</td>
        <td data-quantity="10">10 bottles</td>
        <td data-unitcost="11.00">11.00</td>
        <td data-amount="110.00">110.00</td>
    </tr>
    <tr data-id="2">
        <td data-itemid="002">Vodka</td>
        <td data-quantity="20">20 bottles</td>
        <td data-unitcost="100.00">100.00</td>
        <td data-amount="2000.00">2000.00</td>
    </tr>
</tbody>
</table>
```

```
[
    {"apvhdrid":512, "id":1, "itemid":001, "quantity":10, "unitcost":"10.00", "amount":"110.00"},
    {"apvhdrid":512, "id":2, "itemid":001, "quantity":20, "unitcost":"100.00", "amount":"2000.00"}
]
```

Items (html -> dados estruturados)

Usando Scrapy

```
import scrapy
import re
class DolarSpider(scrapy.Spider):
    name = 'dolar_hoje'
    start_urls = ['https://www.melhorcambio.com/dolar-hoje']
    custom_settings = {
        'USER_AGENT': 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/63.0.3239.132 Safari/5'
    }
    def parse(self, response):
        html = response.text
        preco = re.findall(
            r'<input type="hidden" value="(.*)" id="taxa-comercial">', html
        )[0]
        self.log(preco)
```

Seletor xpath / css

XPath

- Linguagem de consulta para arquivos XML
- Uma espécie de SQL para arquivos XML (pode ser usado em HTML)

Usando xpath e css

```
import scrapy
import re


class DolarSpider(scrapy.Spider):

    name = 'dolar_hoje'
    start_urls = ['https://www.melhorcambio.com/dolar-hoje']

    def parse(self, response):
        preco = response.xpath('//input[@id="taxa-comercial"]/@value')
        self.log(preco.extract_first())
        preco = response.css('#taxa-comercial')[0]
        self.log(preco.attrib['value'])
```

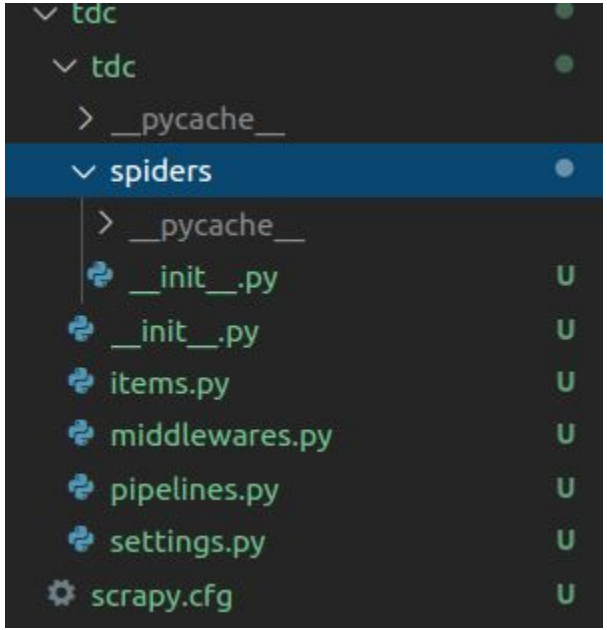
Items

Retornando Items

```
import scrapy
class OlxSpider(scrapy.Spider):

    name = "olx"
    start_urls = ["https://pe.olx.com.br/imoveis/aluguel"]
    custom_settings = {
        "USER_AGENT": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/99.0.4844.82 Safari/537.36"
    }

    def parse(self, response):
        items = response.xpath('//*[@id="ad-list"]/li')
        for item in items:
            yield {
                "url": item.xpath(".//a/@href").extract_first(),
                "titulo": item.xpath(".//h2/text()").extract_first(),
            }
```

Scrapy Project

scrapy startproject tdc



Item Pipeline

pipelines.py

```
class TdcPipeline(object):

    def process_item(self, item, spider):
        # Faz alguma limpeza
        # Salva no banco de dados
        return item
```

.... settings.py

```
ITEM_PIPELINES = {
        tdc.pipelines.TdcPipeline: 300,
}
```

yield Request

Criando novas requisições

```
import scrapy
class OlxSpider(scrapy.Spider):
    name = 'olx'
    start_urls = ['https://pe.olx.com.br/imoveis/aluguel']
    def parse(self, response):
        items = response.xpath('//*[@id="ad-list"]/li')
        for item in items:
            url = item.xpath(".//a/@href").extract_first()
            if url:
                yield scrapy.Request(url=url, callback=self.parse_detail)
    def parse_detail(self, response):
        preco = response.xpath("//h2[@font-weight='400']/text()").extract_first()
        yield {
            'url': response.url,
            'titulo': response.xpath("//title/text()").extract_first(),
            'preco': preco,
        }
```

Plugins e Settings

settings.py

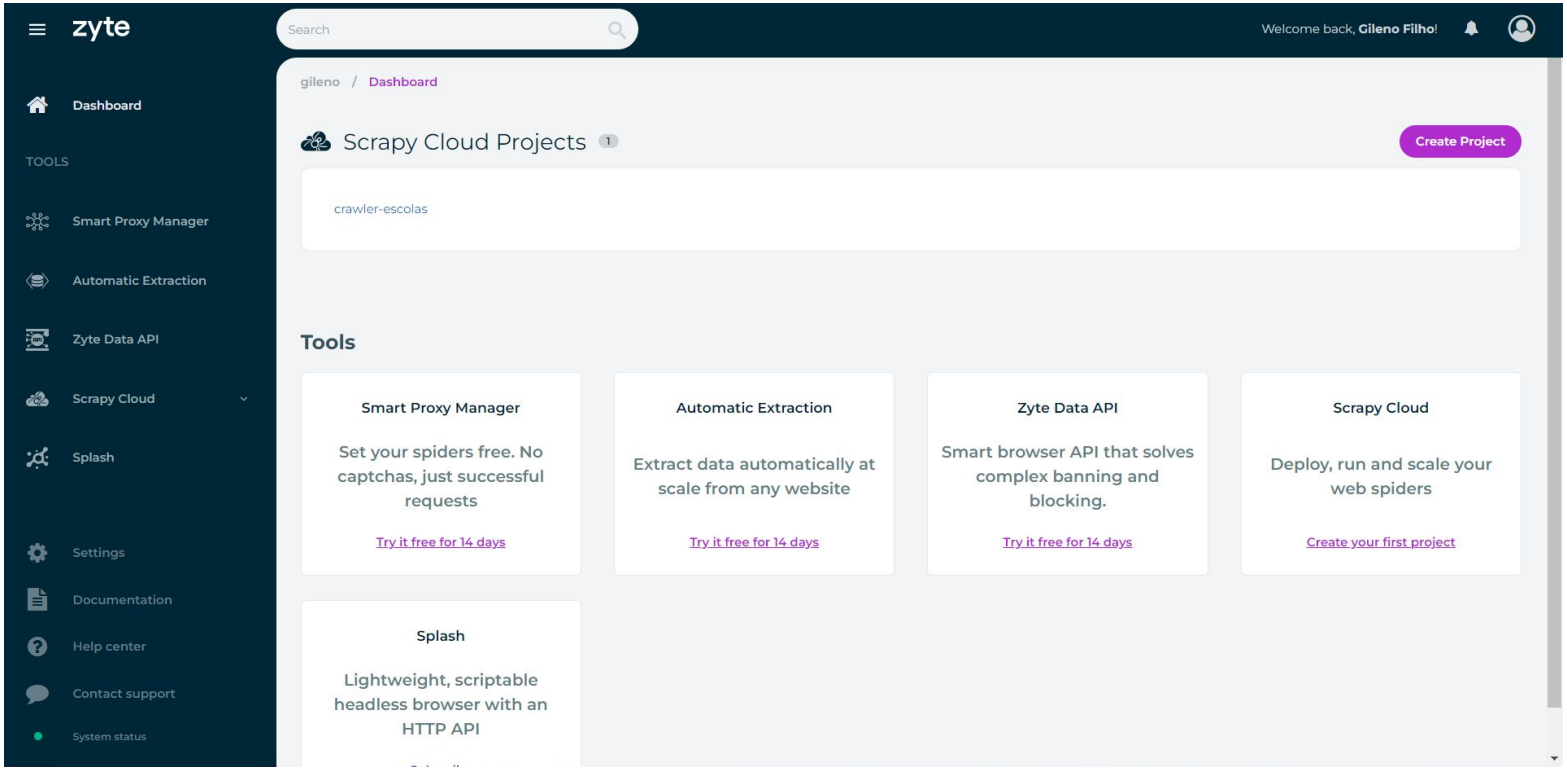
```
SPIDER_MODULES = [tdc.spiders']
NEWSPIDER_MODULE = 'tdc.spiders'

USER_AGENT = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.82 Safari/537.36'

# Obey robots.txt rules
ROBOTSTXT_OBEY = True

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

DOWNLOAD_DELAY = 0.5
```

plugins

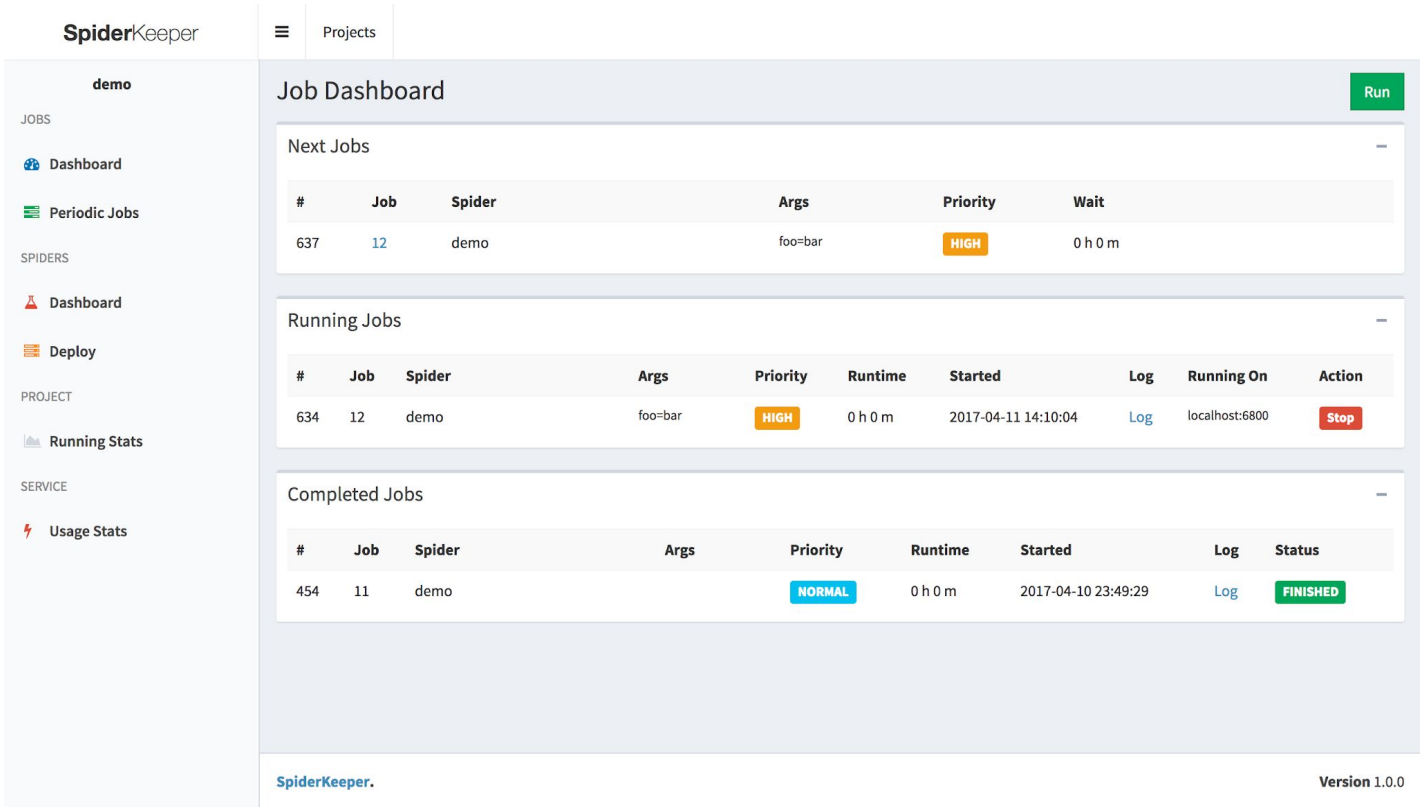
- https://github.com/scrapy-plugins
- Scrapy Splash

Deploy

Scrapinghub -> Zyte



Spider Keeper



Talk is cheap, Show me the code

Obrigado!
Dúvidas?

@gilenofilho
contato@gilenofilho.com.br
https://www.pycursos.com
https://github.com/gileno/tdcrecife2019