



CAIPYRA

Extraindo dados da internet
usando Scrapy

Renne Rocha

Scrapinghub

Laboratório Hacker de Campinas

Grupy-Campinas

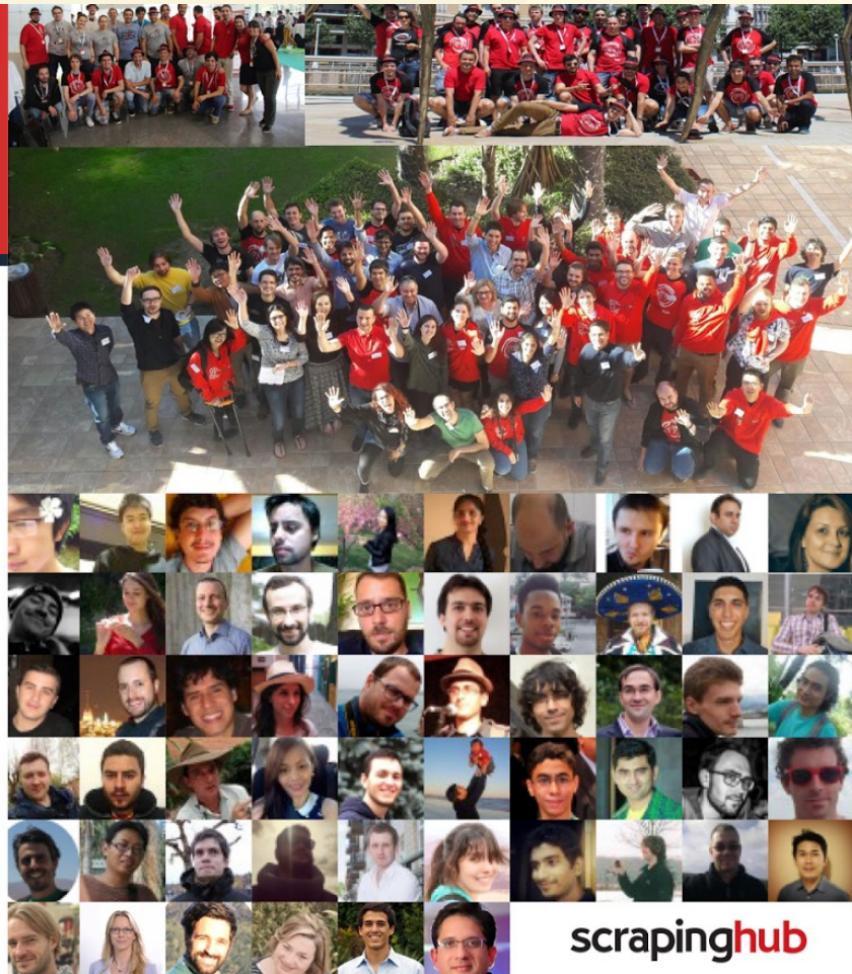
@rennerocha

renne@rennerocha.com

We are Scrapinghub - We embrace work, challenges, and appreciate diversity, we welcome innovation, we are an eclectic team who delivers and have fun.

Scraping**hub** is hiring!

scrapinghub.com/jobs



Getting information off the Internet is like taking a drink from a fire hydrant.

Mitchell Kapor



Web Scraping

Extrair dados **estruturados** de fontes de dados **não estruturadas** (tipicamente páginas web)

Casos de Uso

1. Pesquisas com dados governamentais
2. Monitorar o que estão falando do meu produto
3. Monitorar os produtos dos concorrentes
4. Ofertas de emprego, imóveis, bens de consumo
5. Análise de redes sociais

Ferramentas

urllib2 + re + xml

requests + bs4



Scrapy

<https://scrapy.org/>

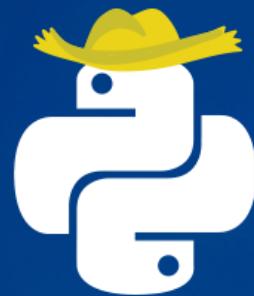
An open source and collaborative framework for extracting the data you need from websites.

In a fast, simple, yet extensible way.

Obtendo os títulos das palestras do Caipyra 2018



Sobre Inscrição Programação Tutoriais Sprints Locais Patrocinadores



CAIPYRA

O único evento de Python com Quentão e Paçoquinha!

8 a 11 de Junho de 2018 — São Carlos / SP



Dia 09/06 - Sábado

08:00 Credenciamento

09:00 Auditório A Abertura

Grupy Sanca

Boas vindas e apresentação do evento.

09:30 Auditório A Indicadores Inteligentes para Detecção de Epidemias de Dengue através do monitoramento de Redes Sociais em Tempo Real
Jadson José Monteiro Oliveira

A dengue é uma das maiores preocupações para a saúde brasileira devido ao aumento no número de casos ao passar dos anos e ao seu potencial devastador para a saúde humana. Como poderíamos utilizar a grande quantidade de dados gerados nas redes sociais como auxílio para a tomada de decisões e para o desenvolvimento de ações reativas e proativas, com o objetivo de tornar o saneamento público mais eficiente e eficaz? Nesta palestra, apresentarei como podemos

Dia 10/06 - Domingo

08:00 Credenciamento

09:00 Auditório A Lightning Talks

Qualquer pessoa presente pode fazer uma apresentação curta, de 5 minutos, sobre o tema que quiser.

09:30 Auditório A Processamento Paralelo para Pythonistas
Juliana Oliveira

Mais de dez anos depois do lançamento do primeiro processador multinúcleo no mercado, ainda precisamos de intensas horas de StackOverflow pra conseguir rodar mais de um processo ao mesmo tempo. Vamos descobrir — na prática — o que funciona, o que não funciona, as bibliotecas e como fazer processamento paralelo em Python.

10:30 Coffee Break



The screenshot shows a Firefox browser window displaying the agenda for the Caipyra 2018 Python event. The page lists various sessions with speakers and descriptions. The developer tools' Inspector tab is active, highlighting the HTML structure of one of the session cards.

Browser Title: Caipyra 2018 - O único evento de Python

Page URL: caipyra.python.org.br/#agenda

Developer Tools Inspector Panel:

```
<div class="inner flex flex-2"> flex
  <!--Dia 09/06-->
  <article>
    <header id="sabado">...</header>
    <div class="media">...</div>
    <div class="media">...</div>
    <div class="media">
      <div class="media-left">...</div>
      <div class="media-body">
        <span class="label label-danger">09:30</span>
        ...
        <span class="label label-danger2">Auditório A</span>
        ...
        <span class="label label-danger4">P</span>
        ...
      </div>
    </div>
    <h4 class="media-heading color-danger">
      Indicadores Inteligentes para Detecção de Epidemias de Dengue através do monitoramento de Redes Sociais em Tempo Real
    </h4>
    <br>
    <b>Jadson José Monteiro Oliveira</b>
  </article>
</div>
```

Session Details:

- Title:** Indicadores Inteligentes para Detecção de Epidemias de Dengue através do monitoramento de Redes Sociais em Tempo Real
- Speaker:** Jadson José Monteiro Oliveira
- Time:** 09:30
- Location:** Auditório A
- Category:** P

Description (Partial):

A dengue é uma das maiores preocupações para a saúde brasileira devido ao aumento no número de casos ao passar dos anos e ao seu potencial devastador para a saúde humana. Como poderíamos utilizar a grande quantidade de dados gerados nas redes sociais como auxílio para a tomada de decisões e para o desenvolvimento de ações reativas e proativas com o

```
import requests  
  
response = requests.get('http://caipyra.python.org.br/')
```

```
import requests
from bs4 import BeautifulSoup

response = requests.get('http://caipyra.python.org.br/')
soup = BeautifulSoup(response.text, 'html.parser')

palestras = soup.find('section', id='palestras')
titulos = [
    palestra.get_text()
    for palestra in palestras.find_all('h4')
]
```

E se estivessemos processando?

- Um e-commerce com milhares de produtos?
- Um site governamental com dados de diversos anos?
- Um site de notícias com dezenas de categorias?

Processamento Síncrono



Scrapy

- Framework especializado em web scraping
- Assíncrono
- Open-source com forte comunidade
- Altamente extensível

Imprensa Oficial

do Município de Jundiaí



05/06/2018
Edição Extra 4407



01/06/2018
Edição 4406



30/05/2018
Edição 4405



25/05/2018
Edição Extra 4404



25/05/2018
Edição 4403



23/05/2018
Edição 4402



18/05/2018
Edição 4401



17/05/2018
Edição Extra 4400

Busca por número da
edição

buscar no site

ok

Edições por data

Selecionar o mês

JUNHO 2018

D	S	T	Q	Q	S	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

« maio

```
# sp_jundiai.py
import scrapy

class SpJundiaiDiarioOficialSpider(scrapy.Spider):
    pass
```

```
# sp_jundiai.py
import scrapy

class SpJundiaiDiarioOficialSpider(scrapy.Spider):
    name = 'sp_jundiai'
```

```
# sp_jundiai.py
import scrapy

class SpJundiaiDiarioOficialSpider(scrapy.Spider):
    name = 'sp_jundiai'

    def start_requests(self):
        url = 'https://imprensaoficial.jundiai.sp.gov.br/'
        yield scrapy.Request(
            url=url,
            callback=self.parse,
        )
```

```
# sp_jundiai.py
import scrapy

class SpJundiaiDiarioOficialSpider(scrapy.Spider):
    name = 'sp_jundiai'

    def start_requests(self):
        url = 'https://imprensaoficial.jundiai.sp.gov.br/'
        yield scrapy.Request(
            url=url,
            callback=self.parse,
        )

    def parse(self, response):
        self.logger.info(
            'Received content from {}'.format(
                response.url))
```

```
$ scrapy runspider sp_jundiai.py
```

```
...
2018-06-05 22:58:28 [scrapy.utils.log] INFO: Scrapy 1.5.0 s
2018-06-05 22:58:28 [scrapy.core.engine] INFO: Spider opened
2018-06-05 22:58:30 [scrapy.core.engine] DEBUG: Crawled (200)
2018-06-05 22:58:30 [sp_jundiai] INFO: Received content from
2018-06-05 22:58:30 [scrapy.core.engine] INFO: Closing spider
2018-06-05 22:58:30 [scrapy.statscollectors] INFO: Dumping
2018-06-05 22:58:30 [scrapy.core.engine] INFO: Spider closed
...

```

Imprensa do Município

li#page-7046.edicao-atual | 145 x



05/06/2018
Edição Extra 4407



01/06/2018
Edição 4406



25/05/2018
Edição 4403



23/05/2018
Edição 4402



Developer Tools - Imprensa Oficial | Prefeitura de Jundiaí - https://www.jundiai.sp.gov.br/impressaooficial

Inspect Console Debug Style Editor Performance

Search HTML

```
<!DOCTYPE html>
<html lang="pt-br"> ev
<script async="" src="//www.google-analytics.com/analytics.js"
></script>
<script id="zm-extension" type="text/javascript"
charset="utf-8" src="moz-extension://746cb3dc-5db0-49fd-
a0c9-5ae0e9f9a0ca/scripts/webrtc-patch.js" async=""></script>
▶ <head> ...
  </head>
  <body class="home blog">
    <div id="main">
      <div id="header"> ...
      <div id="conteudo">
        <ul id="lista-edicoes">
          <li id="page-7051" class="edicao-atual"> ...
          <li id="page-7046" class="edicao-atual"> ...
          <li id="page-7042" class="edicao-atual"> ...
          <li id="page-7038" class="edicao-atual"> ...
          <li id="page-7032" class="edicao-atual"> ...
          <li id="page-7028" class="edicao-atual"> ...
          <li id="page-7021" class="edicao-atual"> ...
          <li id="page-7012" class="edicao-atual"> ...
          <li id="page-7006" class="edicao-atual"> ...
          <li id="page-7002" class="edicao-atual"> ...
          <li id="page-6995" class="edicao-atual"> ...
          <li id="page-6990" class="edicao-atual"> ...
        <hr class="clear">
      </ul>
    </div>
  </div>
</body>
```

li#page-7051.edicao-atual > a > span.imagem > img.c >

digital da Imprensa Oficial de Jundiaí

```
$ scrapy shell https://imprensaoficial.jundiai.sp.gov.br/
```

```
$ scrapy shell https://imprensaoficial.jundiai.sp.gov.br/  
  
In [1]: response.css('#lista-edicoes li.edicao-atual')  
Out[1]:  
[<Selector data='<li class="edicao-atual" id="page-1">'>,  
 <Selector data='<li class="edicao-atual" id="page-2">'>,  
 <Selector data='<li class="edicao-atual" id="page-3">'>,  
 ...]
```

```
$ scrapy shell https://imprensaoficial.jundiai.sp.gov.br/  
  
In [1]: response.css('#lista-edicoes li.edicao-atual')  
Out[1]:  
[<Selector data='<li class="edicao-atual" id="page-1">'>,  
 <Selector data='<li class="edicao-atual" id="page-2">'>,  
 <Selector data='<li class="edicao-atual" id="page-3">'>,  
 ...]  
  
In [2]: diario = response.css(  
...:     '#lista-edicoes li.edicao-atual')[0]  
  
In [3]: diario.extract()  
Out[3]: '<li class="edicao-atual" id="page-7051">  
 \n\n\t\n\t<a href="https://imprensaoficial.j  
 undiai.sp.gov.br/edicao-extra-4407/" (...) </li>'
```

```
# sp_jundiai.py
def parse(self, response):
    diarios = response.css(
        '#lista-edicoes li.edicao-atual')
```

```
# sp_jundiai.py
def parse(self, response):
    diarios = response.css(
        '#lista-edicoes li.edicao-atual')

    for diario in diarios:
        pass
```

```
# sp_jundiai.py
def parse(self, response):
    diarios = response.css(
        '#lista-edicoes li.edicao-atual')

    for diario in diarios:
        diario_url = diario.css(
            'a::attr(href)').extract_first()
        diario_data = diario.xpath(
            './span[2]/text()').extract_first()
```

```
# sp_jundiai.py
def parse(self, response):
    diarios = response.css(
        '#lista-edicoes li.edicao-atual')

    for diario in diarios:
        diario_url = diario.css(
            'a::attr(href)').extract_first()
        diario_data = diario.xpath(
            './span[2]/text()').extract_first()

        yield {
            'url': diario_url,
            'data': diario_data
        }
```



25/05/2018
Edição 4403



23/05/2018
Edição 4402



18/05/2018
Edição 4401



17/05/2018
Edição Extra 4400

3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

» maio



16/05/2018
Edição 4399



11/05/2018
Edição 4398



10/05/2018
Edição Extra 4397



09/05/2018
Edição 4396

Assinatura digital

Veja como [Validar a assinatura digital](#) da Imprensa Oficial de Jundiaí

Não encontrou o que procurava?

Todas as edições da Imprensa Oficial de Jundiaí estão disponíveis na [Biblioteca Pública Municipal Prof. Nelson Foot](#)

Rua Dr. Cavalcanti, 396 - Vila Arens
Jundiaí - SP
CEP 13201-003

The screenshot illustrates a web application interface for the "IMPRENSA OFICIAL" of Jundiaí, São Paulo. The main content area displays a grid of thumbnail previews for different editions of the newspaper. The thumbnails are arranged in two columns. Each thumbnail includes the title "IMPRENSA OFICIAL", the date, and the edition number.

- Top Left: 25/05/2018, Edição 4403
- Top Right: 23/05/2018, Edição 4402
- Bottom Left: 16/05/2018, Edição 4399
- Bottom Right: 11/05/2018, Edição 4398

The right side of the screenshot shows the browser's developer tools, specifically the DOM inspector. A blue selection box highlights the HTML code for the pagination navigation bar. The code includes links for navigating between pages, with the current page (1) highlighted in blue. The developer tools also show the class names and href values for these links.

```
<br class="clear">
</ul>
<br class="clear">
<div class="paginacao">
    <span class="label page">Página 1 de 125</span>
    <span class="current">1</span>
    <a class="inactive" href="https://imprensaoficial.jundiai.sp.gov.br/page/2/">2</a> ev
    <a class="inactive" href="https://imprensaoficial.jundiai.sp.gov.br/page/3/">3</a> ev
    <a class="inactive" href="https://imprensaoficial.jundiai.sp.gov.br/page/4/">4</a> ev
    <a class="inactive" href="https://imprensaoficial.jundiai.sp.gov.br/page/5/">5</a> ev
    <a class="label" href="https://imprensaoficial.jundiai.sp.gov.br/page/2/">Próxima ></a> ev
    <a class="label" href="https://imprensaoficial.jundiai.sp.gov.br/page/125/">Última »</a> ev
</div>
```

The bottom status bar of the browser shows the URL as "a.inactive 24.9333". The footer of the page includes the text "jundiaí - SP CEP 13201-003". The developer tools sidebar on the right lists various CSS properties such as element, pagina, disp, marg, clea, font, etc.

```
# sp_jundiai.py
def parse(self, response):
    diarios = response.css(
        '#lista-edicoes li.edicao-atual')

    for diario in diarios:
        (...)

    prox_paginas = response.css(
        'div.paginacao a.inactive')
    for pagina in prox_paginas:
        url = pagina.css('*::attr(href)').extract_first()
        yield Request(
            url=url,
            callback=self.parse
        )
```

```
2018-06-05 23:18:04 [scrapy.core.scrape] DEBUG: Scraped from <200 https://imprensaoficial.jundiai.sp.gov.br/>
{'url': 'https://imprensaoficial.jundiai.sp.gov.br/edicao-extra-4400/', 'data': '17/05/2018'}
2018-06-05 23:18:04 [scrapy.core.scrape] DEBUG: Scraped from <200 https://imprensaoficial.jundiai.sp.gov.br/>
{'url': 'https://imprensaoficial.jundiai.sp.gov.br/edicao-4399/', 'data': '16/05/2018'}
2018-06-05 23:18:04 [scrapy.core.scrape] DEBUG: Scraped from <200 https://imprensaoficial.jundiai.sp.gov.br/>
{'url': 'https://imprensaoficial.jundiai.sp.gov.br/edicao-4398/', 'data': '11/05/2018'}
2018-06-05 23:18:04 [scrapy.core.scrape] DEBUG: Scraped from <200 https://imprensaoficial.jundiai.sp.gov.br/>
{'url': 'https://imprensaoficial.jundiai.sp.gov.br/edicao-extra-4397/', 'data': '10/05/2018'}
2018-06-05 23:18:04 [scrapy.core.scrape] DEBUG: Scraped from <200 https://imprensaoficial.jundiai.sp.gov.br/>
{'url': 'https://imprensaoficial.jundiai.sp.gov.br/edicao-4396/', 'data': '09/05/2018'}
2018-06-05 23:18:04 [scrapy.core.engine] INFO: Closing spider (finished)
2018-06-05 23:18:04 [scrapy.statscollectors] INFO: Dumping Scrapy stats:
{'downloader/request_bytes': 232,
'downloader/request_count': 1,
'downloader/request_method_count/GET': 1,
'downloader/response_bytes': 31331,
'downloader/response_count': 1,
'downloader/response_status_count/200': 1,
'finish_reason': 'finished',
'finish_time': datetime.datetime(2018, 6, 6, 2, 18, 4, 21267),
'item_scraped_count': 12,
'log_count/DEBUG': 14,
'log_count/INFO': 7,
'memusage/max': 51388416,
'memusage/startup': 51388416,
'response_received_count': 1,
'scheduler/dequeued': 1,
'scheduler/dequeued/memory': 1,
'scheduler/enqueued': 1,
'scheduler/enqueued/memory': 1,
'start_time': datetime.datetime(2018, 6, 6, 2, 18, 3, 518025)}
2018-06-05 23:18:04 [scrapy.core.engine] INFO: Spider closed (finished)
```

Item Exporters

Uma vez que você obteve seus items, você vai querer persisti-los ou exportá-los para utilizar esses dados em outra aplicação.

O Scrapy fornece um conjunto de *Item Exporters* para diferentes formatos como CSV, JSON ou JL.

Item Exporters

Uma vez que você obteve seus items, você vai querer persisti-los ou exportá-los para utilizar esses dados em outra aplicação.

O Scrapy fornece um conjunto de *Item Exporters* para diferentes formatos como CSV, JSON ou JL.

```
$ scrapy runspider sp_jundiai.py -o sp_jundiai.csv  
$ scrapy runspider sp_jundiai.py -o sp_jundiai.json  
$ scrapy runspider sp_jundiai.py -o sp_jundiai.jl
```

Páginas com JavaScript

- Conteúdo dinâmico
- Single Page Applications

casos simples:

- emular requests AJAX

casos complexos:

- Splash (<https://github.com/scrapinghub/splash>)

<http://quotes.toscrape.com/scroll>

Quotes to Scrape - Mozilla Firefox

Quotes to Scrape x +

← → C H quotes.toscrape.com/scroll ... 🌐 ⚡ ⚡ 1 ➞ ⌂

PSPI Staff Deck JSON Editor Online ProductMatching buscacerveja Brewing grain subst...

Developer Tools - Quotes to Scrape - http://quotes.toscrape.com/scroll

Inspect Consol Debug {} Style Edit Performance Memory Network Storage Persist Logs Disable cache

All HTML CSS JS XHR Fonts Images Media WS Other Persist Logs Disable cache

Filter URLs

Sta... Meth... Fil... Doc... Cau... Ty...

200 GET quot... ↴ q... xhr json

Request URL: <http://quotes.toscrape.com/api/quotes?page=2>

Request method: GET

Remote address: 136.243.118.219:80

Status code: ● 200 OK [Edit and Resend](#) [Raw head](#)

Version: HTTP/1.1

Filter headers

Response headers (222 B)

- Connection: keep-alive
- Content-Encoding: gzip
- Content-Type: application/json
- Date: Thu, 07 Jun 2018 00:45:39 GMT
- Server: nginx/1.12.1
- Transfer-Encoding: chunked
- X-Upstream: spidyquotes-master_web

Request headers (375 B)

- Accept: */*
- Accept-Encoding: gzip, deflate
- Accept-Language: en-US,en;q=0.5
- Cache-Control: no-cache
- Connection: keep-alive

by André Gide

Tags: life love

"I have not failed

by Thomas A. Edison

Tags: edison failure

"A woman is like water."

by Eleanor Roosevelt

Tags: misattributed-e

"A day without s

by Steve Martin

Tags: humor obvious

"This life is what

Developer tools - Quotes to Scrape - http://quotes.toscrape.com/scroll

All HTML CSS JS XHR Fonts Images Media WS Other Persist Logs Disable cache

Filter URLs

Sta...	Meth...	Headers	Cookies	Params	Response	Timings	Stack Trace
200	GET	r	Filter properties				
200	GET	b	JSON				
200	GET	d	has_next: true page: 3 quotes: [...] 0: {...} author: {...} goodreads_link: /author/show/4026.Pablo_Neruda name: Pablo Neruda slug: Pablo-Neruda tags: [...] 0: love 1: poetry text: "I love you without knowing how, or when, or from where. I love you simply, without problems or pride: I love you in this way because I do not know any other way of loving but this, in which there is no I or you, so intimate that your hand upon my chest is my hand, so intimate that when I fall asleep your eyes close." 1: {...} author: {...} goodreads_link: /author/show/12080.Ralph_Waldo_Emerson name: Ralph Waldo Emerson slug: Ralph-Waldo-Emerson tags: [...] 0: happiness text: "For every minute you are angry you lose sixty seconds of happiness." 2: {...} author: {...} goodreads_link: /author/show/838305.Mother_Teresa name: Mother Teresa				

3 requests | 12

```
33 <script>
34     $(function(){
35         var page = 1, tag = null, hasNextPage = true;
36         function appendQuotes(quotes) {
37             var $quotes = $('.quotes');
38             var html = $.map(quotes, function(d){
39                 var tags = $.map(d['tags'], function(t) {
40                     return "<a class='tag'>" + t + "</a>";
41                 }).join(" ");
42                 return "<div class='quote'><span class='text'>" + d['text'] + "</span><span>by <small>" +
43             });
44
45             $quotes.append(html);
46         }
47
48         function updatePage(page) {
49             $('#loading').show('fast');
50             $.get('/api/quotes', {page: page}).done(function(data) {
51                 appendQuotes(data.quotes);
52                 hasNextPage = data.has_next;
53                 $('#loading').hide('fast');
54             });
55         }
56         updatePage(page);
57         $(window).on('scroll', function(){
58             var scrollTop = $(window).scrollTop();
59             var heightDiff = $(document).height() - $(window).height();
60             if (hasNextPage && Math.abs(scrollTop - heightDiff) <= 1){
61                 page += 1;
62                 console.log('scrolling to page: ' + page);
63                 updatePage(page);
64             }
65         });
66     });
67 </script>
68 //
```

```
import json
import scrapy
from w3lib.url import add_or_replace_parameter

class ToScrapeScroll(scrapy.Spider):

    name = 'toscrape'
    start_urls = [
        'http://quotes.toscrape.com/api/quotes?page=1'
    ]

    def parse(self, response):
        data = json.loads(response.body)
        for quote in data.get('quotes'):
            yield {
                'text': quote.get('text'),
                'author': quote.get('name')
            }
```

```
import json
import scrapy
from w3lib.url import add_or_replace_parameter

class ToScrapeScroll(scrapy.Spider):
    (...)

    def parse(self, response):
        data = json.loads(response.body)
        for quote in data.get('quotes'):
            (...)

            if data.get('has_next'):
                current_page = data.get('page')
                next_page_url = add_or_replace_parameter(
                    response.url, 'page', current_page + 1)
                yield Request(next_page_url)
```

Medidas anti-bots

- Bloqueio de IP
- Número de requests por segundo
- User-Agent
- Cookies de sessão
- Localização

Dúvidas?

renne@rennerocha.com

@rennerocha
(twitter, telegram, github, bitbucket, etc)