Introdução a Web Scraping

Carlos Magno
Lucas Félix

UFSJ
Universidade Federal de São João del-Rei

VII SECOMP
VIII UBIMUS / I ERICM

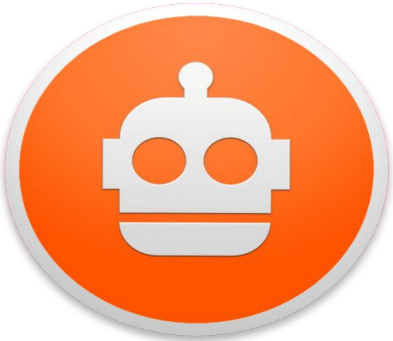
LABORATÓRIO DE MODELAGEM E INTELIGÊNCIA COMPUTACIONAL

O que vamos aprender nesse curso?

- O que é Web Scraping?
- A importância da coleta dos dados.
- Como coletar dados de uma página web usando python.
- As diferenças da coleta de dados em uma página estática e dinâmica.
- Como trabalhar com BeautifulSoup, Requests e Selenium.

O que é Web Scraping?

- É uma maneira de coletar dados de uma página na web.
- É o processo automatizado de extração de dados.
- Crawler é um programa automatizado que irá coletar dados de uma ou um conjunto de páginas pré determinado.
- Raspagem de dados.



Quando se utilizar um Crawler

- Sempre que você desejar coletar dados de uma página web.
- Quando você não encontrar os dados que você deseja de maneira estruturada disponível em arquivos.
- Quando você deseja descobrir novos ***padrões*** em dados presentes.
- Sempre que se precisar coletar dados de um determinado site.
- Diversas páginas fornecem um rico conteúdo que pode ser utilizado com objetivos científicos e profissionais para geração de novos projetos.

O que posso crawlear?

- Sites de notícias.
- Sites com resultado de jogos.
- Sites com candidatos da eleição.
- Site da Bolsa de Valores.
- Site de filmes.
- Site de animes.
- Wikipedia, Globoesporte, climatempo, AnimeList e etc...

Qual é a importância dos dados?

- O petróleo da era digital.
- A partir de dados é possível se extrair insight gerar informação e conhecimento.
- Com dados é possível se predizer tendências, encontrar padrões e solucionar problemas.

Os modos de obter dados

- Utilizando Api’s.
- Através de CSV ou outro formato disponibilizado pela empresa ou órgão.
- Coletando usando Web Scraping.

Que tecnologia vamos utilizar?

- Python.
- BeautifulSoup - permitirá navegar pela estrutura da página.
- Requests.
- Selenium.

Como é a estrutura de uma página web?

Um combinação principalmente de HTML, CSS, JavaScript, Imagens.

HTML - Linguagem de Marcação, onde fica os dados da página.

CSS - Responsável pelo estilo em uma página, contém id e classes que podem ser utilizados para facilitar a extração dos dados.

JavaScript - Utilizado para permitir interatividade com a página.

Estrutura HTML

```
<!DOCTYPE html>
<html lang="pt" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Título</title><style>...</style>
  </head>
  <body>
    <div id= “msg_1” class= “mensagens”>
        <p>Meu nome é Thrawn</p>
    <div>
    <div id= “msg_2” class= “mensagens” >
        <p>Olá meu nome é Vader</p>
        <a href="https://ufsj.edu.br">UFSJ</a>
  </body>
```

O que é Requests?

- Requests é uma biblioteca HTTP de código aberto escrita em python.
- *“Feita para humanos.”*
- *“HTTP para humanos”.*
- Facilita a realização de requisições facilitando a integração de sua aplicação com a web.
- Será responsável por fazer a coleta da página que queremos minerar.
- Permite realizar requisições **get, post.**

Alguns recurso do request

- get - Realiza uma requisição do tipo get.
- status_code.
- encoding.
- content.
- text.

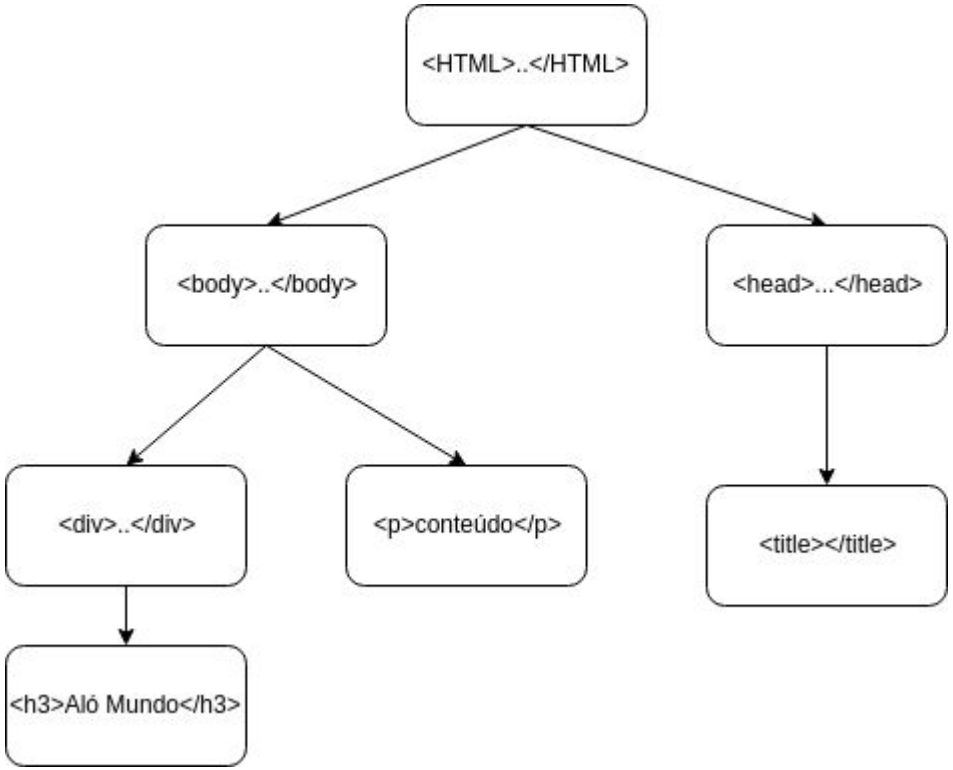
Realizando uma simples requisição

```
#importando a biblioteca

import requests

#chamando a biblioteca e a função desejada.

site= requests.get(“http://google.com”)

print(site.content)

# uma requisição com parâmetros

payload = {'key1': 'value1', 'key2': 'value2'}

r = requests.get("http://httpbin.org/get", params=payload)
```

O que é Beautiful Soup?

- Beautiful Soup é uma biblioteca de código aberto escrita em python para extração de dados de arquivos XML e HTML.
- Transforma HTML e XML em uma árvore de objetos navegáveis em python.
- Gera objetos de 4 tipos:
  - Tags: Corresponde as tags originais do XML, HTML;
  - NavigableString: Corresponde ao texto dentro das tags ;
  - Beautiful Soup: Similar ao objeto tag, porém contém a página inteira ;
  - Comments: Comentários;

Árvore de Navegação



Árvore de Navegação

```
from bs4 import BeautifulSoup
html_doc = """
<html><head><title>The Dormouse's story</title></head>
<body>
<p class="title"><b>The Dormouse's story</b></p>
<p class="story">Once upon a time there were three little sisters; and their names were
</body></html>
"""

soup = BeautifulSoup(html_doc, 'html.parser')
//Navegação através das tags.
print(soup.title)
# <title>The Dormouse's story</title>
print(soup.p)
# <p class= “title”><b>
```

Tag Names e Atributos

- tag.name
- soup.head
- soup.a
- Atributos em Tags:
    - soup.a[‘href’] - Retorna o link dentro da tag de link.

Pesquisando pelas tags usando métodos

- findAll(“p”) - Retorna uma lista com todos os elementos
- find(“p”) - Retorna o primeiro elemento.
- soup.findAll('"div",{"class":"profile-entry"}')
- soup.findAll(id="first")

- soup.select("div p")

Crawlear Páginas Dinâmicas e estáticas

- Podemos coletar os dados de páginas estáticas usando requests e BeautifulSoup.
- Por sua vez para coletar dados de páginas dinâmicas vamos precisar de combinar o Beautiful Soup com Selenium.

O que é o Selenium?

- Selenium é uma ferramenta para testar aplicações web.
- Uma biblioteca que pode ser utilizada para coletar dados de sites dinâmicos que usam javascript.
- Automatização do navegador.

Localizando Elementos

- find_element_by_id
- find_element_by_name
- find_element_by_xpath
- find_element_by_link_text
- find_element_by_partial_link_text
- find_element_by_tag_name
- find_element_by_class_name
- find_element_by_css_selector
- Para Retornar múltiplos elementos acrescente um “s” na chamada:
    a. find_elements_by_id

Uma requisição com selenium

```
#selenium

options = Options()

options.set_headless(headless=True)

#webdriver

navegador = webdriver.Firefox(firefox_options=options)

navegador.get("http://legendas.tv")

destaque = navegador.find_elements_by_class_name("destaque")
```

Extraindo HTML

...

destaques = navegador.find_elements_by_class_name('destaque')

html=destaque.get_attribute('innerHTML')

...

É hora de Codificar.

Agora vamos mostrar alguns exemplos simples para vocês praticarem.

Muito Obrigado!

Referências

- https://imasters.com.br/back-end/aprendendo-sobre-web-scraping-em-python-utilizando-beautifulsoup
- https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- https://selenium-python.readthedocs.io

Análise de dados com redes Complexas

- Aprenda a analisar dados usando uma abordagem de redes complexa.
- Teoria dos Seis Graus de Separação.
- Entenda as complexas relações que estão a sua volta...