Web Scraping e Fundamentos de HTML

Tempo estimado: 10 mins

Objetivos

Após concluir esta leitura, você será capaz de:

- Explicar conceitos-chave relacionados à estrutura HTML e à composição de tags HTML.
- Explorar o conceito de árvores de documentos HTML.
- Familiarizar-se com tabelas HTML.
- Obter uma visão geral dos fundamentos da extração de dados da web usando Python e BeautifulSoup.

Introdução à extração de dados da web

A extração de dados da web, também conhecida como coleta de dados da web ou extração de dados da web, é o processo de extrair informações de sites ou páginas da web. Envolve a recuperação automatizada de dados de fontes da web. As pessoas a utilizam para várias aplicações, como análise de dados, mineração, comparação de preços, agregação de conteúdo e muito mais.

Como funciona a extração de dados da web

Requisição HTTP

O processo geralmente começa com uma requisição HTTP. Um web scraper envia uma requisição HTTP para uma URL específica, similar a como um navegador web faria ao visitar um site. A requisição é geralmente uma requisição HTTP GET, que recupera o conteúdo da página da web.

Recuperação de página da web

O servidor web que hospeda o site responde à solicitação retornando o conteúdo HTML da página da web solicitada. Esse conteúdo inclui o texto visível e os elementos de mídia, além da estrutura HTML subjacente que define o layout da página.

Análise de HTML

Uma vez que o conteúdo HTML é recebido, você precisa analisar o conteúdo. A análise envolve dividir a estrutura HTML em componentes, como tags, atributos e conteúdo de texto. Você pode usar o BeautifulSoup em Python. Ele cria uma representação estruturada do conteúdo HTML que pode ser facilmente navegada e manipulada.

Extração de dados

Com o conteúdo HTML analisado, os web scrapers podem agora identificar e extrair os dados específicos de que precisam. Esses dados podem incluir texto, links, imagens, tabelas, preços de produtos, artigos de notícias e muito mais. Os scrapers localizam os dados procurando por tags HTML, atributos e padrões relevantes na estrutura HTML.

Transformação de dados

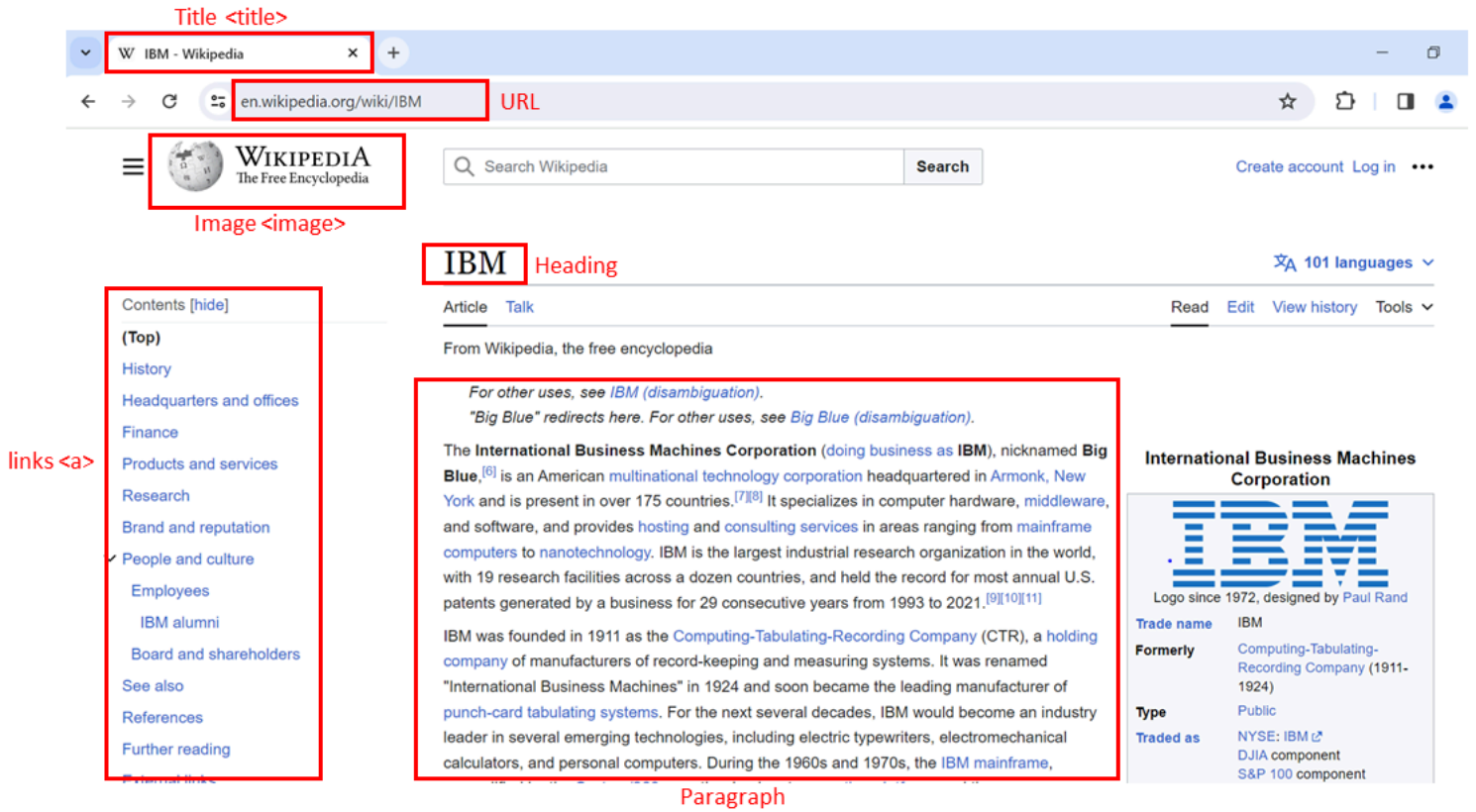
Os dados extraídos podem precisar de processamento e transformação adicionais. Por exemplo, você pode remover tags HTML do texto, converter formatos de dados ou limpar dados desorganizados. Esta etapa garante que os dados estejam prontos para análise ou outros casos de uso.

Armazenamento

Após a extração e transformação, você pode armazenar os dados coletados em vários formatos, como bancos de dados, planilhas, JSON ou arquivos CSV. A escolha do formato de armazenamento depende dos requisitos específicos do projeto.

Automação

Em muitos casos, scripts ou programas automatizam a extração de dados da web. Essas ferramentas de automação permitem a extração recorrente de dados de várias páginas da web ou sites. A extração automatizada é especialmente útil para coletar dados de sites dinâmicos que atualizam seu conteúdo regularmente.



Estrutura HTML

A linguagem de marcação de hipertexto (HTML) serve como a base das páginas da web. Compreender sua estrutura é crucial para a extração de dados.

- `<html>` é o elemento raiz de uma página HTML.
- `<head>` contém meta-informações sobre a página HTML.
- `<body>` exibe o conteúdo na página da web, frequentemente os dados de interesse.
- `<h3>` tags são cabeçalhos do tipo 3, tornando o texto maior e em negrito, tipicamente usados para nomes de jogadores.
- `<p>` tags representam parágrafos e contêm informações sobre os salários dos jogadores.

Composição de uma tag HTML

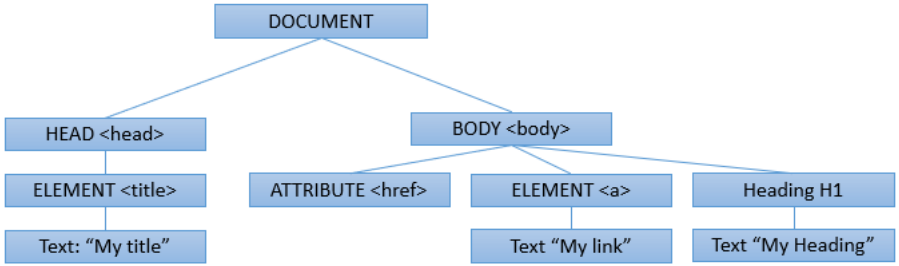
As tags HTML definem a estrutura do conteúdo da web e podem conter atributos.

- Uma tag HTML consiste em uma tag de abertura (início) e uma tag de fechamento (fim).
- As tags têm nomes (`<a>` para uma tag de âncora).
- As tags podem conter atributos com um nome de atributo e valor, fornecendo informações adicionais à tag.

Árvore de documentos HTML

Você pode visualizar documentos HTML como árvores, com tags como nós.

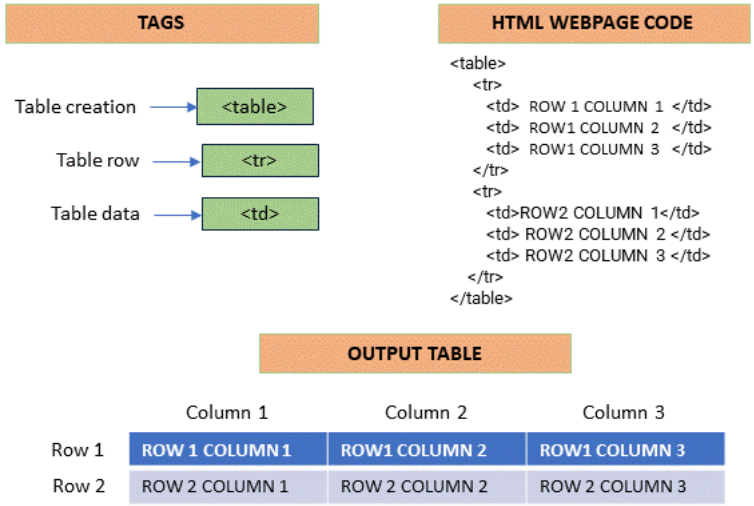
- As tags podem conter strings e outras tags, tornando-as as filhas da tag.
- As tags dentro da mesma tag pai são consideradas irmãs.
- Por exemplo, a tag `<html>` contém tanto as tags `<head>` quanto `<body>`, tornando-as descendentes de `<html>` mas filhas de `<html>`. `<head>` e `<body>` são irmãs.



Tabelas HTML

Tabelas HTML são essenciais para apresentar dados estruturados.

- Defina uma tabela HTML usando a tag `<table>`.
- Cada linha da tabela é definida com uma tag `<tr>`.
- A primeira linha geralmente utiliza a tag de cabeçalho da tabela, tipicamente `<th>`.
- A célula da tabela é representada pelas tags `<td>`, definindo células individuais em uma linha.



Web scraping

Web scraping envolve extrair informações de páginas da web usando Python. Pode economizar tempo e automatizar a coleta de dados.

Ferramentas necessárias

A extração de dados da web requer código em Python e dois módulos essenciais: Requests e Beautiful Soup. Certifique-se de que ambos os módulos estejam instalados em seu ambiente Python.

```
# Import Beautiful Soup to parse the web page content
from bs4 import BeautifulSoup
```

Buscando e analisando HTML

Para começar a raspagem de dados da web, você precisa buscar o conteúdo HTML de uma página da web e analisá-lo usando o Beautiful Soup. Aqui está um exemplo passo a passo:

```
import requests
from bs4 import BeautifulSoup
# Specify the URL of the webpage you want to scrape
url = 'https://en.wikipedia.org/wiki/IBM'
# Send an HTTP GET request to the webpage
response = requests.get(url)
# Store the HTML content in a variable
html_content = response.text
# Create a BeautifulSoup object to parse the HTML
soup = BeautifulSoup(html_content, 'html.parser')
# Display a snippet of the HTML content
print(html_content[:500])
```

Navegando na estrutura HTML

BeautifulSoup representa o conteúdo HTML como uma estrutura em forma de árvore, permitindo uma navegação fácil. Você pode usar métodos como find_all para filtrar e extrair elementos HTML específicos. Por exemplo, para encontrar todas as tags de âncora () e imprimir seu texto:

```
# Find all <a> tags (anchor tags) in the HTML
links = soup.find_all('a')
# Iterate through the list of links and print their text
for the link in links:
    print(link.text)
```

Extração de dados personalizada

A extração de dados da web permite que você navegue pela estrutura HTML e extraia informações específicas com base em suas necessidades. Este processo pode envolver a busca por tags, atributos ou conteúdo de texto específicos dentro do documento HTML.

Usando BeautifulSoup para análise de HTML

Beautiful Soup é uma ferramenta poderosa para navegar e extrair partes específicas de uma página da web. Ela permite que você encontre elementos com base em suas tags, atributos ou texto, facilitando a extração das informações que você está interessado.

Usando pandas read_html para extração de tabelas

Pandas, uma biblioteca Python, fornece uma função chamada read_html, que pode extrair automaticamente dados das tabelas de websites e apresentá-los em um formato adequado para análise. É semelhante a pegar uma tabela de uma página da web e importá-la para uma planilha para análise adicional.

Conclusão

Nesta leitura, você aprendeu sobre web scraping com BeautifulSoup e Pandas, com ênfase na extração de elementos e tabelas. BeautifulSoup facilita a análise de HTML, enquanto o read_html do Pandas simplifica a extração de tabelas. A leitura também destacou a prática responsável de web scraping, garantindo a conformidade com os termos dos sites. Armado com esse conhecimento, você pode se envolver com confiança na extração precisa de dados.

Autor

Akansha Yadav

Changelog

Data	Versão	Alterado por	Descrição da Alteração
2023-04-11	0.1	Akansha Yadav	Versão inicial criada
2024-19-02	0.2	Gagandeep	Revisão de ID
2024-21-02	0.3	Mary Stenberg	Revisão de QA
–>			

© IBM Corporation. Todos os direitos reservados.