Web Scraping: Como Coletar Dados de Sites (com R e Python)

Resolvi compilar este guia, passo a passo, sobre como usar códigos de programação (R e Python) para coletar e organizar dados de tabelas em sites. No caso em que utilizamos o fluxo, o objetivo final foi criar uma lista completa das Terras Indígenas (TIs) do Brasil com as respectivas Unidades Federativas (UF) e o nome completo do Estado.

O processo foi dividido em três etapas, usando a técnica de **Web Scraping** (raspagem de dados da web).

No repositório Github estão os arquivos .R e .ipynb (notebook Jupyter), e existe uma explicação no readme mais técnica e detalhada do processo de funcionamento de cada script.

Também deixo como bônus dois outros notebooks Jupyter: "Limpeza análise e salvamento.ipynb" (esse notebook lê arquivos csv, limpa dados duplicados e linhas com dados faltantes e faz uma análise exploratória dos dados, com a possibilidade de salvar essas análises em arquivos csv independentes do arquivo principal). E também o arquivo "Web Scraping de Tabelas Salvando em csv.ipynb" (esse arquivo busca por tabelas em paginas web, lista todas tabelas encontradas e pergunta quais queremos salvar. As tabelas escolhidas são salvas em arquivos csv independentes).

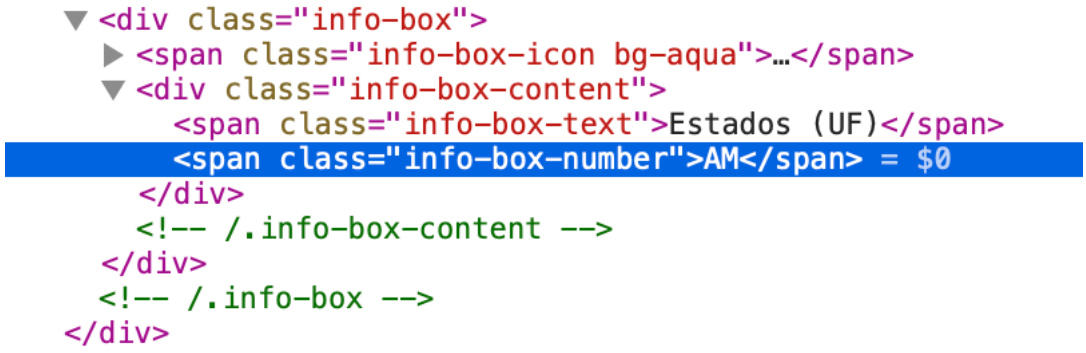
Os arquivos .R e .ipynb rodam tanto no vscode e IDEs similares como windsurf e cursor, quanto no google colab. Os arquivos com a extensão .ipynb são o formato padrão para o colab, e rodam sem nenhum problema. Para usar R, é preciso alterar o ambiente de execução do colab, na opção de "Python 3" para "R".

🔑 Conceito Chave: O Inspetor Web (Inspecionar Elemento)

A chave para qualquer Web Scraping encontrar os dados que queremos é entender como a informação alvo está "vestida" no site. O inspetor web (ou a ferramenta "Inspecionar Elemento" do seu navegador) permite que você veja o código HTML por trás de tudo que aparece na tela.

Como Usar o Inspetor Web para Encontrar o "Endereço" dos Dados:

1. **Abrir o Site-Alvo:** Naveguar até a página com a tabela que você quer copiar.
2. **Abrir o Inspetor:** Clicar com o botão direito do mouse sobre a tabela e selecionar **"Inspecionar"** ou **"Inspecionar Elemento"** (ou use a tecla F12). Uma janela lateral ou inferior será aberta mostrando o código HTML da página.
3. **Localizar a Tabela:** Na janela do inspetor, você pode usar o ícone de seta (geralmente no canto superior esquerdo da janela do inspetor) para clicar na tabela. O navegador irá destacar a parte exata do código HTML que corresponde a essa tabela.
4. **Descubrir o Seletor (Endereço):** Procurar pela *tag* <table> ou (alguns sites usam tags fora de padrão). Ela geralmente tem atributos como id ou class. Por exemplo, no nosso caso, a tabela tinha um conjunto de classes: class="table table-striped tablesorter".
5. **Criar o Seletor CSS:** Esse conjunto de classes ou o ID da tabela se torna o "endereço" (o Seletor CSS) que diz ao seu código de raspagem onde exatamente buscar os dados. No script R do Passo 1, o endereço usado é: “table.table.table-striped.tablesorter"
6. **Dica de ouro:** Na duvida de como especificar o seletor CSS no script, tire um print de tela do inspetor web com o elemento alvo especificado, como na imagem abaixo, e peça para uma IA criar a referencia.



7. O bloco de código deve algo como:

```
response = requests.get(url_detalhe, headers=headers)
response.raise_for_status()
soup = BeautifulSoup(response.text, 'html.parser')
info_boxes = soup.find_all('span', class_='info-box-number')
```

Passo a Passo do Projeto

O fluxo de trabalho começa com a extração da tabela principal em R (Extrair infos via seletor CSS.R), continua com a busca de informações detalhadas em Python (Web Scraping Terras Indigenas.ipynb), e finaliza com a limpeza e enriquecimento dos dados também em Python(Adicionar coluna com map.ipynb).

IMPORTANTE: A ordem descrita acima deve ser respeitada para que o correto fluxo de extração e transformação seja executado.

💻 Passo 1: Copiar a Tabela Principal (Usando R)

Neste passo, usamos a linguagem R e o pacote rvest para baixar a tabela principal do site e salvá-la como um arquivo base (terras_indigenas.csv).
Rodar o script "Extrair infos via seletor CSS.R".

🔍 Passo 2: Coletar as Siglas de UF (Usando Python)

A tabela principal nem sempre tem todos os dados. Neste caso, a sigla da UF (Estado) está na página de detalhes de cada TI. Usamos Python, as bibliotecas requests e BeautifulSoup para automatizar a visita a essas páginas, para coletar e preencher a informação que faltava.

Rodar o script "Web Scraping Terras Indigenas.ipynb". Um novo arquivo será salvo no mesmo diretório, já com as novas informações coletadas (terras_indigenas_com_uf.csv).

📝 Passo 3: Adicionar os Nomes Completos dos Estados (Usando Python/Pandas)

Agora que temos a sigla da UF, podemos facilmente adicionar o nome completo do estado.

Isso é feito usando um **dicionário de mapeamento** no Python.

Rodar o script Adicionar coluna com map.ipynb". Será criado e salvo o arquivo com as informações finais que queremos (terras_indigenas_com_uf_estado.csv).

Scripts para Teste e Modificação

No repositório Github estão os scripts do projeto e os scripts bónus (análise exploratória de dados e web scraping de tabelas em sites), e também uma documentação mais técnica e explicativa sobre o funcionamento dos códigos. Sinta-se à vontade para baixar, alterar e utilizar os códigos conforme sua necessidade.