Relatório 8 - Web Scraping com Python p/ Ciência de Dados (II)

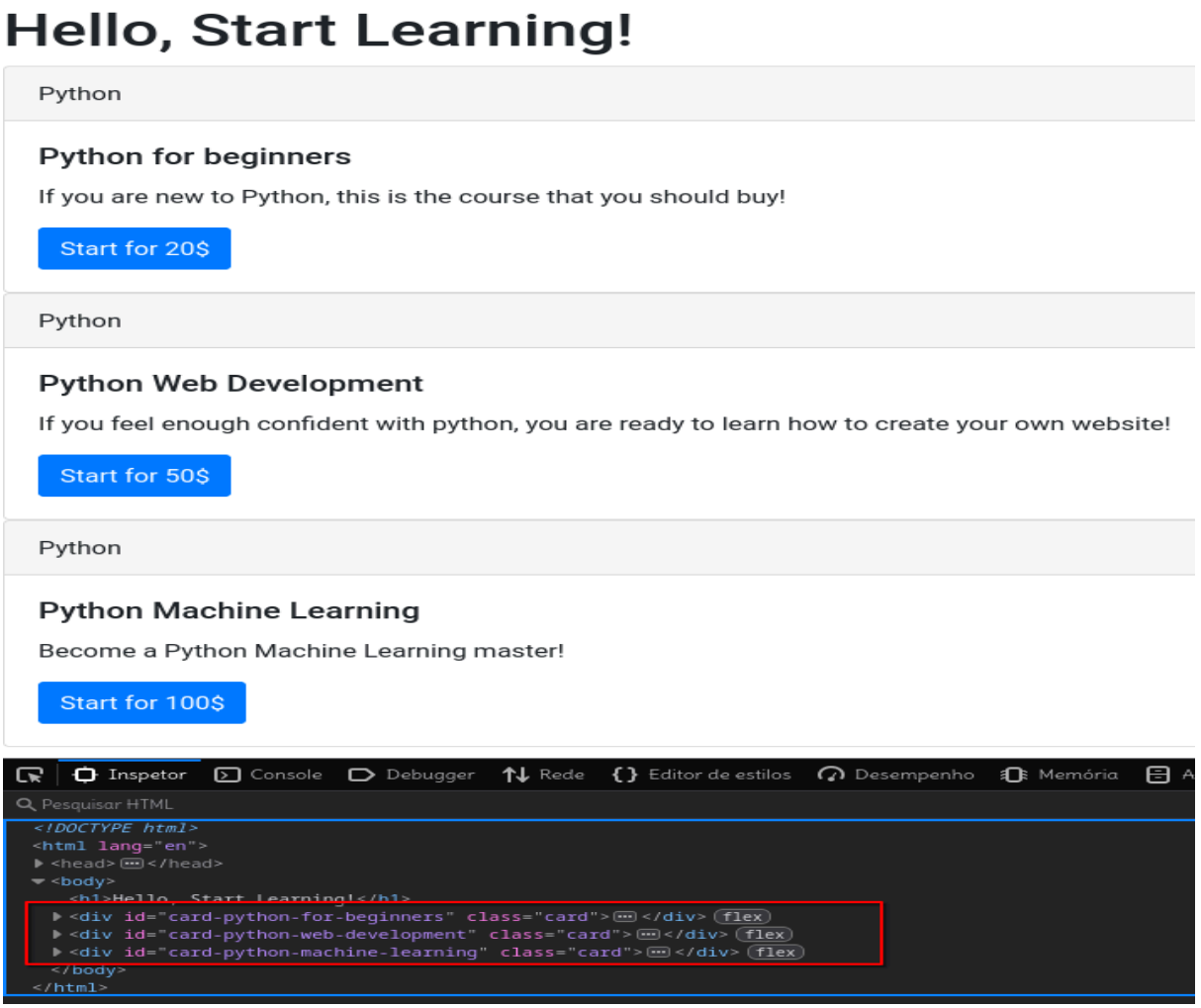
Felipe Fonseca

Descrição da atividade

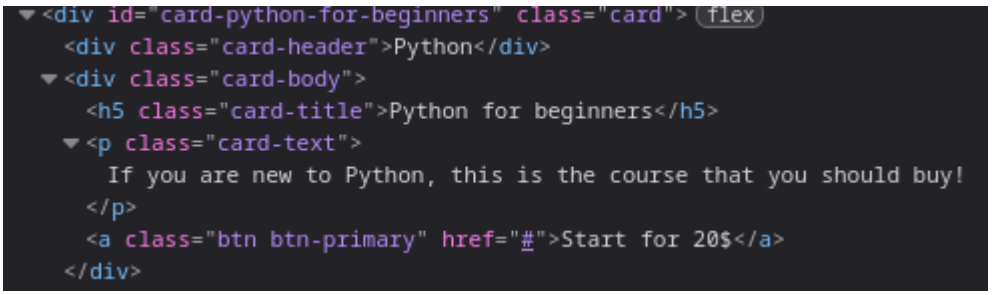
O objetivo deste card era aprender a executar Web Scraping, ou seja, conseguir retirar informações dos sites de forma estruturada para conseguir utilizar esses dados, tudo isso utilizando uma biblioteca chamada beautifulsoup, que serve justamente para extrair dados de arquivos html e xml.

Na atividade é introduzida a biblioteca Beautifulsoup, que serve para fazer a leitura do arquivo html e extrair dados dele de forma muito fácil e otimizada. Além disso, também é introduzido a biblioteca requests, que serve para pegar o arquivo html da página através da url.

No primeiro código, é utilizado apenas a biblioteca BeautifulSoup, pois o teste é feito com um arquivo html baixado no próprio computador. Através disso o que é feito é a leitura dos cards de atividades, que são apresentados no arquivo através de divs. Esses elementos são descobertos através da análise do site em questão, através da ferramenta de inspecionar do navegador.



Nesse exemplo, ao passar o mouse sobre as div, é possível perceber que cada “card” de curso é uma das divs, e dentro dessas divs que estão os itens de cada curso.



Através disso também é possível descobrir como achar os outros elementos do site, como o nome do curso, que é o ‘card-title’ que é do tipo h5, ou então o texto do card, que é do tipo ‘p’ (paragraph) e tem nome de ‘card-text’.

Essa é justamente a parte mais importante da atividade, aprender a analisar os elementos de um site é a base, sem isso, é impossível fazer qualquer um dos próximos passos como filtragem dos elementos.

Então ainda neste código, após achar todas as divs com a função find_all(), foi feito uma filtragem, que para cada div, pega-se o h5 (título do curso) e o link ‘a (botão de preço) através do atributo ‘text’ que tem ao fazer a leitura com o beautifulsoup, e printa esses valores na tela.

No segundo código foi feito algo bem semelhante, mas agora utilizando a biblioteca requests pra pegar o html da página, utilizando apenas a url como argumento. Através da função get, é passado a url como argumento dentro da string, e então para agilizar já puxamos apenas o texto através do atributo .text.

É feito novamente a leitura do arquivo então com o beautifulsoup, porém dessa vez achando todas as listas com um nome de classe específico, através novamente da função find_all(). Essas listas são os elementos da página timejobs.com, e cada lista é um “card” de emprego. Então novamente é feita a análise do site a fim de achar a melhor forma de filtrar os elementos que é desejado. Nesse caso, o span é onde tinha a data de publicação do emprego, um subtítulo h3 é onde tinha o nome da empresa, depois tinha uma div com várias skills de emprego, e então mais o link de como achar o card do emprego.

Algumas formas extras de extrações que foram utilizadas aqui é por exemplo o find(), que diferente do find_all() que acha todos os elementos com aquela característica, o find acha apenas o primeiro. Além disso é feito uma tática com a função replace para tentar limpar os espaços dos nomes para melhor formatação, tática que não funcionou tanto assim e eu corrigi em uma parte da prática posteriormente. Além disso é mostrado como pegar o link do href, afinal o link possui um endereço e o texto nele, utilizando apenas o atributo text, seria pego a mensagem do link, e não o link em si, e para isso ele é pego como se fosse um dicionário, através de colchetes com o nome do atributo, nesse caso ‘href’ dentro.

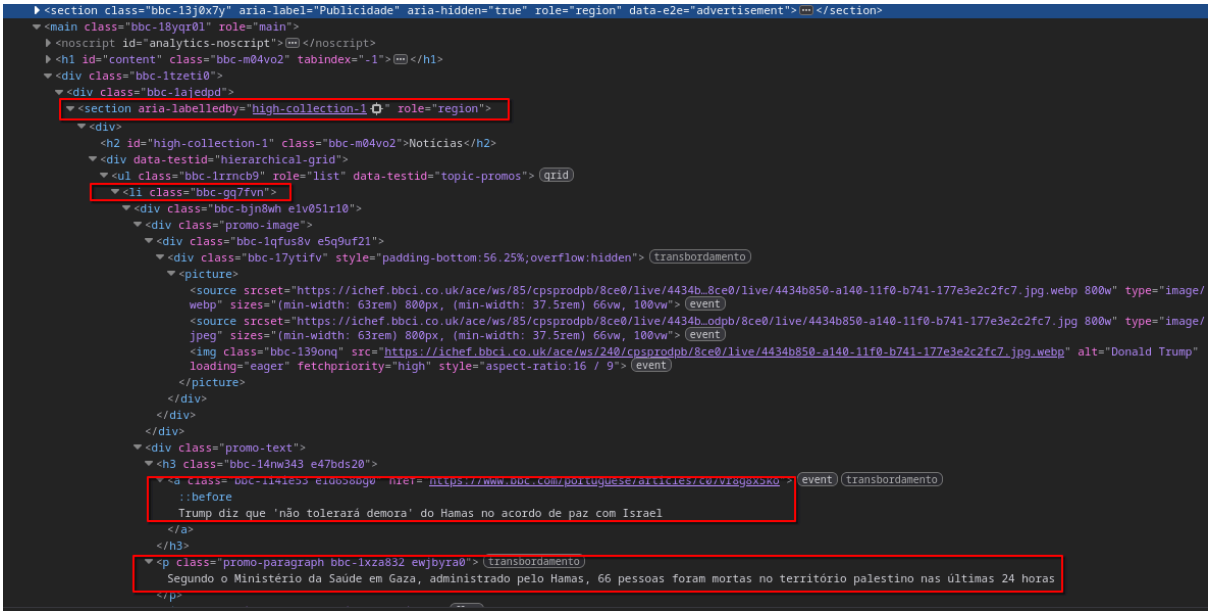
Para finalizar também foi feito algumas funções como salvar em um arquivo, ou então utilizar a função time para botar um temporizador no código, porém isso é algo mais simples de se fazer e não tem relação completa com o foco principal do card, então não vou aprofundar nessa parte.

Prática

Na primeira parte da prática eu foquei em tentar alterar algumas coisas do código da aula que eu não estava gostando. Então primeiramente eu separei tudo em funções diferentes invés de uma só, deixando uma maior organização do código, e coloquei também uma pequena interface de terminal para que o usuário escolha o filtro, ou seja, se quer colocar uma skill familiar ou uma que não seja familiar.

Após isso, fiz pequenas alterações na formatação dos textos, por exemplo utilizando a função lower() para que a filtragem não fosse case-sensitive. Também utilizei métodos diferentes do replace() para limpar os espaços. No exemplo do nome da empresa e das skills eu utilizei uma string com o espaço em branco, e utilizei a função join nela com o texto, na qual também foi utilizado a função split. No final o resultado foi uma boa formatação, sem espaços desnecessários, mas ainda mantendo o espaço entre os nomes da empresa/skills, coisa que não acontecia ao utilizar o replace.

No segundo código eu fiz uma raspagem de dados do site da bbc, buscando as informações das notícias. Através da análise do inspecionar foi possível observar que existiam várias seções, cada uma com várias notícias, porém o site no geral era bem estático e possui uma estrutura simples para pegar os dados.



Nesse caso, os atributos importantes eram a section, na qual eu tive que fazer uma filtragem pois eu só queria as sections que tinha o atributo aria-labelledby contendo algumas palavras específicas, e depois as notícias estavam dentro de listas “li”, e para cada notícia eu peguei então os dados de parágrafo, o título (nesse caso era o texto do link), o link em si, e algumas informações de data de postagem e o momento, tipo “2 horas atrás”. Para cada elemento, eu primeiro verifiquei se ele existia, pois em algumas notícias não existiam parágrafos por exemplo na página inicial, então para não dar erro, foi feito tratamentos para esses casos.

Conclusões

Esse card aborda de forma simples e rápida como analisar um site através do html dele, e então como fazer as raspagens desses dados através da biblioteca Beautifulsoup.

De forma geral, é um card que eu tive algumas dificuldades na hora de analisar os sites em si, isso porque alguns dos sites que eu estava querendo utilizar para prática incluíam javascript, o que impossibilitou que eu conseguisse pegar todas as informações deles.

Apesar disso, todas as habilidades desenvolvidas no card são importantes e devem servir de base para muita coisa. A raspagem de dados se mostrou algo muito interessante ao meu ver e pode ser utilizada em muitos contextos diferentes, desde automação até análise de dados.

Sobre as bibliotecas, a beautifulsoup eu não conhecia e achei interessante a forma como ela trabalha, tendo várias funções e métodos que ajudam a fazer a filtragem dos dados de forma muito fácil.

Referências

https://spotintelligence-com.translate.goog/2023/09/18/top-20-essential-text-cleaning-techniques-practical-how-to-guide-in-python/?_x_tr_sl=en&_x_tr_tl=pt&_x_tr_hl=pt&_x_tr_pto=tc

https://www.youtube.com/watch?v=XVv6mJpFOb0

https://realpython-com.translate.goog/beautiful-soup-web-scraper-python/?_x_tr_sl=en&_x_tr_tl=pt&_x_tr_hl=pt&_x_tr_pto=tc

https://canaltech.com.br/seguranca/o-que-e-web-scraping/