**Scraping com Python – Coleta De Dados Na Web – Anotações**

1. **Aula 1 – Nosso Primeiro Scraping:**
   1. Vamos fazer o scraping, ou seja, pegar as informações automaticamente de um site fictício da alura que foi desenvolvido para isso:
      1. <https://alura-site-scraping.herokuapp.com/index.php>.
      2. Para fazer o scraping utilizaremos 2 bibliotecas: urllib e bs4. Mais precisamente, para esse primeiro contato, utilizaremos as funções urlopen e BeautifulSoup, de cada uma respectivamente:



* + 1. Para abrir a url precisamos usar a função que importamos passando a url como parâmetro e em seguida, para ler o html da página passada, usar o.read() e tendo como resposta:





* + 1. Para organizar melhor e facilitar a visualização, utilizamos a BeautifulSoup(html, ‘html.parser’), passando a leitura do html e um parâmetro para organizando:





* + 1. Como queremos pegar algo em particular, clicamos com o botão direito em cima da informação que queremos obter e clicamos em “inspecionar elemento” abrindo uma aba que nos mostra exatamente qual a tag html em que essa informação está e suas classes e id’s. Em seguida, com essa informação, utilizamos o método .find(‘tag’, id=’se\_tiver\_e\_quiser’):



* + 1. Como observamos ele nos trouxe a tag inteira. Para pegar somente o texto utilizamos a função .get\_text(). Fizemos isso para o título e parágrafo da página que estamos scraping:



* 1. O que aprendemos:
     1. A configurar um ambiente de desenvolvimento com a distribuição Anaconda;
     2. A verificar as versões de bibliotecas Python;
     3. A executar um web scraping simples, do começo ao fim.

1. **Aula 2 – Obtendo e Limpando o HTML:**
   1. Entendendo a Web:
      1. Obtemos nossos dados pelo web scraping pois fazemos uma requisição a partir do computador/cliente para o servidor e o servidor nos devolve/responde o html da página.
   2. Obtendo o Conteúdo HTML de um Site:
      1. Utilizando o que vimos até agora, não será possível fazer a requisição e obter a informação de qualquer site, pois a maioria possui um tipo de proteção para isso. Com isso temos que utilizar outros métodos da biblioteca urllib, tais como:



* + 1. Nesse caso utilizamos o Request junto com o cabeçalho do site, desse modo ele permite fazermos a abertura e leitura da url. Podemos ainda colocar uma exceção para que ele exiba qual dos erros foi ocasionado:







* + 1. O headers foi feito através da inspeção de elementos:

