# Guilherme Sousa Inácio

# Teste para Desenvolvedor Python Júnior, com técnicas de WebScraping.

# 1 - Já teve contato com a biblioteca beautifulsoup?

R: Sim, ano passado estudei por conta, baseado em meu interesse em extração de dados e automação, as técnicas de webscraping, e me deparei com essa biblioteca, bem como outras bibliotecas em outras linguagens de programação. Possuo familiaridade com esse tipo de biblioteca pois utilizo a biblioteca pupetter do NodeJS que faz um trabalho semelhante de webscraping.

# 2 - Se sim, poderia explicar seu funcionamento aplicado?

R: Sim. A biblioteca permite que o desenvolvedor extraia dados de uma página web em formato de árvore, sendo muito mais rápido trabalhar dessa forma onde cada elemento HTML é entendido como um objeto do que utilizar outras técnicas mais complexas como RegExp (utilizado para buscas em strings que são textos).

# 3 - Já teve contato com a biblioteca scrapy?

R: Sim, em paralelo ao estudo da biblioteca beautifulsoup, também encontrei conteúdo relacionado ao framework scrapy.

# 4 - Se sim, poderia explicar seu funcionamento aplicado?

R: Sim, a biblioteca scrapy, assim como a beautifulsoup permite o webcrawling de páginas web que não possuem api aberta. Utilizando os chamados "Spiders" da biblioteca se pode se definir como será a navegação pelas páginas e como será manipulado o resultado obtido dessas páginas.

# 5 - Poderia desenvolver um webcrawler para a coleta no site <https://www.melodybrazil.com/>, para todos os CDs disponíveis no site. Os resultados devem ser salvos em uma planilha xls e conter:

# - url da publicação,

# - o nome da publicação,

# - data da publicação,

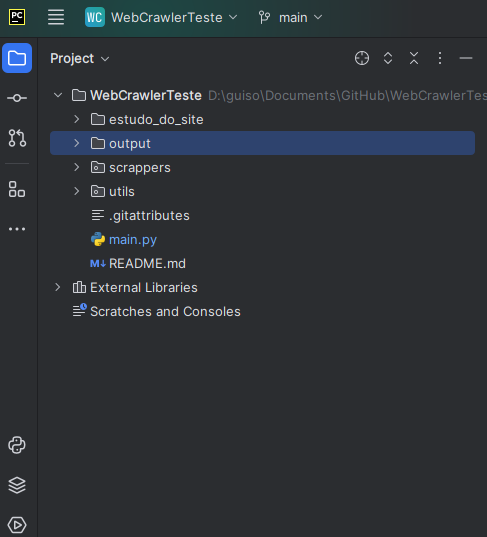
# - publicado por e

# - url de download do cd.

# 6 - Coloque em anexo na resposta deste e-mail, o web crawler desenvolvido e os resultados obtidos em formato xls.

# Sobre o WebScrapper:

## O programa é divido da seguinte forma.



Sendo “WebCrawlerTeste” o diretório raiz do projeto.

A pasta output é onde se encontram (ou se encontrarão após a execução do programa) os arquivos solicitados.

Dentro dessa pasta são gerados três arquivos, sendo eles:

* date\_info.json
* musics.json
* cds\_melody\_brasil.xlsx

# Funcionamento

Primeiramente, observei que a navegação entre as páginas do site Melody Brasil não era feita somente mediante ao número da página atual, mas sim por um parâmetro passado na URL da página chamado “updated-max”. A princípio pensei ser um formato datetime do python ( o que se comprovou posteriormente), e ao longo de minhas busca nos scripts do site e pesquisas para auxílio no projeto encontrei sobre o método encodeUriComponent(), que transforma o padrão isoformat de datas para um padrão de URLs. Dessa forma meu webscrapper coleta primeiramente essas datas, que são disponibilizadas na API do site. Cada música tem sua data de publicação, e cada página do site utiliza a décima música (contando a partir do index da primeira) para buscar em seu repositório sobre as próximas músicas.

O arquivo que faz tal procedimento no meu scraper chama-se “enconde\_uri.py”, sendo responsável por ao final de todo esse processo, gerar um array contendo dados para poder realizar crawling nas páginas reais do site.

Posteriormente o arquivo “get\_pages.py” realiza todo o processo restante de crawling, sendo responsável por extrair os dados sobre título da música, quem publicou, data e hora da publicação, url onde a publicação se encontra no site, o link de download da música e adicionalmente o link da thumbnail da música.

Todo o meu processo de entendimento de como crawlear o site foi baseado em scripts do próprio site, irei enviá-los junto a esse documento no arquivo zipado dentro da pasta “estudo do site”.