Web Scrapping

Projeto realizado para capturar indíce da Ibovespa e os 5 ativos com maior crescimento do dia.

No projeto foi utilizado o selenium para fazer a automação do Google Chrome para abrir as páginas onde se encontram as informações e capturar os dados. Na manipulação de dados (leitura, organização e exportação) foi utilizado o pacote pandas e para os arquivos externos foi considerado arquivos em formato Excel (.xlsx).

Qualquer dúvida, os sites de consulta para os comandos do selenium foram:

https://selenium-python.readthedocs.io/ - https://www.selenium.dev/documentation/

Criado por Guilherme Augusto Magalhães para fins acadêmicos em 10/10/2022

Contato: guiiimagalhaes@gmail.com

Instalação do selenium

Executar apenas na primeira vez, caso não tenha o pacote selenium

In [1]:
```
#!pip install selenium
```

Ajustes iniciais

Importação dos pacotes

Os pacotes utilizados do Selenium foram os pacotes apontados conforme documentação. Além deles, foi utilizado o pacote pandas para manipulação dos data frames e o pacote datetime para registrar a data que foi atualizado

In [2]:
```
from selenium import webdriver #Controlar o navegador
from selenium.webdriver.common.keys import Keys #Escrever dados no navegador
from selenium.webdriver.common.by import By #Selecionar itens do navegador

import pandas as pd

from datetime import date
```

Ajuste da data

In [3]:
```
date = date.today()
date = date.strftime("%d/%m/%Y")
```

Leitura dos arquivos com dados passados ou criação de tabela nova

Na primeira execução, não irá existir nenhum arquivo histórico. Para esse caso, foi criado uma estrutura de try/except para verificar se existem os arquivos com dados históricos e caso contrário criar os data frames novos.

In [4]:
```
try:
    dfIndices = pd.read_excel("Indices.xlsx")
    dfAcoes = pd.read_excel("Acoes.xlsx")
except:
    names = ["Data", "Indice", "Pontos", "Variacao"]
    dfIndices = pd.DataFrame(columns = names)
    names = ["Data", "Ativo", "Valor", "Variacao Dia", "Variacao Semana", "Variacao
    dfAcoes = pd.DataFrame(columns = names)
```

Web Scrapping

Dados do Ibovespa

Abrindo o navegador e acessando o site

In [5]:
```
navegador = webdriver.Chrome()
    #Comando para abrir um novo navegador controlado pelo selenium

navegador.get("https://www.b3.com.br/pt_br/")
    #Comando para acessar o site informado
    #Nesse caso, para extrair o índice da Ibovesta foi utilizado o site da B3
```

Fazendo a leitura dos dados

Para leitura dos dados, foi utilizado o método 'find_element' do webdriver.

Ele requisita dois parâmetros: tipo de elemento e tag do elemento.

```
-Tipo de elemento: Foi utilizado o XPATH devido o mesmo ser fácil de
encontrar e configurar.
-Tag do elemento: Basta abrir a janela de inspeção do navegador e,
clicando com o botão direito, selecionar Copy -> XPath, no painel de
inspeção dos elementos da página HTML.
```

In [6]:
```
#Leitura dos dados
variacao = navegador.find_element(By.XPATH, "//div[@id='ibovPct']").text
pontos = navegador.find_element(By.XPATH, "//div[@id='ibovPts']").text
```

Verificação dos dados

Ao executar as primeiras leituras, pode=se verificar que os dados vieram com os símbolos (% e 'pts') e no formato texto (string), conforme registrado abaixo:

print(str(variacao) + " " + str(type(variacao))) -> -0,37% <class 'str'>

print(str(pontos) + " " + str(type(pontos))) -> 115.940pts <class 'str'>

Para esse caso foi retirado os símbolos e ajustado os valores:

```
-Para a variação foi trocado a vírgula por ponto e transformado em
float (número decimal)
-Para o valor foi retirado o ponto e transformado em int (número
inteiro)
```

In [7]:
```
variacao = variacao.replace("%","")
variacao = variacao.replace(",",".")
variacao = float(variacao)

pontos = pontos.replace("pts","")
pontos = pontos.replace(".","")
pontos = int(pontos)
```

Após essa transformação, pode-se verificar que os dados ficaram configurados corretamente:

print(str(variacao) + " " + str(type(variacao))) -> -0.37 <class 'float'>

print(str(pontos) + " " + str(type(pontos))) -> 115940 <class 'int'>

Registro dos dados na tabela

Com os dados prontos em variáveis, basta fazer um append (acrescentar no fim) no data frame com os dados dos Índices

In [8]:
```
dfIndices = dfIndices.append({'Data': date, 'Indice': "Ibovespa", 'Pontos': pontos,
```

Dados das ações

Abrindo o navegador e acessando o site

In [9]:
```
navegador.get("https://www.infomoney.com.br/ferramentas/altas-e-baixas/")
    #Comando para acessar o site informado
```

No site informado, existe uma tabela organizada decrescentemente pela variação do dia com todas as ações da bolsa, contendo algumas informações a respeito dos ativos.

Para esse caso, foram extraídos os dados dos 5 ativos com maior variação positiva no dia. Caso seja necessário alterar a quantidade de ativos a serem verificados, basta mudar a variável abaixo

In [10]:
```
r = 5
```

Fazendo a extração, transformação e carregamento dos dados

Como será extraído mais que apenas um ativo, foi criado um loop for para o processo, de acordo com o número configurado na linha acima.

Extração

Foi utilizado o mesmo método anteriormente utilizado, com um pequeno ajuste na tag do elemento para que o mesmo percorra as linhas da tabela.

Transformação

Conforme verificado na primeira extração, os dados necessitam de alguns ajustes. Nessa tabela não existe nenhum símbolo junto do valor, mas deve ser necessário alterar a vírgula para ponto para que o pandas reconheça corretamente.

Carregamento

Também foi utilizado um append no data frame para carregar todos os dados dos ativos, fazendo a conversão para float no momento de carregamento..

In [11]:
```
for i in range(r):
    #Extração dos dados
    ativo = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr[' +
    valor = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr[' +
    varDia = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr['
    varSem = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr['
    varMes = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr['
    varAno = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr['
    var12Mes = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr[
    valMin = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr['
    valMax = navegador.find_element(By.XPATH, '//*[@id="altas_e_baixas"]/tbody/tr['

    #Transformação dos dados
    valor = valor.replace(",",".")
    varDia = varDia.replace(",",".")
    varSem = varSem.replace(",",".")
    varMes = varMes.replace(",",".")
    varAno = varAno.replace(",",".")
    var12Mes = var12Mes.replace(",",".")
    valMin = valMin.replace(",",".")
    valMax = valMax.replace(",",".")

    #Consolidação dos dados
    dfAcoes = dfAcoes.append({'Data': date,
                              'Ativo': ativo,
                              'Valor': float(valor),
                              'Variacao Dia': float(varDia),
                              'Variacao Semana': float(varSem),
                              'Variacao Mes': float(varMes),
                              'Variacao Ano': float(varAno),
                              'Variacao 12 meses': float(var12Mes),
                              'Valor Minimo': float(valMin),
                              'Valor Maximo': float(valMax)
                             }, ignore_index = True)
```

Verificação e exportação dos dados

Verificação do dataframe com dados dos ativos

In [12]:
```
dfAcoes
```

Out[12]:

	Data	Ativo	Valor	Variacao Dia	Variacao Semana	Variacao Mes	Variacao Ano	Variacao 12 meses	Valor Minimo	Valor Maximo
0	10/10/2022	SLCE3	44.73	6.04	6.05	3.33	14.52	8.63	42.54	45.34
1	10/10/2022	SMTO3	27.67	5.53	5.53	8.42	-18.11	-19.53	26.17	27.87
2	10/10/2022	BRFS3	14.55	5.13	5.13	12.97	-35.39	-44.74	13.86	14.60
3	10/10/2022	JBSS3	25.68	4.81	4.82	2.23	-30.41	-27.46	24.46	25.75
4	10/10/2022	BEEF3	13.20	4.18	4.18	5.26	29.74	28.84	12.56	13.29

Verificação do dataframe com dados da Ibovespa

In [13]:
```
dfIndices
```

Out[13]:

	Data	Indice	Pontos	Variacao
0	10/10/2022	Ibovespa	115940	-0.37

Exportação dos dataframes para arquivos em Excel

In [14]:
```
dfIndices.to_excel("Indices.xlsx", index=False)
dfAcoes.to_excel("Acoes.xlsx", index=False)
```