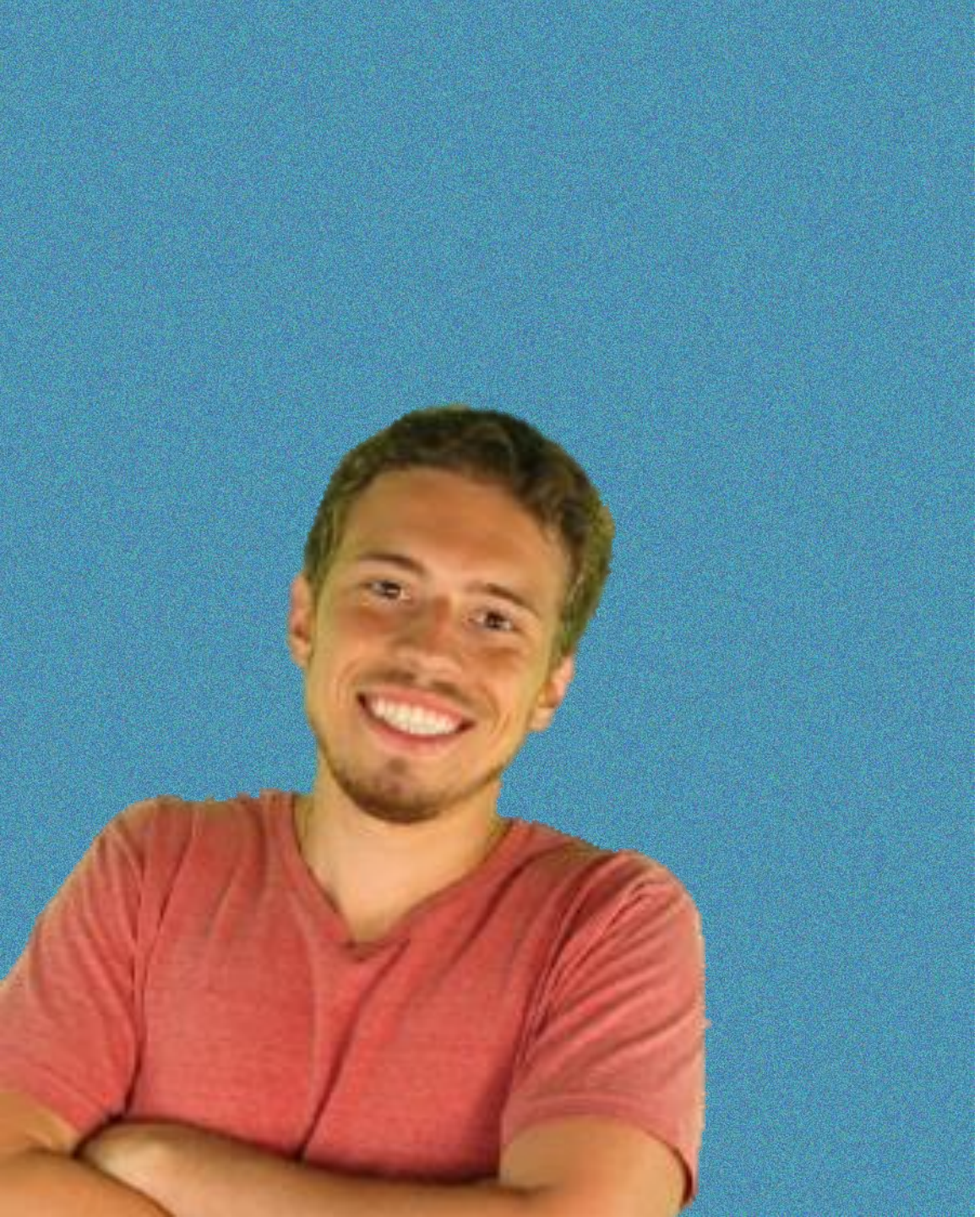


INTENSIVÃO DE PYTHON

100% ONLINE & GRATUITO

Apostila Completa
Aula 4

Aprenda como automatizar processos que tenham interface com a internet

Impressionador do absoluto zero!

Parte 1

Introdução

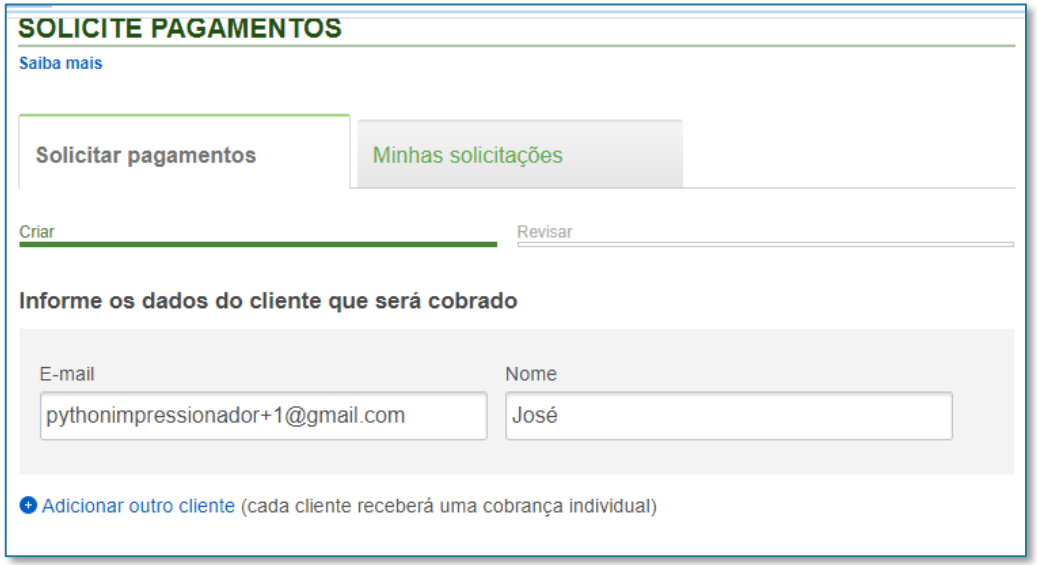
O que vamos aprender

Na quarta aula do Intensivão do Python você vai aprender a criar um código para automação de processos. No dia a dia das empresas, é muito comum que existam operações extremamente manuais que além de repetitivas (chatas) são suscetíveis a erro visto que são feitas manualmente. Vamos aprender como criar um código que você possa resolver esse problema sem nem tocar no mouse ☺. Aprenda como **fazer uma uma automação com integração web** com os conceitos abaixo:

- Importando dados de bases .csv
- Jupyter Notebook
- Importando bibliotecas
- Webdriver
- Usando Selenium

Após todos esses conhecimentos, seremos capazes de transformar um processo extremamente operacional

... em processo automático e sem erros! Tudo graças a você! ☺

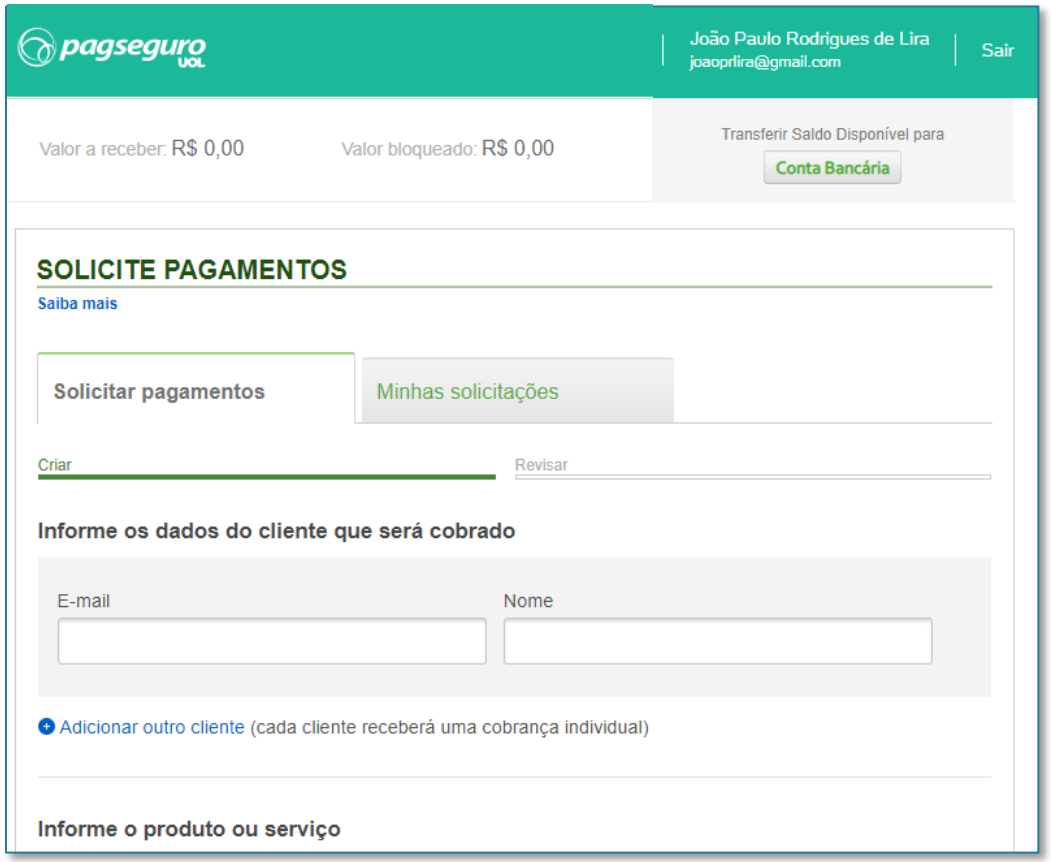


Entendendo o problema

Imagine que você trabalhe no setor de “contas a pagar” de uma empresa.
Os clientes possuem 30 dias para pagar, no entanto, os números de inadimplência vem crescendo nos últimos meses e entrar em contato com esses clientes tem sido cada vez mais demorado e “braçal”.
Essas cobranças são feitas pelo pagseguro e as pessoas que estão inadimplentes estão na nossa base excel.

Nosso objetivo é **automatizar** esse processo para reduzir a carga braçal dos trabalhos

	A	B	C	D	E
1	Cliente	Nome	Valor Pago	Valor Total Devido	Email
2	1,1122E+10	José	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+1@gmail.com
3	1,1122E+10	Maria	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+2@gmail.com
4	1,1122E+10	João	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+3@gmail.com
5	2,2233E+10	Antônio	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+4@gmail.com
6	2,2233E+10	Francisco	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+5@gmail.com
7	2,5567E+10	Carlos	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+6@gmail.com
8	2,89E+10	Paulo	R$ 8.341,00	R$ 8.950,00	pythonimpressionador+7@gmail.com



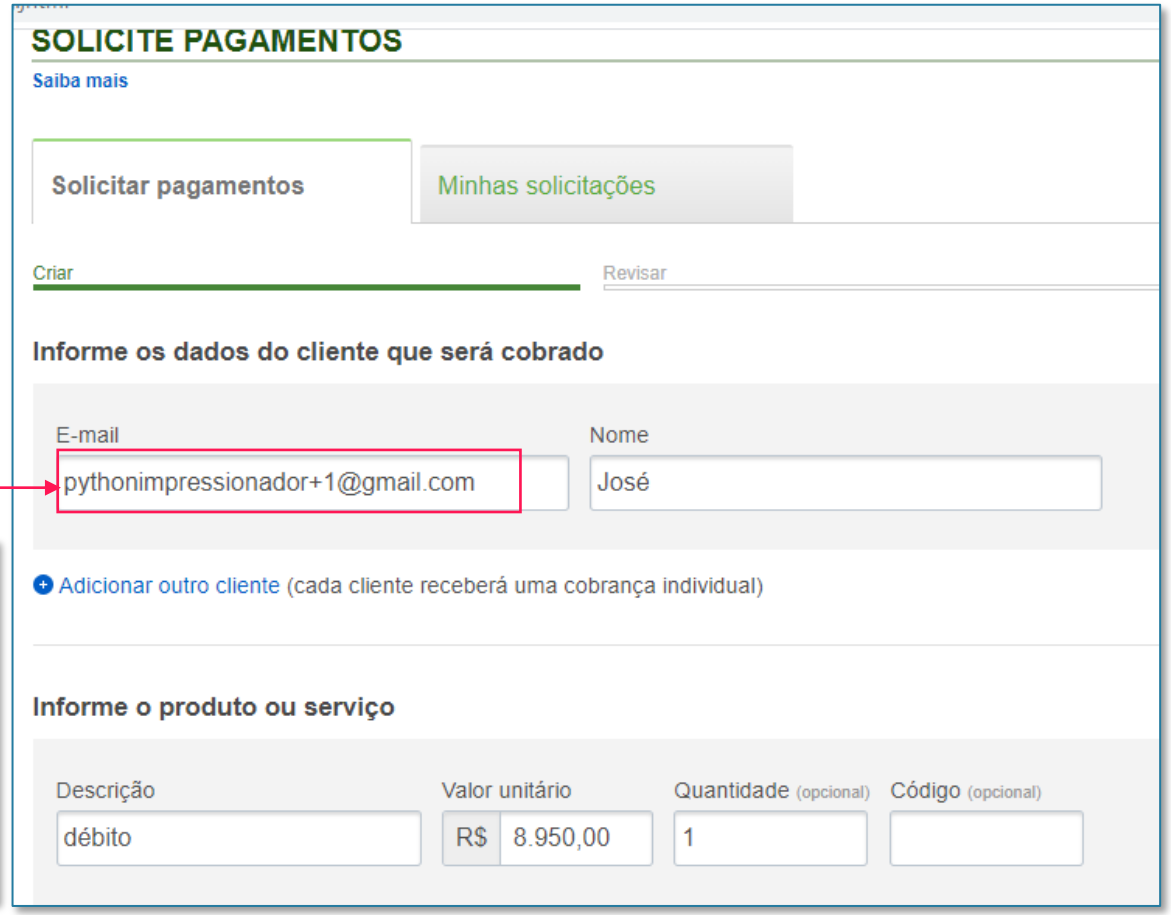
Entendendo a solução final

Nossa solução final será:

1) Pegar os dados de clientes inadimplentes na base de dados;
2) Entrar no site do PagSeguros;
3) Preencher os campos solicitados pelo site;
4) Emitir a cobrança.

Tudo isso automaticamente!! Apenas rodando o código que vamos criar.

	A	B	C	D	E
1	Cliente	Nome	Valor Pago	Valor Total Devido	Email
2	1,1122E+10	José	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+1@gmail.com
3	1,1122E+10	Maria	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+2@gmail.com
4	1,1122E+10	João	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+3@gmail.com
5	2,2233E+10	Antônio	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+4@gmail.com
6	2,2233E+10	Francisco	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+5@gmail.com
7	2,5567E+10	Carlos	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+6@gmail.com
8	2,89E+10	Paulo	R$ 8.341,00	R$ 8.950,00	pythonimpressionador+7@gmail.com



Parte 2

Importando bibliotecas

Importando base de dados (1/3)

Como vimos na aula 1 do Intensivão, vamos usar bibliotecas que nos facilitem importar dados de planilhas Excel, arquivos .csv, etc.

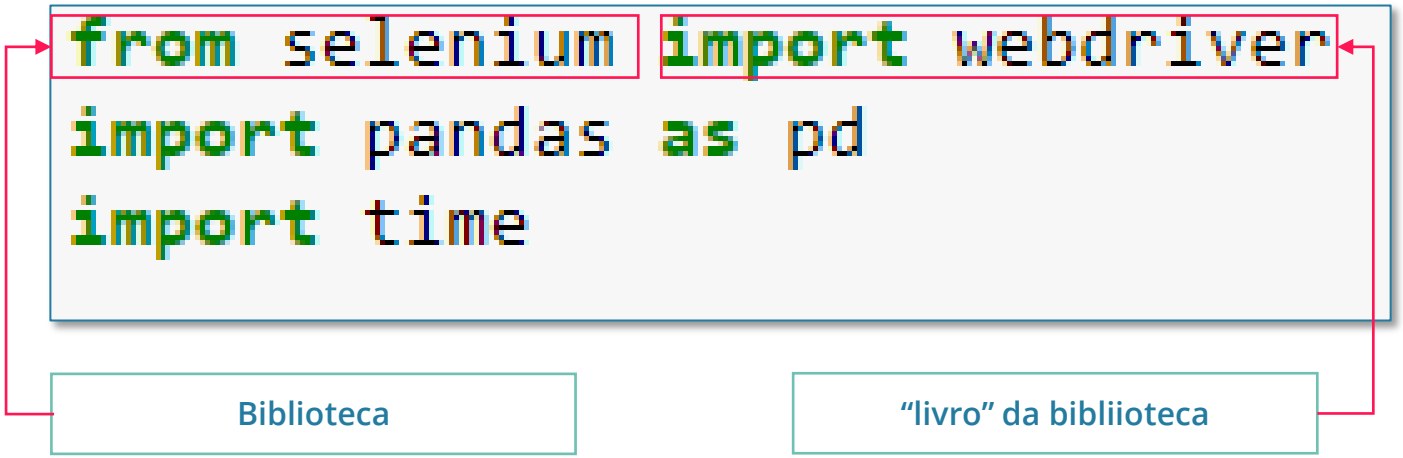
Novamente usaremos o PANDAS. Caso você não saiba do que estamos falando aqui, dá uma olhadinha na apostila da Aula 1 do intensivão!! Lá a gente explica o que são bibliotecas e para que servem ☺.

No entanto, além do pandas iremos importar mais duas bibliotecas: time e selenium

Antes de entendermos no detalhe o que elas fazem e para que servem, vamos nos atentar a uma diferença na hora da importação.

Podemos perceber que na primeira linha importamos o selenium utilizando a estrutura from antes do import.

Essa estrutura significa dizer que estamos definindo qual a biblioteca e vamos dizer assim o "livro" dessa biblioteca.



Importando base de dados (2/3)

Como assim, um livro de uma biblioteca?

Antes de entendermos no python, imagine uma biblioteca de fato.

Grande, correto? Muitas das informações ali, são úteis mas não naquele momento...

Possivelmente só vamos conseguir ler 1, 2 ou 3 livros de uma vez. Não faz sentido alugar toda a biblioteca, apenas o que vamos precisar.

Aqui é exatamente a mesma coisa! Não temos que importar toda uma biblioteca se apenas uma parte dessa biblioteca nos interessa.

No nosso caso, o webdriver(vamos explicar melhor o que é, e para que serve mais a frente).

Isso torna nosso código mais simples e rápido!

Importando base de dados (3/3)

Para essa aula estamos usando o Jupyter e nele já possuímos uma série de pacotes "pré-instalados" mas as vezes precisamos instalar pacotes adicionais.

O Python, possui um "instalador embutido" que se chama pip.

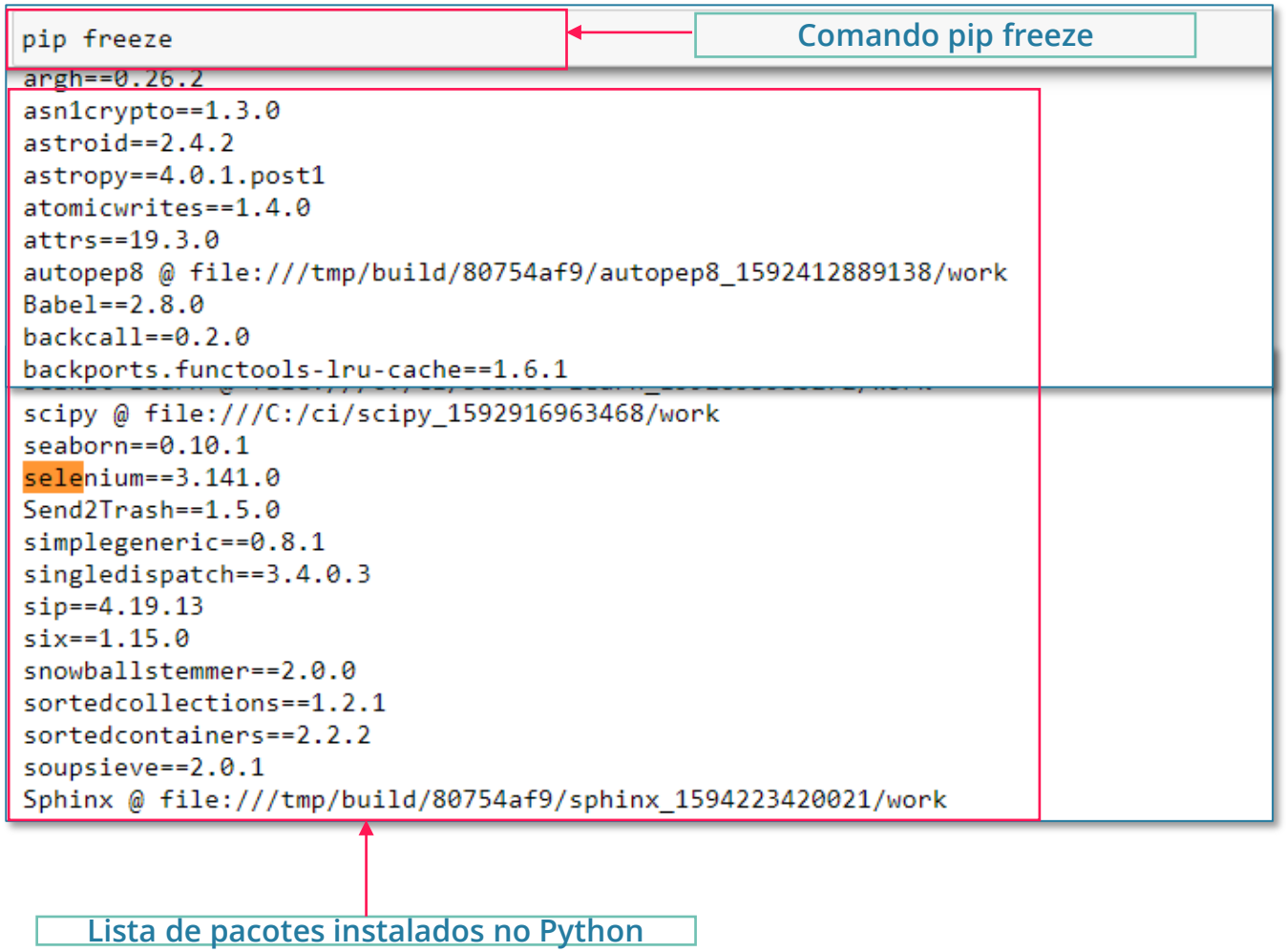
Por ele, é possível, instalar, desinstalar pacotes.

Caso, você queira saber se o selenium está instalado basta usar o comando abaixo em uma das células do Jupyter:

pip freeze

Caso não encontre na lista o selenium, use o comando abaixo para instalar:

pip install -U selenium



Parte 3

Importando os dados

INTENSIVÃO DE PYTHON
100% ONLINE & GRATUITO

Importando os dados com o pandas

Agora que já importamos nossas bibliotecas, vamos importar nossos dados da base de excel “Clientes Pagamento.xlsx”.

Como vimos na aula 1 do intensivão, para importarmos dados de planilhas excel usamos o comando pd.read_excel.

Lembrando que essa importação precisa ser “armazenada” em uma variável. No nosso caso usaremos a variável clientes_df .

	A	B	C	D	E
1	Cliente	Nome	Valor Pago	Valor Total Devido	Email
2	1,1122E+10	José	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+1@gmail.com
3	1,1122E+10	Maria	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+2@gmail.com
4	1,1122E+10	João	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+3@gmail.com
5	2,2233E+10	Antônio	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+4@gmail.com
6	2,2233E+10	Francisco	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+5@gmail.com
7	2,5567E+10	Carlos	R$ 8.950,00	R$ 8.950,00	pythonimpressionador+6@gmail.com
8	2,89E+10	Paulo	R$ 8.341,00	R$ 8.950,00	pythonimpressionador+7@gmail.com

```
from selenium import webdriver
import pandas as pd
import time


clientes_df = pd.read_excel('Clientes Pagamento.xlsx', dtype={'Cliente': object})
```

- Variável que receberá os dados importados
- Função de importação de dados
- Nome da base de dados
- Define que a coluna ‘Cliente’ será do tipo objeto

Visualizando os dados importados

Após a importação dos dados podemos usar o **display()** para visualizarmos os dados e se a importação foi feita corretamente.

```
from selenium import webdriver
import pandas as pd
import time


clientes_df = pd.read_excel('Clientes Pagamento.xlsx', dtype={'Cliente': object})
display(clientes_df)
```

Função display que apresenta os dados armazenados

	Cliente	Nome	Valor Pago	Valor Total Devido	Email
0	11122233312	José	8950	8950	pythonimpressionador+1@gmail.com
1	11122233345	Maria	8950	8950	pythonimpressionador+2@gmail.com
2	11122233367	João	8950	8950	pythonimpressionador+3@gmail.com
3	22233344456	Antônio	8950	8950	pythonimpressionador+4@gmail.com
4	22233344484	Francisco	8950	8950	pythonimpressionador+5@gmail.com
5	25566677829	Carlos	8950	8950	pythonimpressionador+6@gmail.com

Parte 3

Interface com uma página na web

INTENSIVÃO DE PYTHON {#}
100% ONLINE & GRATUITO

Selenium

Conforme explicamos anteriormente o **Selenium** é uma biblioteca assim como o Pandas.

Essa biblioteca é muito utilizada para a interface com a internet. Ela funciona como um robô que clica, insere dados, etc em páginas WEB como se você estivesse fazendo.

Muito útil para processos repetitivos como este que temos aqui.

Aqui temos links de documentação para aqueles que gostariam de se aprofundar no tema e funcionalidades:

https://pypi.org/project/selenium/

https://www.selenium.dev/documentation/en/

Selenium – webdriver (1/3)

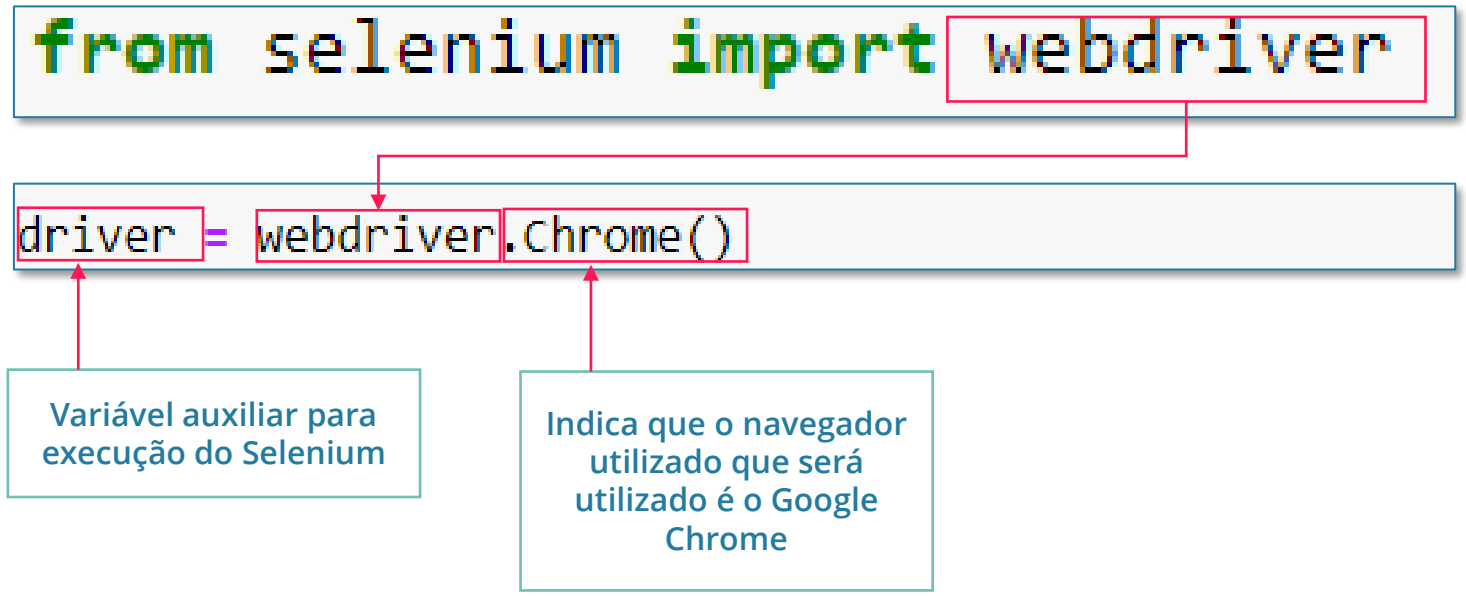
Vamos para nosso código. O momento agora é criar um código que nos permita acessar o pagseguro, logar e imputar os dados da nossa base.

Se você se lembra do nosso passo anterior, vai perceber que estamos usando o webdriver que importamos anteriormente.

O webdriver possui uma particularidade. Ele precisa ser instalado e colocado dentro da **MESMA pasta** que o Python possui seu executável. O próximo slide será só sobre isso, então não se preocupe.

Voltando para nosso código, podemos perceber que criamos uma variável **driver**. Ela nos permitirá armazenar e transitar nos diferentes campos que encontraremos nas páginas WEB.

Outro ponto é que vemos .Chrome(). Isso nos indica que acessaremos essas páginas via Google Chrome.



Selenium – webdriver (2/3)

Antes de continuarmos vamos instalar o webdriver.

Passo1: Entre no link para outros navegadores acessem os links ao lado.

Chrome:	https://sites.google.com/a/chromium.org/chromedriver/downloads
Edge:	https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/
Firefox:	https://github.com/mozilla/geckodriver/releases
Safari:	https://webkit.org/blog/6900/webdriver-support-in-safari-10/

Passo2: Baixe a última versão.

Current Releases

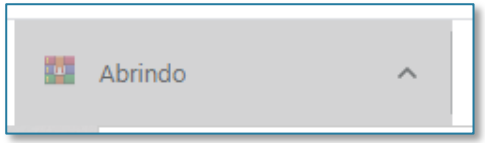
- If you are using Chrome version 89, please download ChromeDriver 89.0.4389.23
- If you are using Chrome version 88, please download ChromeDriver 88.0.4324.96
- If you are using Chrome version 87, please download ChromeDriver 87.0.4280.88
- For older version of Chrome, please see below for the version of ChromeDriver that supports it.

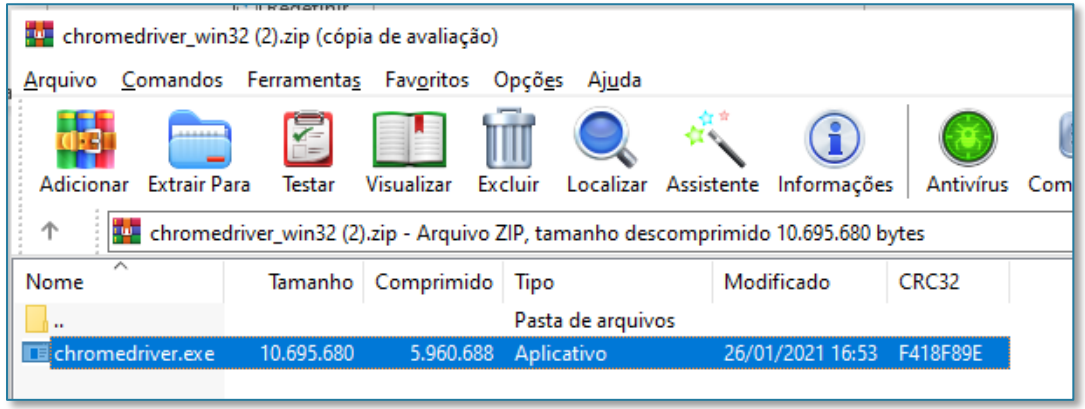
Name	Last modified
Parent Directory	
chromedriver_linux64.zip	2021-01-28 17:30:52
chromedriver_mac64.zip	2021-01-28 17:30:53
chromedriver_mac64_m1.zip	2021-01-28 17:30:55
chromedriver_win32.zip	2021-01-28 17:30:57
notes.txt	2021-01-28 17:31:00

Escolha seu sistema operacional.

Nesse caso Windows

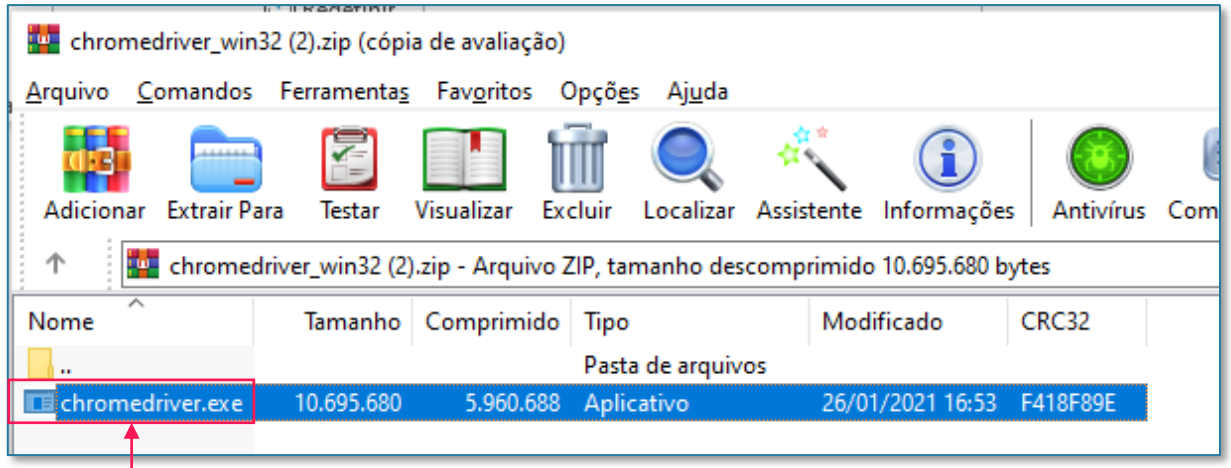
Passo3: Abra o arquivo baixado





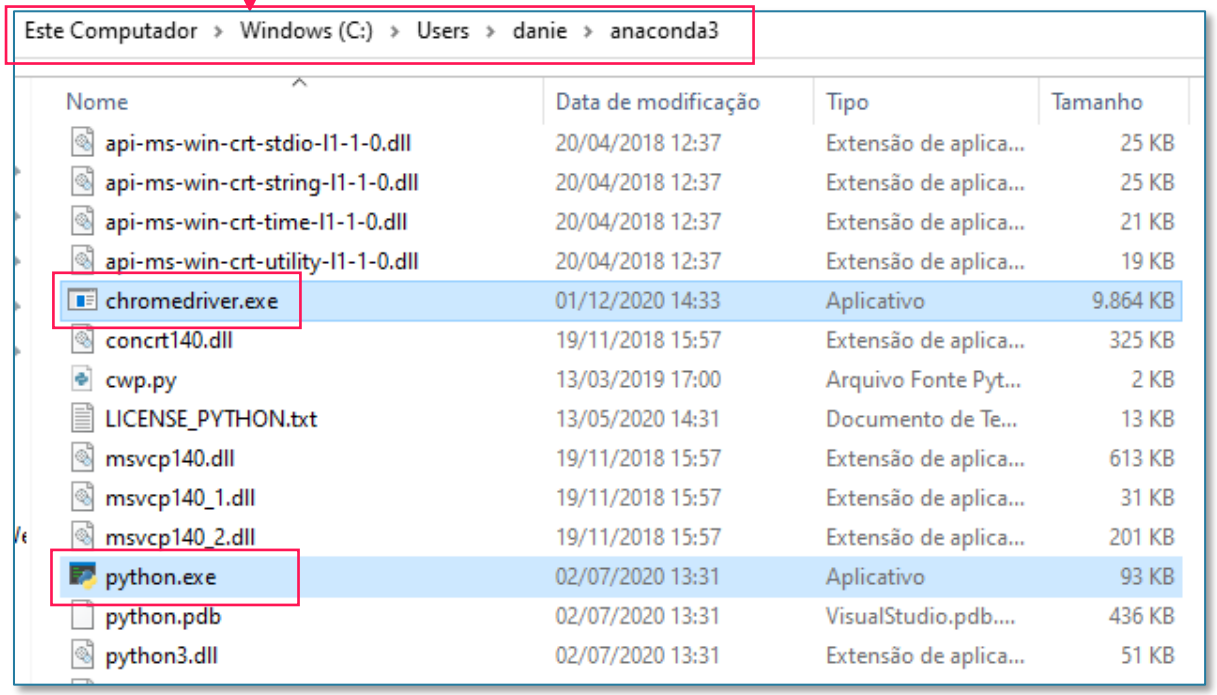
Selenium – webdriver (3/3)

Passo4: Coloque o arquivo **chromedriver.exe** na mesma pasta que o arquivo **python.exe**



Extraia o arquivo chromedriver.exe

Pasta que contém o arquivo Python.EXE. ATENÇÃO!!!
Este caminho pode variar!!!



Acessando o site e colocando login e senha (1/2)

Agora que temos o web driver instalado podemos usá-lo para acessar o site do pagseguro.

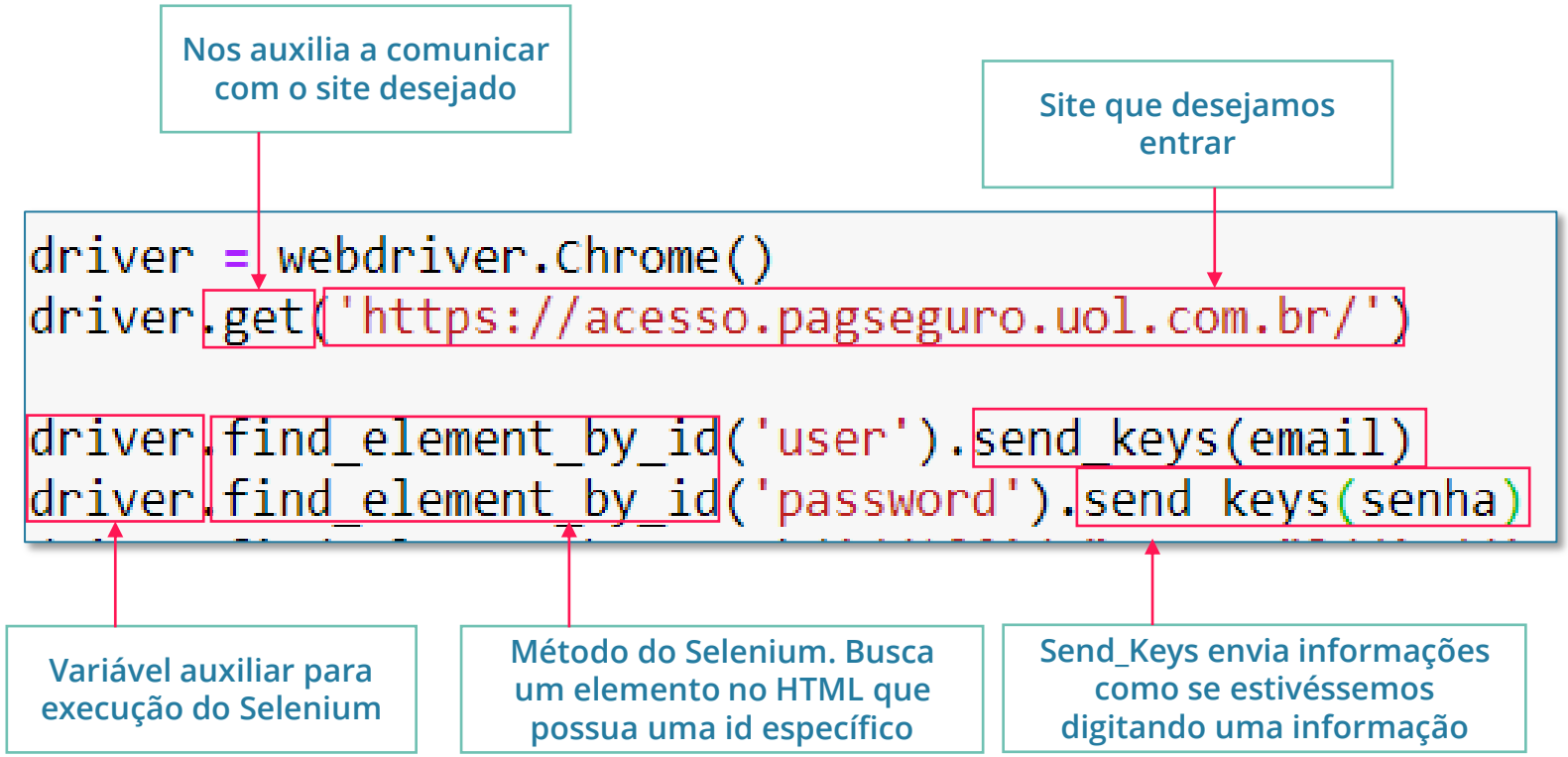
Como podemos ver, para realizar essa tarefa, vamos utilizar o método **.get(site desejado)**.

Ao executarmos essa parte do código uma nova janela do navegador será aberta e a url indicada será carregada.

Como não estamos logados no site, vamos precisar acessar os campos de login e senha e colocar nossos dados.

Mas como o Selenium sabe onde clicar? Usando o métdo **find_element_by_id**. Vamos entender agora, o que significa esse método.

Obs: Para efeito de apostila utilizamos 'email' e 'senha' por uma questão de segurança de dados. No seu caso, é possível colocar diretamente o login e senha.



Acessando o site e colocando login e senha (2/2)

Para entendermos o método find_elemento_by_id, precisamos entender um pouco sobre como é construída uma página e o que significa este 'id' do método.

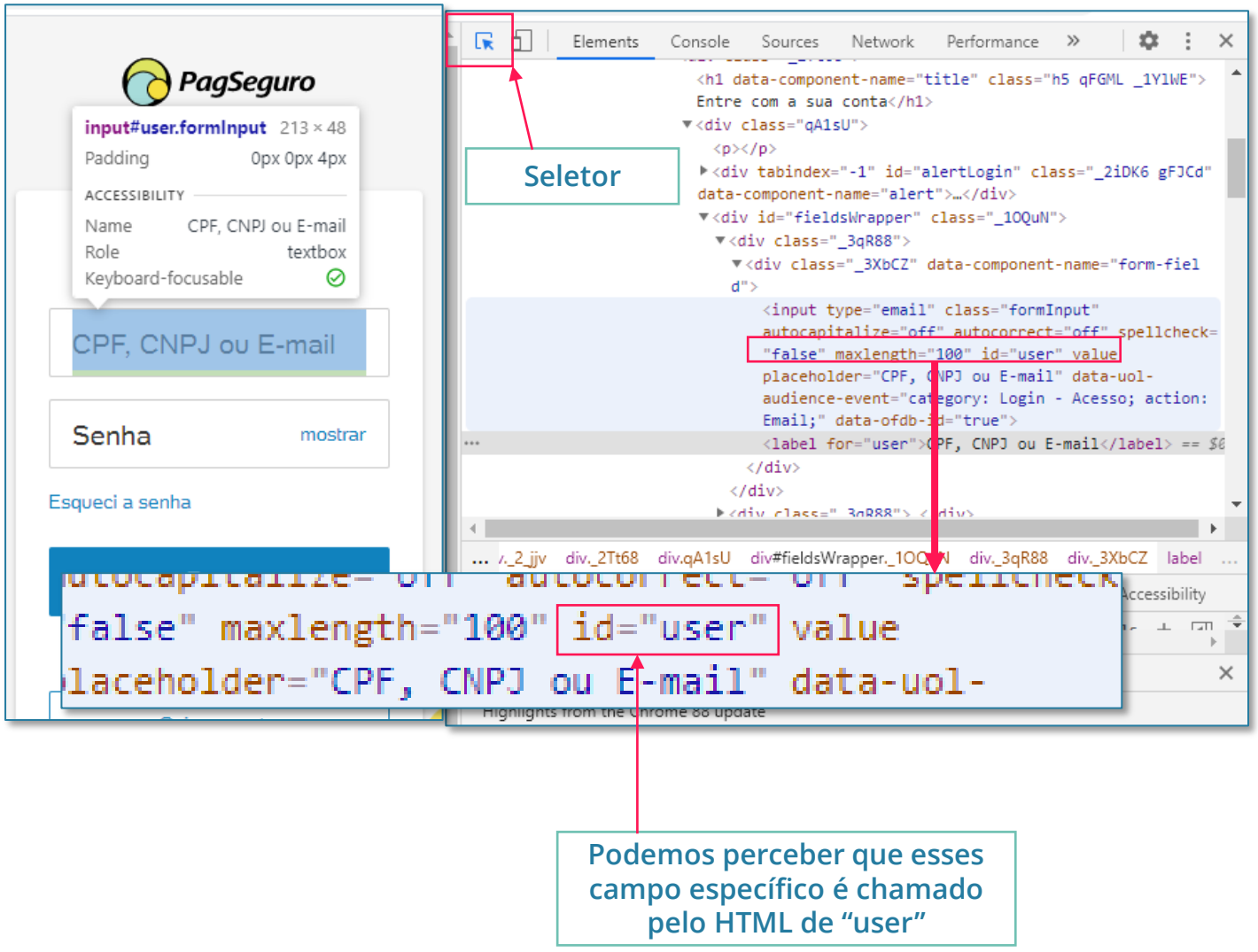
Páginas Web são criadas em 99,9% dos casos com o uso do HTML que é outra linguagem de programação mas com propósitos bem distintos do Python.

Se você clicar com o botão direito do mouse na página e clicar em INSPECIONAR, esta janela CHEIA de códigos aparecerá para você.

Este é o HTML desta página. Em outras palavras, como ela é construída.

Nosso objetivo é descobrir algo que seja único para podermos indicar ao Selenium onde ir.

Para isso podemos usar a função SELETOR indicada na imagem e o código ID que é bem comum de encontrarmos nos HTML que foram bem feitos.



Acessando página *Solicite pagamentos*

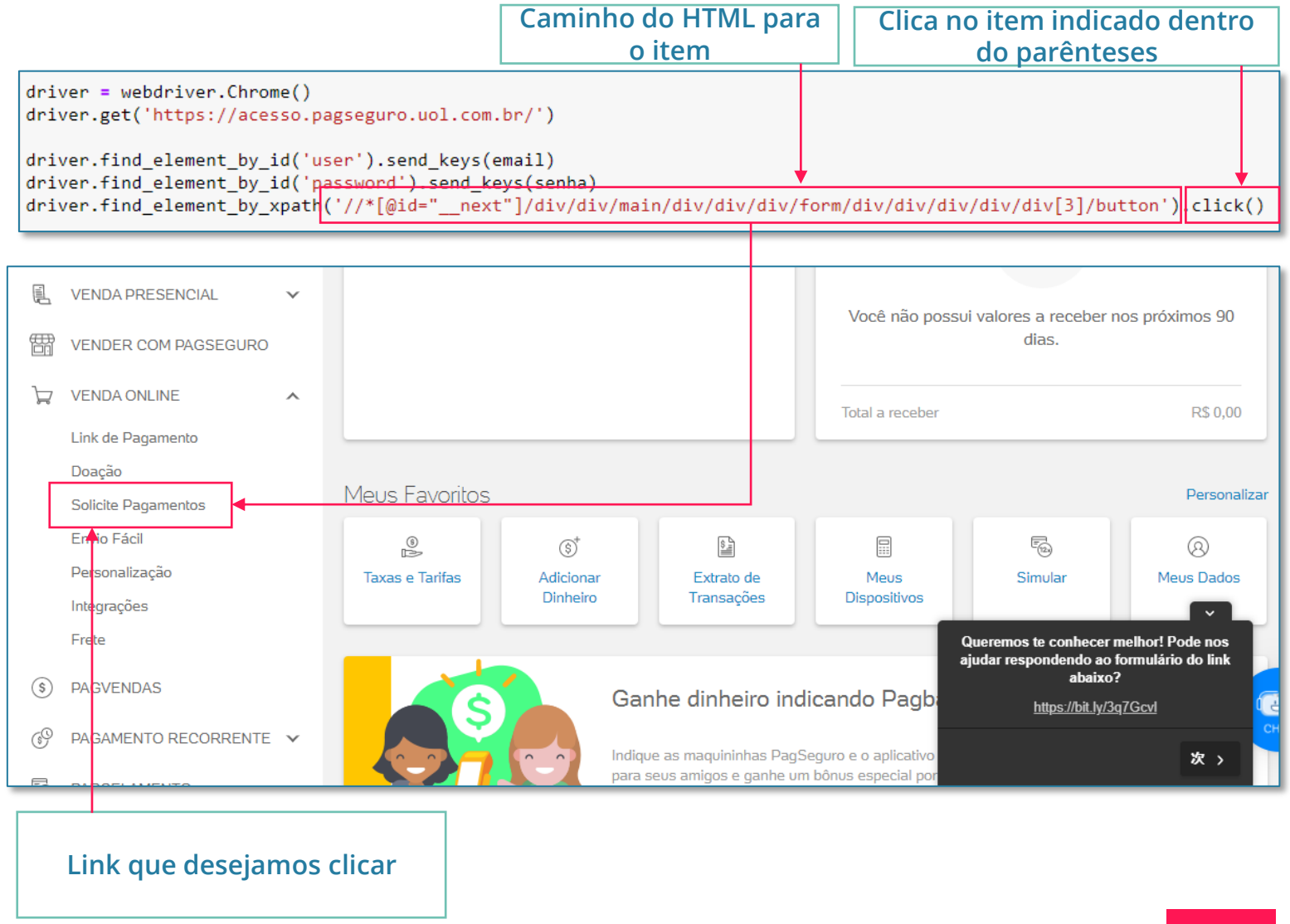
Se rodarmos estas duas linhas de código entraremos no pagseguro com a página ao lado.

Agora, precisamos fazer a mesma etapa anterior mas buscando agora o campo **venda online>Solicite pagamentos**

No Selenium temos alguns métodos distintos que nos ajudam nessa interação com a página. Além do find_element_by_id, temos também o **find_element_by_xpath.**

Em comparação com o by_id este pode ser um pouco mais confuso, no entanto segue a mesma lógica de buscar no html da página algo que seja único. Nesse caso, usamos o caminho seguindo a estrutura do HTML.

Por fim, temos o **.click()** que é exatamente o que está escrito. Clica no item que indicamos dentro do parênteses.



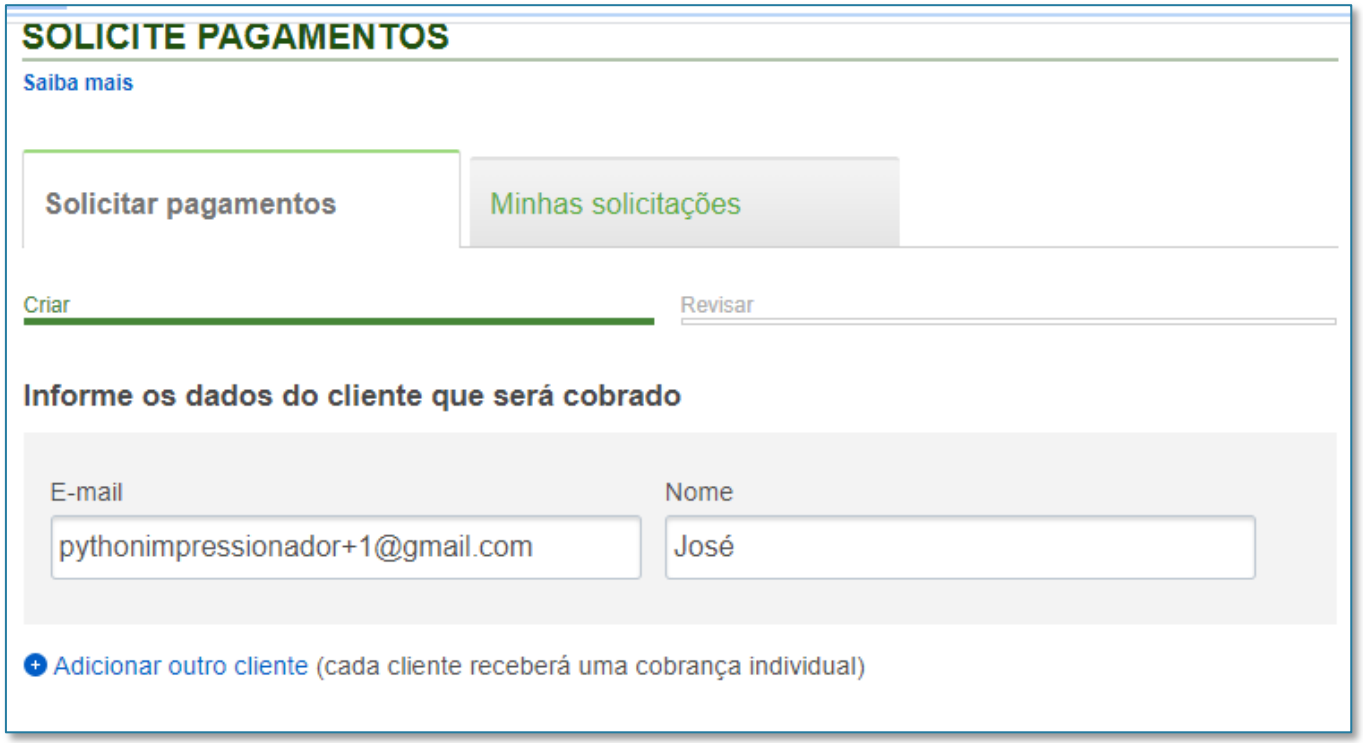
Acessando página *Solicite pagamentos*

Entrando na página de solicitar pagamentos, vamos unir agora o Selenium a nossa base de dados importada via Pandas.

Para isso, vamos precisar de uma estrutura de repetição que nos permite preencher o formulário e solicitar o pagamento para cada um dos nomes na nossa base.

Quando falamos de estrutura de repetição o mais usado é o **FOR**.

Além disso, usaremos o webdriver para indicar qual o campo receberá cada uma das informações.



Estrutura de repetição

Vamos agora entender o código dentro do FOR.

Percorre todos os clientes na base clientes_df coletando a linha

Calcula a dívida do cliente baseado na linha do for

```
for linha in clientes_df.index:
    divida = clientes_df.loc[linha, 'Valor Total Devido'] - clientes_df.loc[linha, 'Valor Pago']
    if divida > 0:
        driver.get('https://pagseguro.uol.com.br/operations/charging.jhtml')
        email = clientes_df.loc[linha, 'Email']
        driver.find_element_by_xpath('//*[@id="newRequestForm"]/div[1]/section[1]/div/fieldset[1]/input').send_keys(email)
        nome = clientes_df.loc[linha, 'Nome']
        driver.find_element_by_xpath('//*[@id="newRequestForm"]/div[1]/section[1]/div/fieldset[2]/input').send_keys(nome)
        driver.find_element_by_xpath('//*[@id="newRequestForm"]/div[1]/section[2]/div/fieldset[1]/input').send_keys('Cobrança pag
        valor = divida
        texto_valor = f'{valor:.2f}'
        driver.find_element_by_xpath('//*[@id="newRequestForm"]/div[1]/section[2]/div/fieldset[2]/div/input').send_keys(texto_val
        driver.find_element_by_id('sendNewCharging').click()
        time.sleep(0.5)
        driver.find_element_by_id('buttonConfirmSubmit').click()
        time.sleep(0.5)
```

Checa se há divida ou não

Preenche campo de valor da dívida

Submete a solicitação

Preenche os campos de informações do cliente

Acabou

Pronto! Agora é tomar seu café enquanto seu código trabalha para você ☺.

OBS: Não é o Lira!

INTENSIVÃO DE PYTHON

100% ONLINE & GRATUITO

Ainda não segue a gente no Instagram e nem é inscrito no nosso canal do Youtube? Então corre lá!

@hashtagprogramacao

youtube.com/hashtag-programacao