Como criar seu primeiro teste automatizado com Selenium WebDriver?

Bom, falando aqui de iniciante para iniciante eu vou tentar diminuir os gargalos de aprendizagem que tive durante o estudo do Selenium Webdriver em Java. Mas antes disso, vamos esclarecer algumas coisas aqui.

O que eu consigo automatizar com o Selenium Webdriver?

Então, em geral Testes de Interface do usuário, ou seja, você consegue simular a manipulação da página como se fosse um usuário normal.

Tá, mas para que eu vou querer automatizar isso se eu já faço manualmente?

Então, automatizando os principais fluxos de execução da sua aplicação, você traz uma certa tranquilidade para a sua equipe, visto que, toda vez que tiver um build do projeto, a automação vai rodar e garantir que aquilo que já funcionava antes continua funcionando e isso dá mais segurança a toda equipe. Além disso, vale lembrar que, nós como Homo Sapiens, estamos sujeitos a temperamentos, as vezes você pode não estar 100% no dia, ou simplesmente se sente tão confortável com o sistema sempre indo tudo certinho que

você deixa de testar uma parte importante, bom, é ai que entra o problema, isso pode gerar certos prejuízos a equipe.

O que é recomendado que eu automatize com o Selenium Webdriver?

O que o usuário deveria fazer no seu sistema? Criar um questionário? Fazer uma compra? Enviar um arquivo? Enfim, depende muito de sistema para sistema, mas eu vou considerar aqui o seguinte: Abrir a página, realizar o login(ou tentar), clicar em um botão, preencher um campo e capturar um texto que são os comandos mais utilizados e a automação desse tipo de teste gira basicamente nisso, cabendo a você somente saber organizar.

Quais conteúdo eu já devo ter uma noção para poder compreender e desenvolver com facilidade a automação?

· Conhecimento básico na linguagem Java

· Conhecimento básico em HTML e CSS

· Ter a JDK do Java já instalada

· Um pouquinho de inglês também ajuda, mas nada fora do normal

Preparando o ambiente

No seu Google Chrome vá na opção Ajuda e Sobre o Google Chrome, lá vai informar a versão do seu Chrome, o meu por exemplo está na versão 84.0.4147

Sabendo isso, vá agora no site do chromedriver https://chromedriver.chromium.org/downloads e realize o download da versão do chromedriver igual à do seu. Irá vir um arquivo .rar, você extrai e terá um executável.

Para facilitar você vai criar uma pasta no seu diretório C: com o nome que você preferir, eu aqui irei chamar de "projetinho", dentro dessa pasta você coloca o executável do chromedriver.

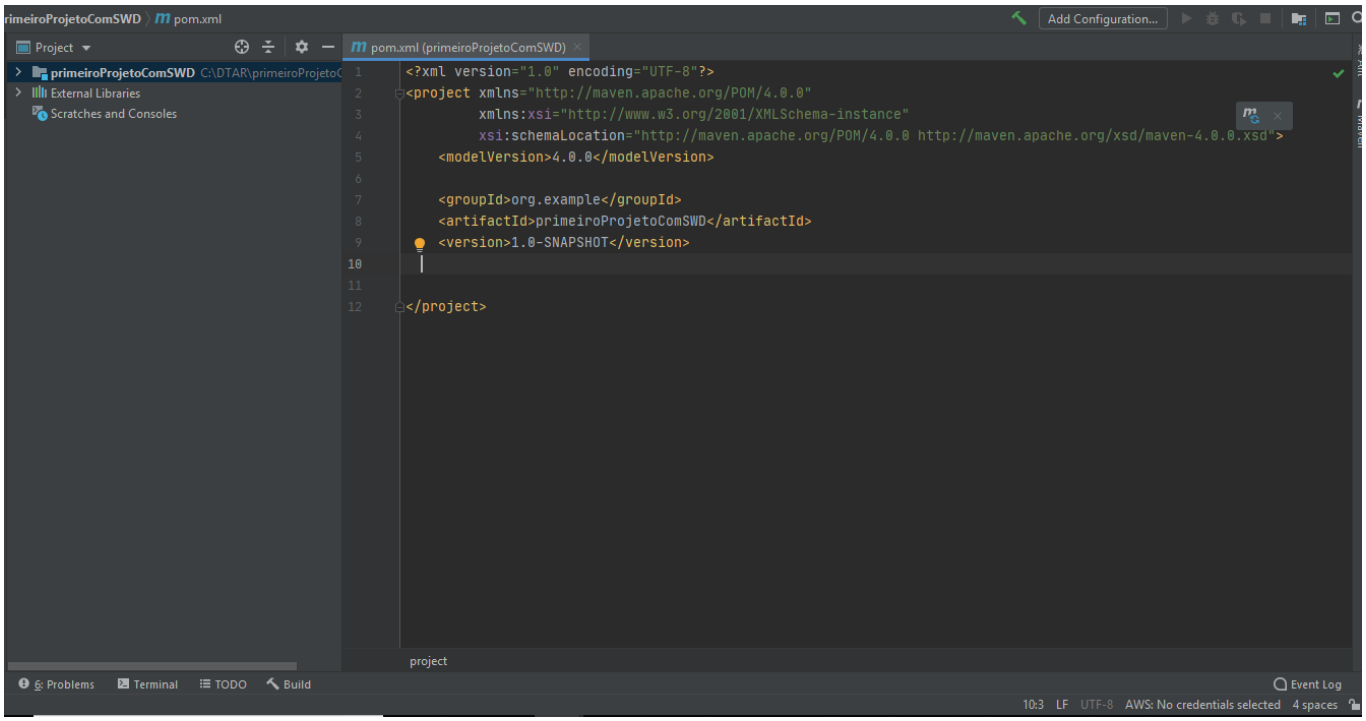
Criando um projeto na sua IDE de escolha

File > New > Project

Na tela que abrir você vai em Maven, lá em Project SDK precisa já estar selecionado a SDK, se não tiver você vai lá e seleciona. Caso não apareça você vai precisar instalar. Aperta em Next, em Name você põe o nome do seu projeto, em Location você localiza a pasta onde você quer que o projeto seja criado, aqui eu recomendo em C:\...Nome do seu projeto e aperte Finish.

Feito isso seu primeiro projeto será criado, provavelmente você já irá cair na tela do "pom.xml", esse arquivo ajuda

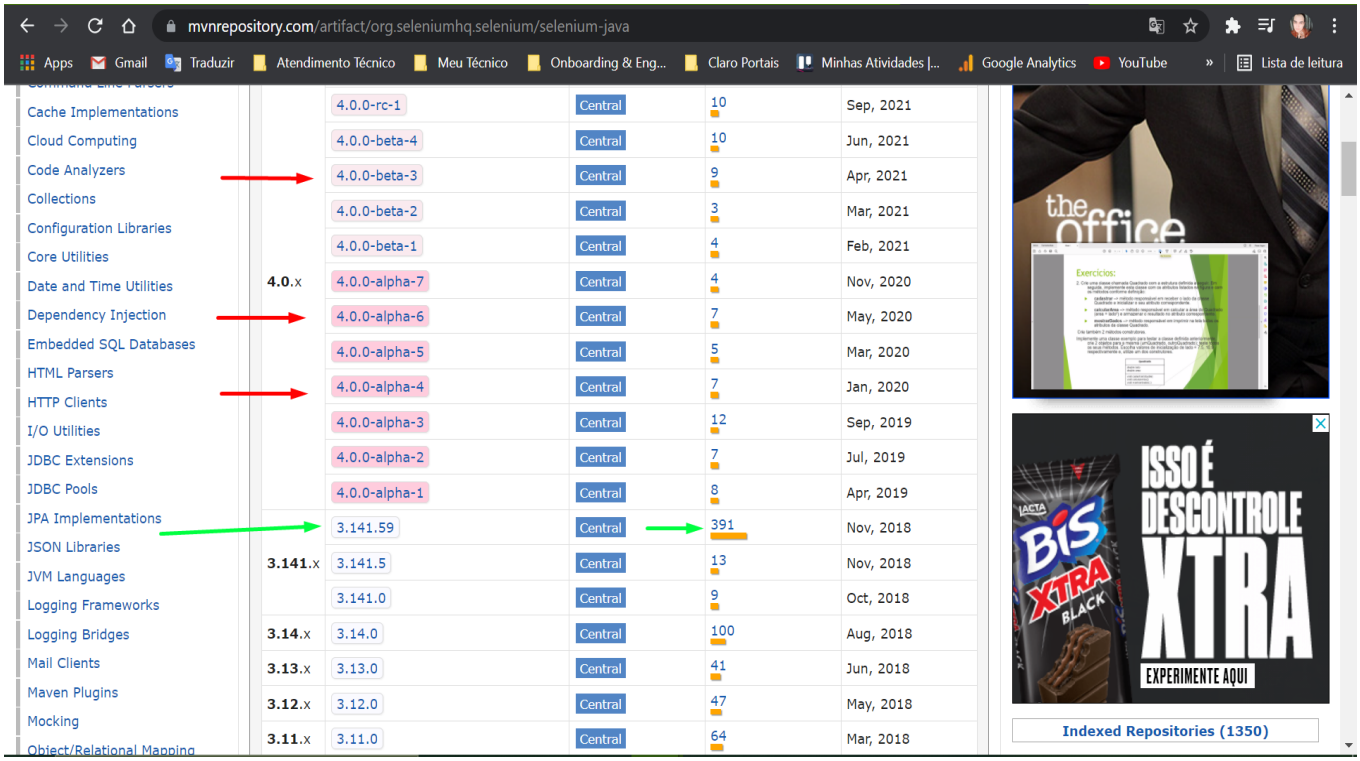
você a configurar e importar o que você vai precisar nos testes.



Dentro do pom.xml você precisará incluir as dependências do Junit e selenium-java que se encontram no link abaixo, é só procurar pelo nome da extensão que quer usar Ex.JUnit ou Selenium.

https://mvnrepository.com/

Obs. Versões alpha e beta são versões mais instáveis então procure sempre versões mais estáveis e que tenham mais pessoas usando como no exemplo abaixo:

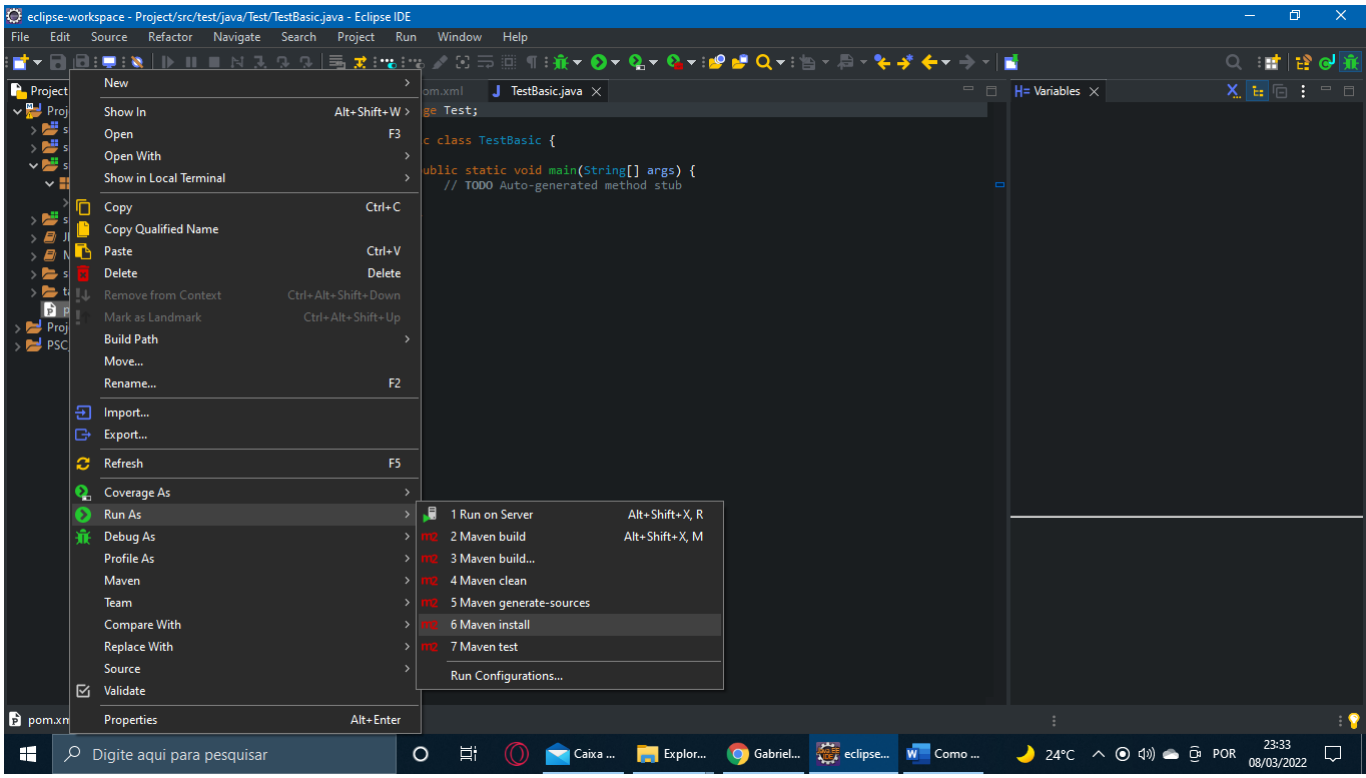


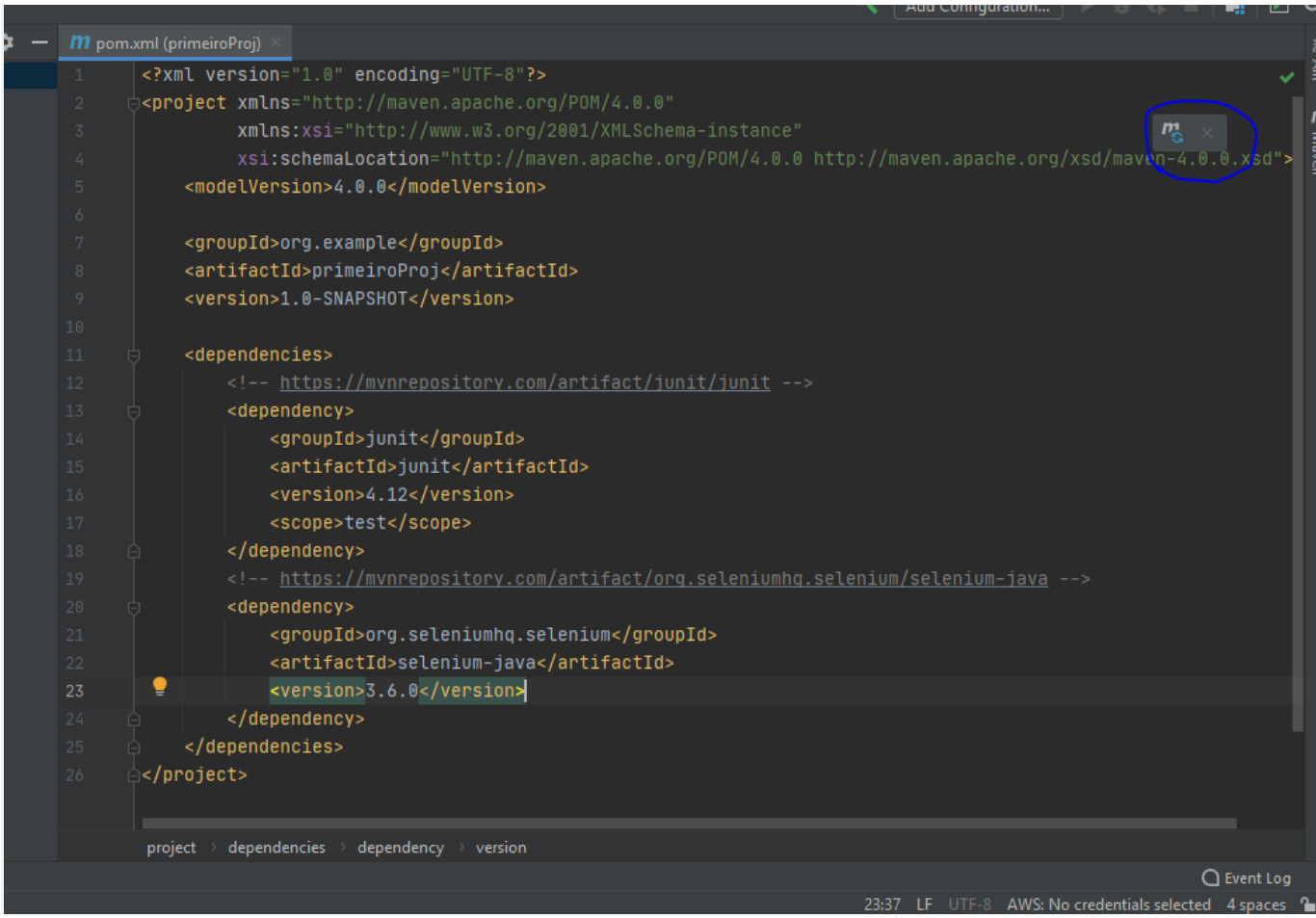
Para facilitar, irei pôr o código aqui já, mas caso tenha atualização de versões é bom vocês buscarem no link acima citado as versões mais atualizadas.

```
<dependencies>
    <!-- https://mvnrepository.com/artifact/junit/junit -->
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>3.6.0</version>
    </dependency>
</dependencies>
```

Feito isso vai aparecer na tela um botão do maven que é o Load Maven Changes, você precisa rodar esse load par a ele carregar as dependências.

Ou caso esteja no Eclipse deverá ir em seu projeto, clicar com o botão direto no arquivo pom.xml já com as dependências setadas nele e mandar executar um maven install como no print abaixo


eclipse-workspace - Project/src/test/java/Test/TestBasic.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
New
Show In Alt+Shift+W
Open F3
Open With
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Mark as Landmark Ctrl+Alt+Shift+Up
Build Path
Move...
Rename... F2
Import...
Export...
Refresh F5
Coverage As
Run As
Debug As
Profile As
Maven
Team
Compare With
Replace With
Source
Validate
Properties Alt+Enter
1 Run on Server Alt+Shift+X, R
2 Maven build Alt+Shift+X, M
3 Maven build...
4 Maven clean
5 Maven generate-sources
6 Maven install
7 Maven test
Run Configurations...
TestBasic.java
Variables
Digite aqui para pesquisar
24°C
POR
23:33
08/03/2022


pom.xml (primeiroProj)
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<groupId>org.example</groupId>
<artifactId>primeiroProj</artifactId>
<version>1.0-SNAPSHOT</version>
<dependencies>
<!-- https://mvnrepository.com/artifact/junit/junit -->
<dependency>
<groupId>junit</groupId>
<artifactId>junit</artifactId>
<version>4.12</version>
<scope>test</scope>
</dependency>
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
<dependency>
<groupId>org.seleniumhq.selenium</groupId>
<artifactId>selenium-java</artifactId>
<version>3.6.0</version>
</dependency>
</dependencies>
</project>
project dependencies dependency version
Event Log
23:37 LF UTF-8 AWS: No credentials selected 4 spaces

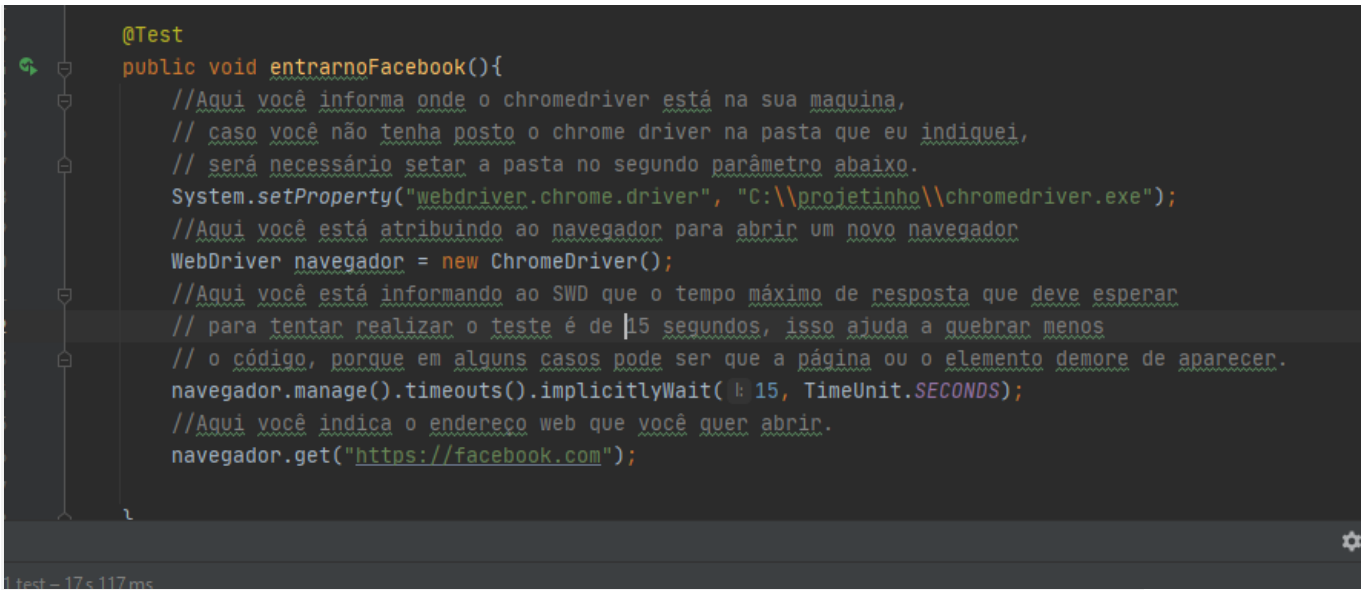
Carregando as alterações do Maven

Pronto, o ambiente já está configurado.

Abrindo o navegador pela primeira vez

Para criar a primeira classe de testes você vai em “src > test > java” crie um package chamado testes e dentro dele uma nova classe. ATENÇÃO, ESSA NOVA CLASSE DEVE TERMINAR COM O TEXTO “Test”, exemplo “primeiroTesteTest”. Isso vale também para qualquer outra classe que você vá rodar os testes, por exemplo “DashboardTest”, “LoginTest” etc. Você também deve por @Test acima do método de teste que você criou.

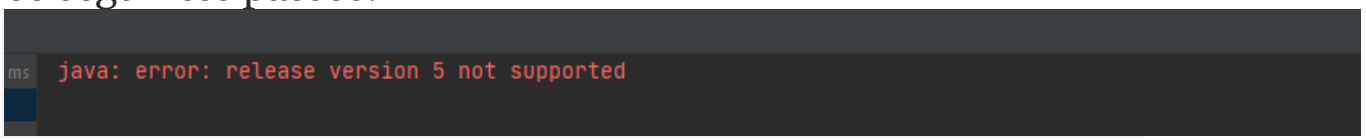
Agora dentro da classe você vai criar o método de teste que aqui eu nomeei como entrar no Facebok, mas você pode criar como quiser. Vou pôr o código aqui comentado explicando cada linha. Sugiro que você remova os comentários!



Abrindo o navegador

Caso você aperte no Run, aquele botão verde que fica ao lado do código ele já irá executar a o teste abrindo o navegador.

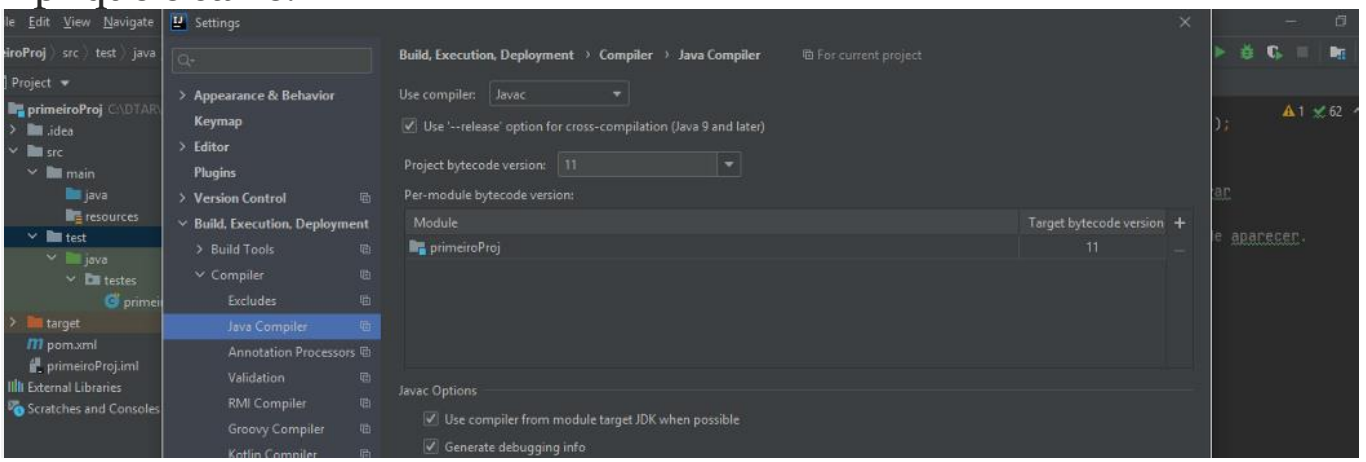
Se você receber esse erro quando tente executar o teste siga os seguintes passos:



Erro da versão

File > Setting > Build, Execution, Deployment > Compiler > Java Compiler

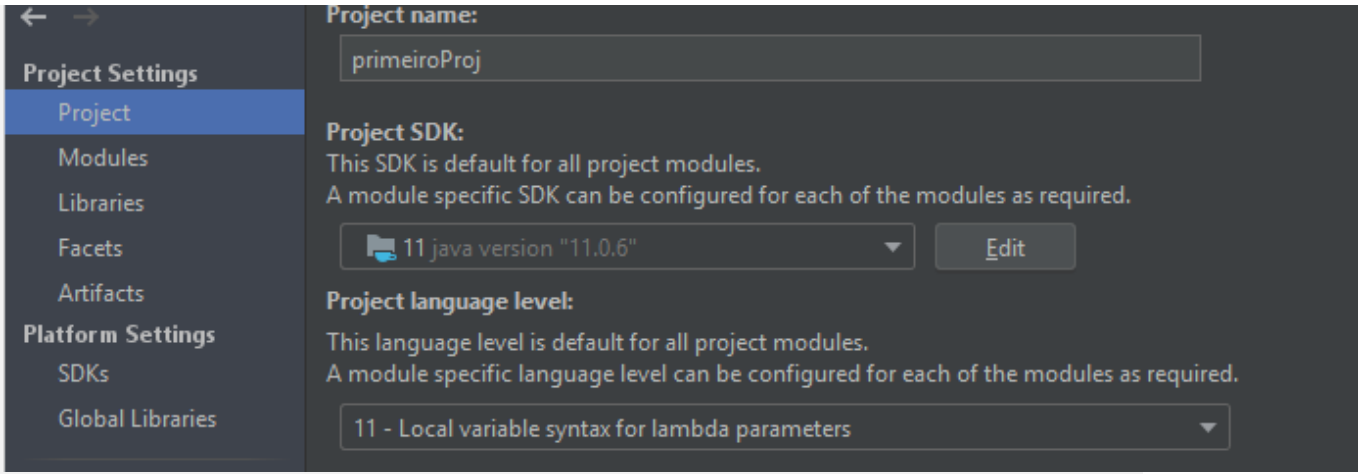
Em “Project bytecode version” selecione o “11”. Mais abaixo em “Target bytecode version” selecione “11” também. Aplique e salve.



Corrigindo o erro1

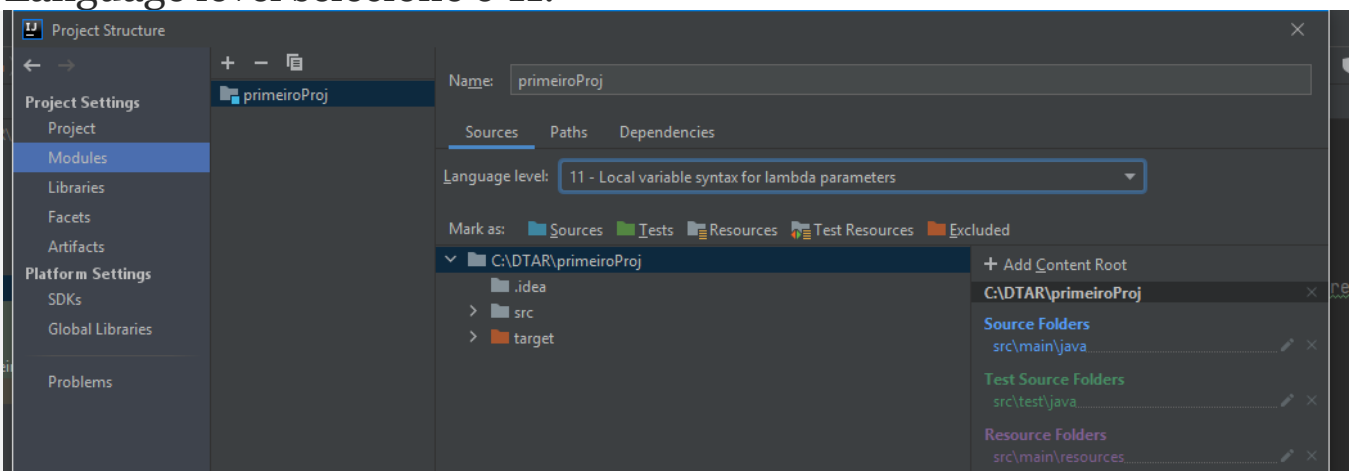
Agora em File > Project Structure > Project

Em Project SDK selecione 11.



Corrigindo o erro 2

Saindo de Project passe para a aba baixo “Modules” em Language level selecione o 11.



Corrigindo o erro 3

Pronto, é só rodar novamente no Run que agora irá executar.

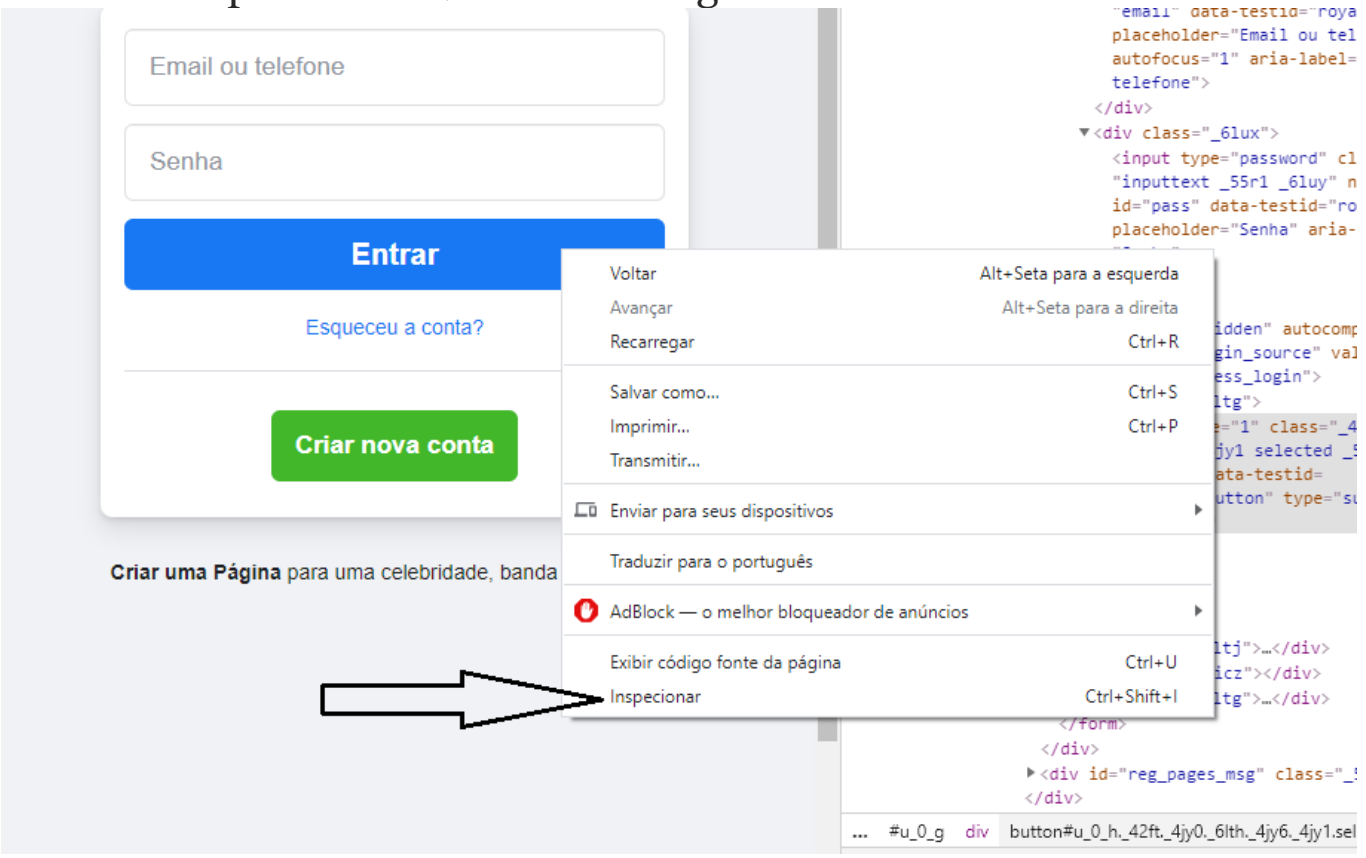
Localizando elementos

Existem 8 formas de encontrar elementos no Selenium WebDriver. Pode ser pelo nome da classe, css selector, id, name, link text, tag name e xpath. Aqui eu vou utilizar pelo id, xpath, link text e nome da classe. Caso queira saber mais

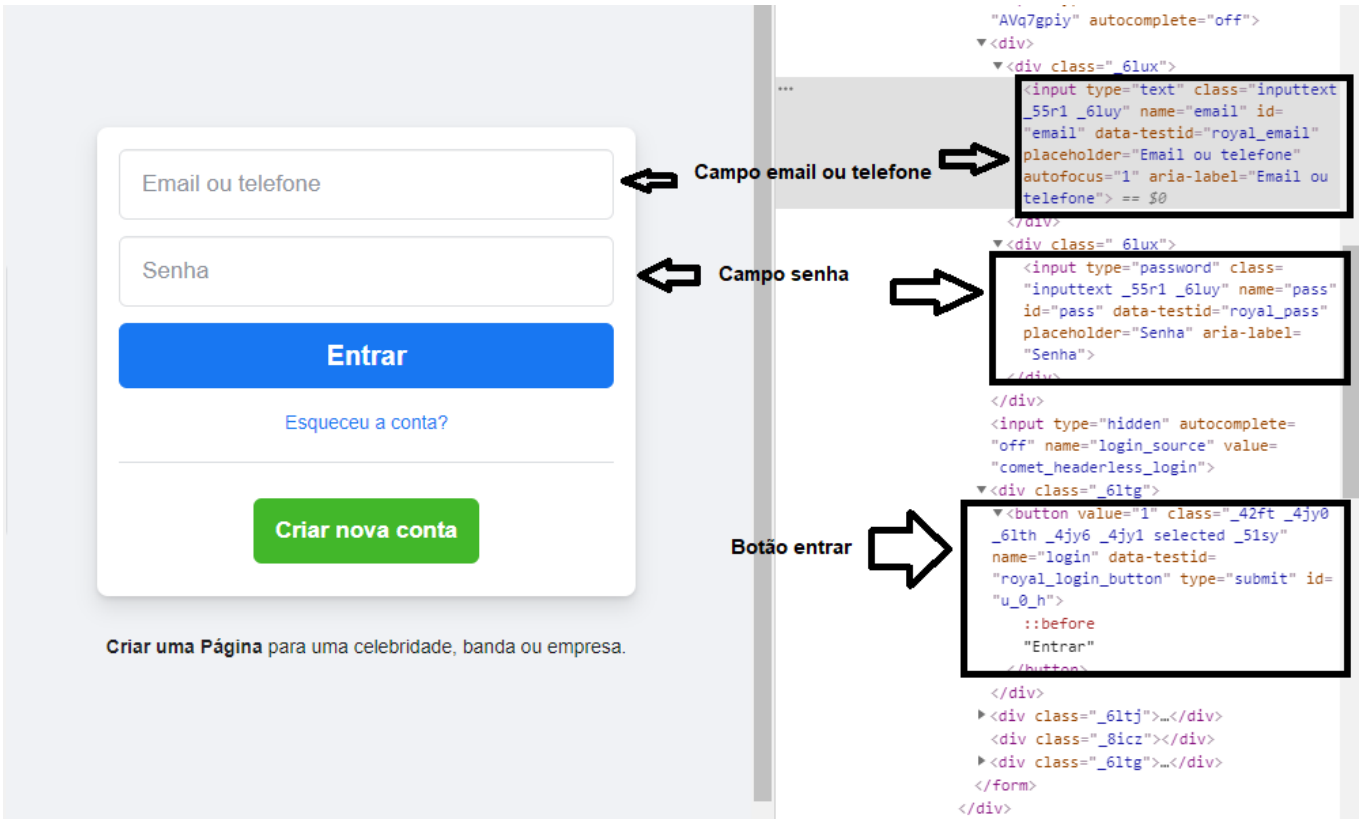
sobre a localização de elementos

acesse https://www.selenium.dev/documentation/legacy/selenium_ide/#locating-elements

Para achar os elementos na sua página você deve clicar com o botão direito em cima do elemento que deseja capturar e clica em Inspecionar. Observe as imagens:



Inspecionando os elementos



Inspecionando os elementos

Aqui estão alguns dos principais métodos que você irá utilizar para localizar os elementos e realizar ações durante a escrita dos scripts:

findElement() — Você indica pelo que deve ser buscado, dentro dos () você informa os parâmetros que podem ser algum dos 8 que informei acima.

click() — Você informa para ser realizada ação de click

sendKeys() — Você informa que deve ser enviado um texto

getText() — Você informa que deve ser capturado o texto

assertEquals() — Você faz uma comparação de igualdade entre os parâmetros informados.

Escrevendo os primeiros comandos

```
//Encontrando o campo email/telefone pelo id e preenchendo
navegador.findElement(By.id("email")).sendKeys( ...charSequences: "marcinho_boladão@test.com");
//Pelo name
navegador.findElement(By.name("email")).sendKeys( ...charSequences: "marcinho_boladão@test.com");
//Pelo xpath
navegador.findElement(By.xpath("//input[@type='text']")).sendKeys( ...charSequences: "marcinho_boladão@test.com");

//Encontrando o campo senha e preenchendo pelo id
navegador.findElement(By.id("pass")).sendKeys( ...charSequences: "123456");
//Pelo name
navegador.findElement(By.name("pass")).sendKeys( ...charSequences: "123456");
//Pelo xpath
navegador.findElement(By.xpath("//input[@type='password']")).sendKeys( ...charSequences: "123456");
```

Preenchendo campo login e senha

No seu caso você só irá utilizar uma forma de encontrar o elemento, eu coloquei três para ficar um pouco mais claro. Não é em todos os casos que o elemento vai ter um Id ou um Name para encontrar, nesses casos você pode utilizar o Xpath que parece ser um pouco mais difícil, mas ajuda bastante nessas situações. Como se pode observar dentro dos parâmetros você indica ali o Id, Name ou o caminho do Xpath. Dentro do sendKeys() você indica o texto que você quer inserir ou, você pode por uma variável que já contenha o texto, aí você é quem decide.

Agora vamos encontrar o botão Entrar

```
//Encontrando o botão Entrar pelo name
navegador.findElement(By.name("login")).click();
//Pelo id
navegador.findElement(By.id("u_0_h")).click();
//Pelo xpath
navegador.findElement(By.xpath("//button[@type='submit']")).click();
//Pelo texto que ele possui
navegador.findElement(By.linkText("Entrar")).click();
```

Clicando no botão entrar

Como se pode observar neste caso escrevi 4 formas diferentes de encontrar o botão Entrar. Escolha a que melhor se aplica ao seu projeto.

Para finalizar vamos utilizar o getText() e o assertEquals(). Aqui eu vou comparar um texto com a captura do erro quando você não insere nenhuma informação para login.

```
navegador.findElement(By.linkText("Entrar")).click();
navegador.findElement(By.name("login")).click();


String erro = navegador.findElement(By.xpath("//div[@role='alert']")).getText();
assertEquals( expected: "O email ou o número de telefone inserido não " +
        "corresponde a nenhuma conta. Cadastre-se para abrir uma conta.", erro);
```

Capturando erro de campos obrigatórios

Repare que, depois de tentar entrar com um login e senha errado, eu informei para clicar novamente em entrar, porque nesse caso a página de login e senha estarão em branco na segunda tentativa, assim surgir o erro.

Extra — Executando o teste sem abrir o Navegador

Algo que será muito utilizado mais a frente é executar o código sem abrir o navegador, isso será necessário para implementar o código em uma possível integração contínua caso queira. É muito simples, mas deu um trabalho encontrar, então vamos lá.

```
ChromeOptions options = new ChromeOptions();
options.setHeadless(true);
System.setProperty("webdriver.chrome.driver", "C:\\projetinho\\chromedriver.exe");
WebDriver navegador = new ChromeDriver();
navegador.manage().timeouts().implicitlyWait( 15, TimeUnit.SECONDS);
navegador.get("https://facebook.com");
```

Executando o projeto em Headless

Pronto, inserindo essas duas primeiras linhas de código, o navegador não irá mais abrir, sendo assim, o teste será executado em modo Headless. Caso você queira voltar para o navegador aparecer, mude de true para false.

Para aprender mais sobre o Selenium WebDriver recomendo que vocês acessem a documentação e também tem um curso ótimo do Júlio de Lima, que além de aprender a criar os scripts de automação ele ensina como organizar seu código com boas práticas, afinal, mais importante que escrever o código, é escrevê-lo bem!

Link da documentação.
https://www.selenium.dev/documentation/

Curso de Júlio de lima:

https://www.youtube.com/watch?v=lELENTQ-PWc