

# Automação de Testes com Selenium WebDriver e Java

**Camila Cavalcante**

Tech Education Coordinator  
DIO

 [linkedin.com/in/cami-la](https://www.linkedin.com/in/cami-la)

 [github.com/cami-la](https://github.com/cami-la)

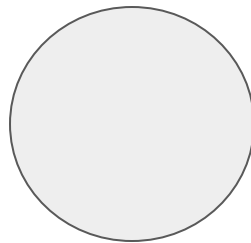
 [instagram.com/camimi\\_la](https://www.instagram.com/camimi_la)

# Objetivo Geral

Conheça o Selenium WebDriver, a principal ferramenta de automação de páginas Web. Nesse contexto, explore a linguagem de programação Java e entenda como o Selenium automatiza as ações diretamente em seu browser.

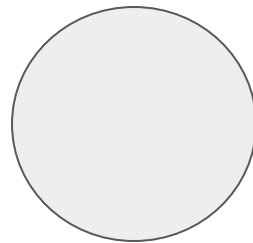


# Pré-Requisitos



- JDK 8+
- IDE para desenvolvimento Java (Usarei Eclipse IDE)
- Sintaxe básica Java
- Princípios da Web (HTML/CSS e JS)
- Noção acerca de testes unitários

# Percurso



## Parte 1

Visão geral: Testes de unidade e E2E

## Parte 2

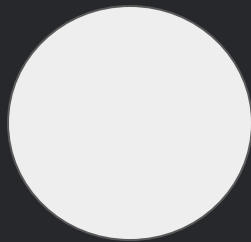
Overview: Testes unitários com JUnit 5

## Parte 3

Overview: Testes E2E com Selenium WebDriver

## Parte 4

Para Saber Mais

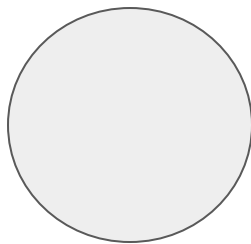


## Parte 1

# Visão geral: Testes de unidade e Aceitação



// Automação de Testes com Selenium WebDriver e Java



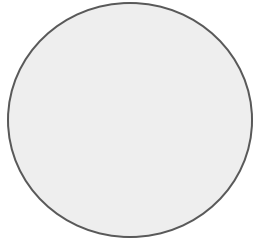
# Por que testamos?

- O software deve fazer o que o cliente precisa de maneira confiável, segura, eficiente e flexível.
- Para um software ser testado corretamente, esse processo deve ser automatizado com o auxílio de ferramentas com esta finalidade.
- Os testes automatizados vêm como uma forma de poupar tempo de detecção de erros e de aumento de confiabilidade com relação aos testes em si.

# Pirâmide de testes

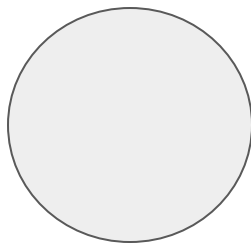


# Testes Unitários



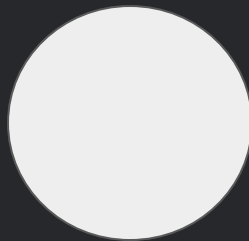
- São testes que verificam se uma parte específica do código, costumeiramente a nível de função, está funcionando corretamente.
- Estes tipos de testes são frequentemente escritos por desenvolvedores quando trabalham no código, para assegurar que a função específica está executando como esperado.
- Testes Unitários não dependem de nenhum processo ou sistema externo (banco de dados, console, rede e etc.)





# Testes de Aceitação

- "Teste de aceitação", categoria que também recebe nomes como "Teste de UI" ou "E2E".
- O teste de aceitação é a ação de teste final antes da implementação do software.
- A meta deste teste é verificar se o software está pronto e pode ser utilizado pelos usuários, para desempenhar as funções e tarefas para as quais o software foi construído.



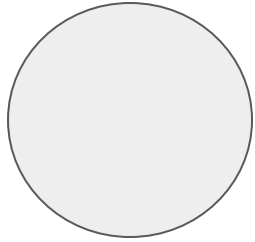
Parte 2



# Testes unitários com JUnit 5

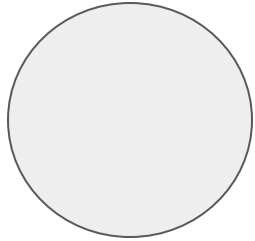
// Automação de Testes com Selenium WebDriver e Java

# JUnit



- Para efetuar os testes automatizados precisamos de um framework auxiliar de testes unitários e assim efetuar as operações de lógica de negócio na camada do servidor.
- Existem dois principais frameworks que podemos usar que se comunicam muito bem com o Selenium WebDriver: o JUnit e o TestNG.
- O JUnit é um framework open-source que possibilita a criação das classes de testes e tem como objetivo facilitar a criação de casos de teste, além de permitir escrever testes que retenham seu valor ao longo do tempo.

# Writing Tests



```
MyFirstJUnitJupiterTests.java Calculator.java
1 package com.company;
2
3 import org.junit.jupiter.api.Assertions;
4 import org.junit.jupiter.api.Test;
5
6 class MyFirstJUnitJupiterTests {
7     //Classe que será testada
8     private final Calculator calculator = new Calculator();
9
10    @Test //Anotação para indicar que este é um método de teste
11    void addition() {
12        //Given
13        int number1 = 1;
14        int number2 = 1;
15        //When
16        int actual = calculator.add(number1, number2);
17        //Then
18        int expected = 2;
19        Assertions.assertEquals(expected, actual);
20    }
21 }
22
```

JUnit

Finished after 0,19 seconds

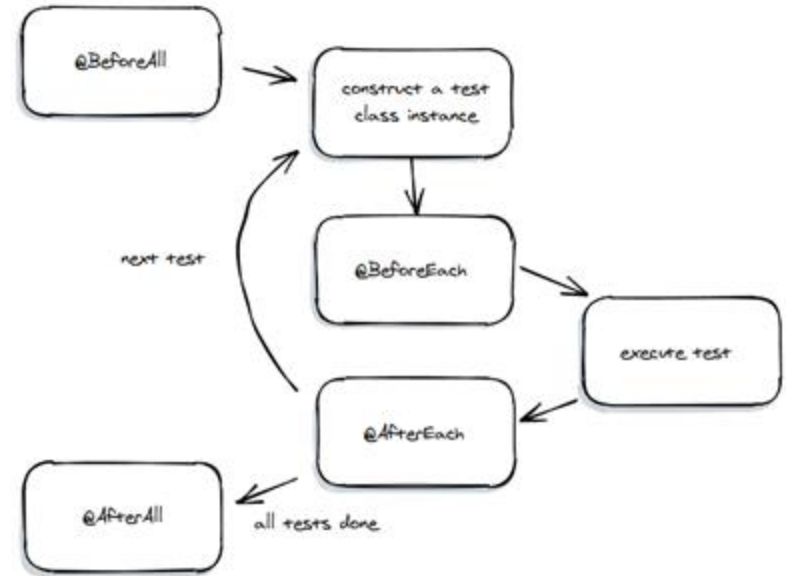
Runs: 1/1   Errors: 0   Failures: 0

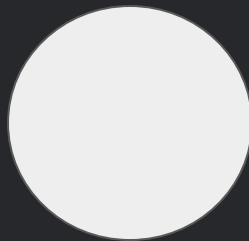
MyFirstJUnitJupiterTests [Runner: JUnit 5] (0,035 s)

Failure Trace

# JUnit 5 Lifecycle

```
1 LoginTest.java
2
3 import org.junit.jupiter.api.Assertions;
4 import org.junit.jupiter.api.BeforeEach;
5 import org.junit.jupiter.api.Test;
6
7
8 import dev.camila.curso.selenium.webdriver.pages.LoginPage;
9
10 class LoginTest {
11     private LoginPage loginPage;
12
13     @BeforeEach
14     void setUp() throws Exception {
15         loginPage = new LoginPage();
16         loginPage.visit("http://automationpractice.com/index.php?controller=aut
17     }
18
19     @AfterEach
20     void tearDown() throws Exception {
21         this.loginPage.quitWebDriver();
22     }
23
24     @Test
25     void test() {
26         //when
27         loginPage.signin();
28         //then
29         String expected = "MY ACCOUNT";
30         String actual = loginPage.getMyAccountMessage();
31         Assertions.assertEquals(expected, actual);
32     }
33 }
```





## Parte 3

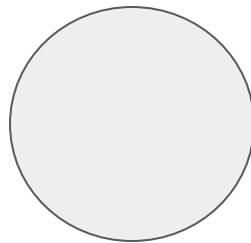
# Testes E2E com



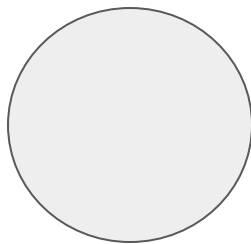
# Selenium WebDriver

// Automação de Testes com Selenium WebDriver e Java

# Selenium



- Selenium é um conjunto de ferramentas de código aberto multiplataforma, usado para testar aplicações web pelo browser de forma automatizada.
- Ele executa testes de funcionalidades da aplicação web e testes de compatibilidade entre browser e plataformas diferentes.
- O Selenium suporta diversas linguagens de programação, como por exemplo C#, Java e Python, e vários navegadores web como o Chrome e o Firefox.



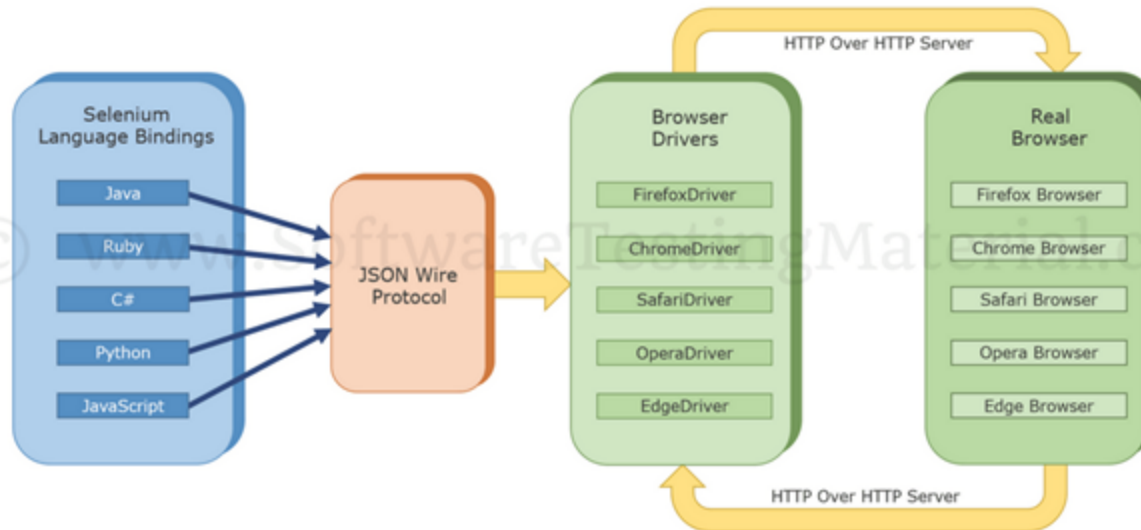
# Selenium WebDriver

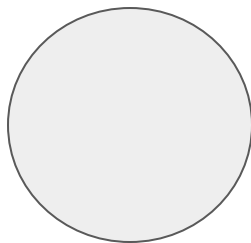
- O ecossistema do Selenium é bem completa, tendo: Selenium IDE, Selenium WebDriver e Selenium Grid.
- O Selenium WebDriver usa o próprio driver do navegador para a automação.
- É a forma mais moderna de interação atualmente, pois cada browser possui o seu respectivo driver, permitindo a interação entre o script de teste e o respectivo browser.



# Selenium WebDriver

## Selenium WebDriver Architecture

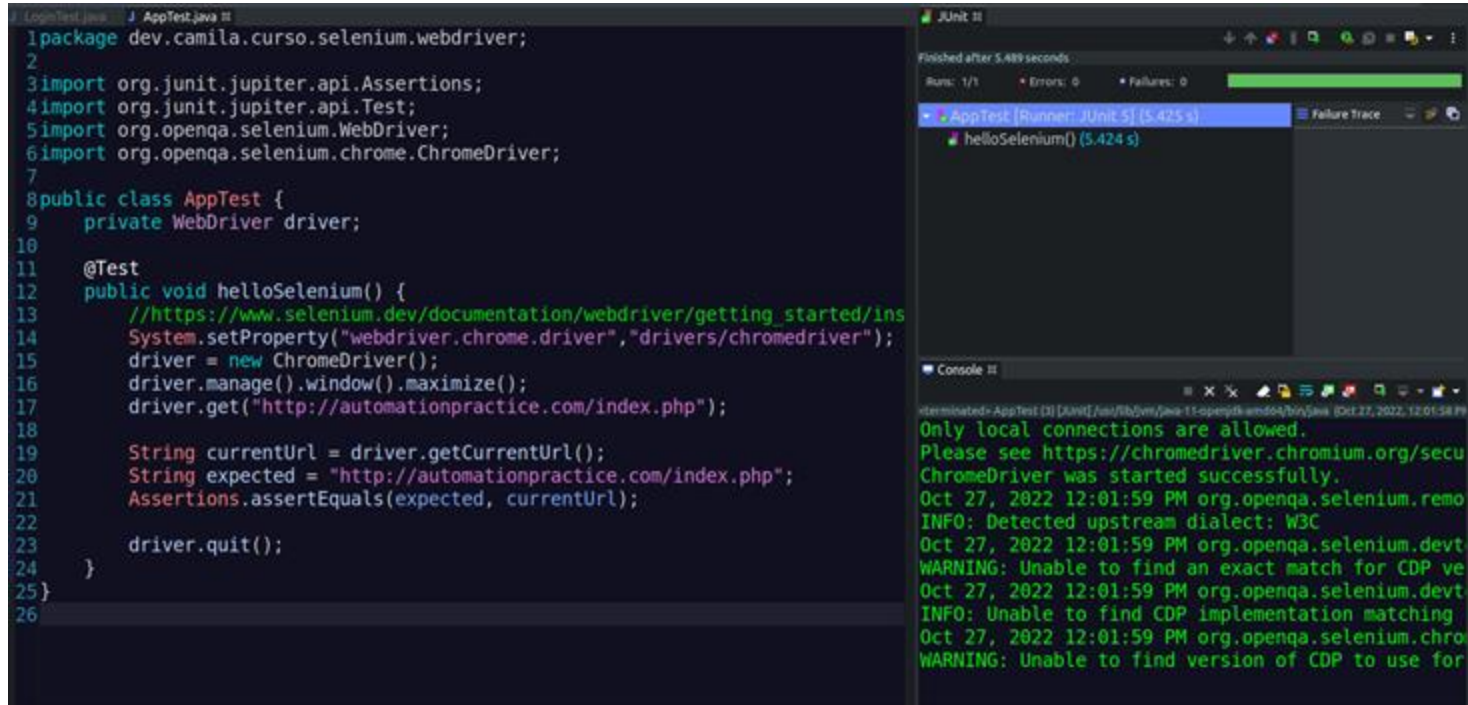




# Selenium WebDriver

- A Selenium Client Library consiste em linguagens como Java, Ruby, Python, C# e etc. Após os casos de teste acionados, o código do Selenium será convertido para o formato Json.
- O Json gerado é disponibilizado para os drivers do navegador por meio do protocolo http.
- Cada navegador tem um driver de navegador específico. Assim que o driver do navegador recebe instruções, ele as executa no navegador. Em seguida, a resposta é dada de volta na forma de resposta HTTP.

# Writing Tests



```
1package dev.camila.curso.selenium.webdriver;
2
3import org.junit.jupiter.api.Assertions;
4import org.junit.jupiter.api.Test;
5import org.openqa.selenium.WebDriver;
6import org.openqa.selenium.chrome.ChromeDriver;
7
8public class AppTest {
9    private WebDriver driver;
10
11    @Test
12    public void helloSelenium() {
13        //https://www.selenium.dev/documentation/webdriver/getting_started/ins
14        System.setProperty("webdriver.chrome.driver", "drivers/chromedriver");
15        driver = new ChromeDriver();
16        driver.manage().window().maximize();
17        driver.get("http://automationpractice.com/index.php");
18
19        String currentUrl = driver.getCurrentUrl();
20        String expected = "http://automationpractice.com/index.php";
21        Assertions.assertEquals(expected, currentUrl);
22
23        driver.quit();
24    }
25}
26
```

JUnit 11

Finished after 5.489 seconds

Runs: 1/1 • Errors: 0 • Failures: 0

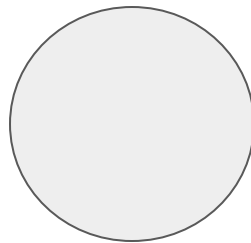
AppTest [Runner: JUnit 5] [5.425 s]

helloSelenium() (5.424 s)

Console 11

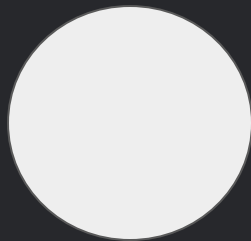
```
<terminated> AppTest (3) [JUnit] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Oct 27, 2022, 12:01:58 PM)
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for details.
ChromeDriver was started successfully.
Oct 27, 2022 12:01:59 PM org.openqa.selenium.remote.ProtocolHandshake$1 lambda$run$0
INFO: Detected upstream dialect: W3C
Oct 27, 2022 12:01:59 PM org.openqa.selenium.remote.ProtocolHandshake$1 lambda$run$0
WARNING: Unable to find an exact match for CDP version
Oct 27, 2022 12:01:59 PM org.openqa.selenium.remote.ProtocolHandshake$1 lambda$run$0
INFO: Unable to find CDP implementation matching
Oct 27, 2022 12:01:59 PM org.openqa.selenium.remote.ProtocolHandshake$1 lambda$run$0
WARNING: Unable to find version of CDP to use for
```

# Para Saber Mais



- [Dominando IDEs Java](#)
- [Gerenciamento de Dependências e Build em Java com Maven](#)
- [Introdução ao Junit 5](#)
- [Resumo: Automatizando Testes com Selenium WebDriver](#)
- [Código-Fonte](#)



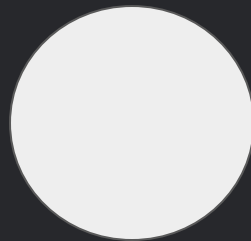


Hands On!

***“Falar é fácil.  
Mostre-me o código!”***

**Linus Torvalds**





# Dúvidas?

- > Comunidade Online (Rooms)
- > Fórum do Bootcamp e/ou Artigos
- > Central de Ajuda DIO

