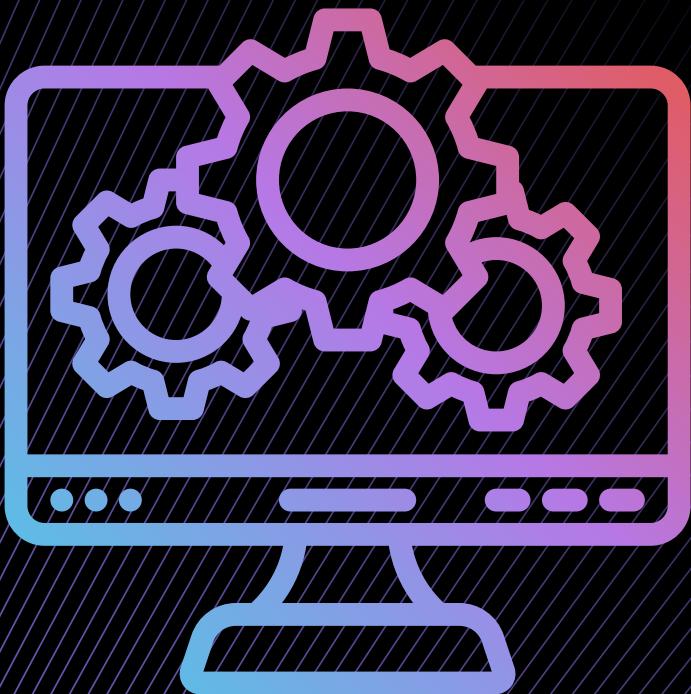


The background features a dark navy blue gradient. A large, semi-transparent, irregular shape composed of numerous small, light blue dots is positioned in the upper right quadrant, extending towards the center. This dot pattern has a slightly darker, more solid appearance at its edges.

Selenide

O QUE É?

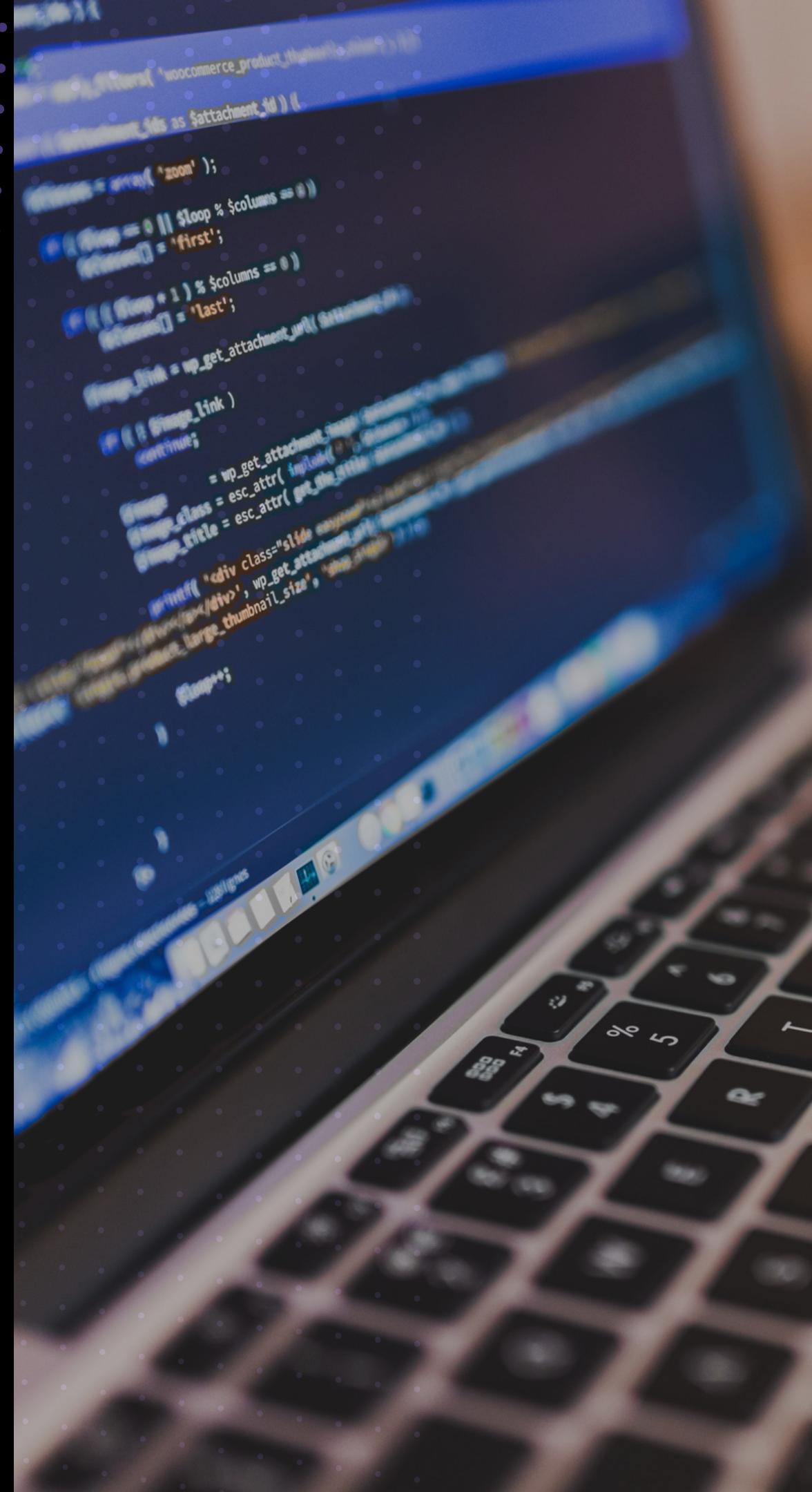


- Selenide é um framework para automação de testes.
- É uma aplicação baseada no Selenium WebDriver para o desenvolvimento de testes funcionais.
- Oferecer uma ótima sintaxe para a escrita em Java.

Por que foi desenvolvida?

Uma equipe de desenvolvimento web chamada de Codeborne percebeu que ao usar o Selenium WebDriver teriam que escrever o mesmo código nos testes, tinham que inicializar e fechar o navegador a cada teste, duplicações desnecessárias.

Assim, a equipe Codeborne desenvolveu uma API para retirar o código repetitivo, dando origem ao Selenide.



VANTAGENS!



- Selenide torna seus testes estáveis, resolvendo (quase) todos os problemas de tempo de renderização.
- Espera inteligente
- Suporte Ajax
 - Selenide tem métodos embutidos para espera dos elementos mudarem seu estado.

VANTAGENS!



- WebDriver transparente
 - O Selenide iniciará e desligará o navegador automaticamente.
- Métodos de conveniência
 - API concisa para que seus testes sejam mais curtos e mais legíveis.
- Capturas de tela automatizadas
 - Quando seu teste falhar, o Selenide fará a captura de tela automaticamente.

Instalação

```
<name>testweb</name>
<!-- FIXME change it to the project's website --&gt;
&lt;url&gt;<a href="http://www.example.com">http://www.example.com</url>

<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
</properties>

<dependencies>
    <!-- https://mvnrepository.com/artifact/org.testng/testng -->
    <dependency>
        <groupId>org.testng</groupId>
        <artifactId>testng</artifactId>
        <version>7.6.0</version>
        <scope>test</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/com.codeborne/selenide -->
    <dependency>
        <groupId>com.codeborne</groupId>
        <artifactId>selenide</artifactId>
        <version>6.5.1</version>
    </dependency>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.11</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Como funciona a seleção de elementos (HTML)

Selenide fornece comandos simples, veja alguns deles.

- **\$(*String seletorCss*)**: Seleciona um elemento (HTML) pelo seu seletor marcado, por exemplo, o id. Retorna um objeto do tipo SelenideElement com referência a um elemento desejado.
- **\$(*By*)**: Semelhante ao comando anterior, com a diferença no modo em que passamos a instrução para escolher o elemento (HTML) utilizando um método static chamado *By*.
- **\$\$(*String seletorCss*)**: Seleciona um elemento (HTML) pelo seu seletor CSS, por exemplo, o id. Retorna um objeto do tipo ElementsCollection com referência a vários elementos (HTML).
- **\$\$(*By*)**: Similar ao comando anterior, mas a diferença está apenas no modo em que passamos a instrução para escolher o elemento utilizando um método static chamado *By*.

Como funciona a seleção de elementos (HTML)

A tambem métodos "**static**".

- **byText**: Buscar elemento (html) por texto;
- **withText**: Buscar elemento com determinado texto especificado;
- **by**: Buscar elemento pelo atributo;
- **byTitle**: Buscar elemento pelo atributo title;
- **byValue**: Buscar elemento pelo atributo value.

Arquivo "index.html"

```
<html>
    <head>
        <title>Seletores</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <div id="texto">Obter elemento com id: texto</div>
        <a title="Meu titulo">Obter elemento com titulo: Meu titulo</a>
        <form id="idForm">
            <input name="nomeInput" type="text"
                   value="Obter elemento pelo value"/>
        </form>
    </body>
</html>
```

```

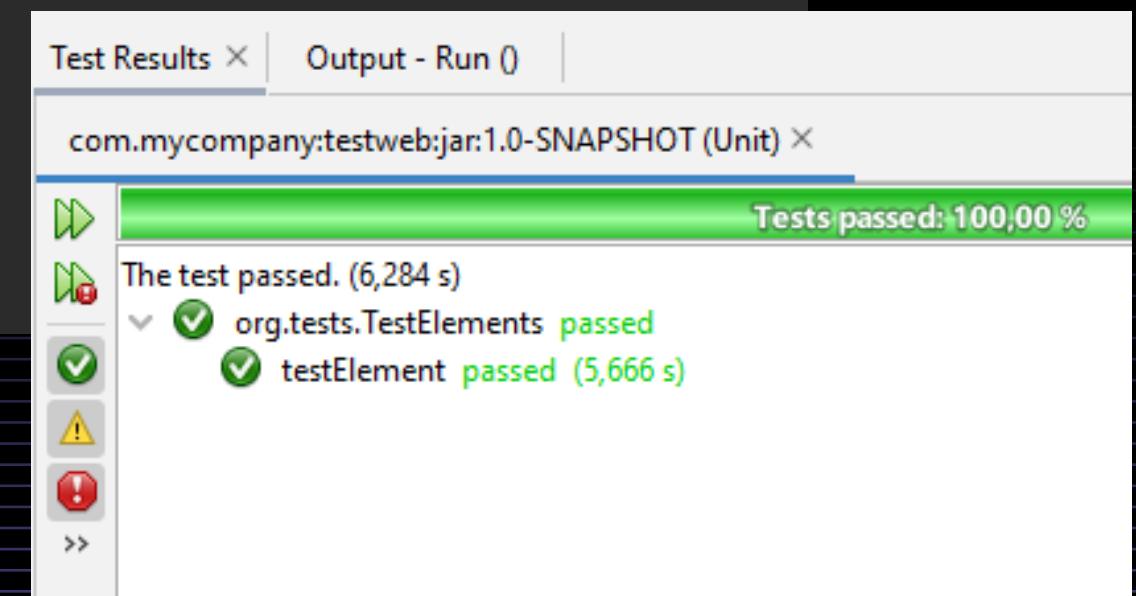
@Test
public void testElement() {
    open("http://localhost/sites/index.html");
    assertEquals("<div id=\"texto\">Obter elemento com id: texto</div>",
                $(By.id("texto")).toString());
    assertEquals("<div id=\"texto\">Obter elemento com id: texto</div>",
                $("#texto").toString());

    assertEquals("<input name=\"nomeInput\" type=\"text\" value=\"Obter elemento pelo value\"></input>",
                $(By.name("nomeInput")).toString());

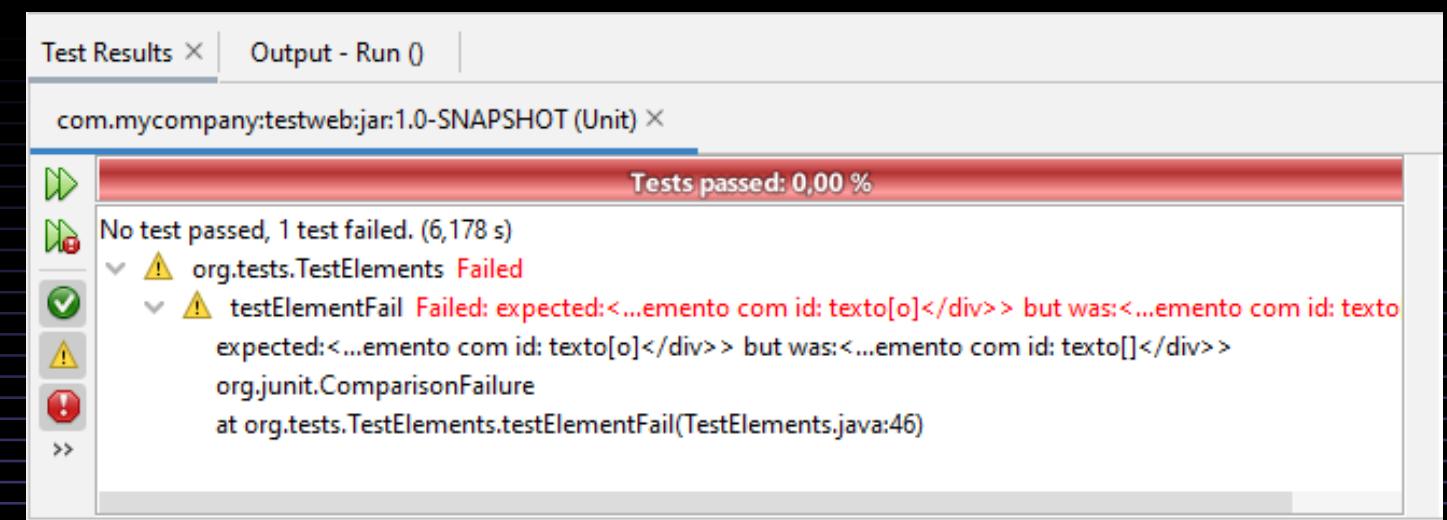
    assertEquals("<a title=\"Meu titulo\">Obter elemento com titulo: Meu titulo</a>",
                $(by("title", "Meu titulo")).toString());
    assertEquals("<a title=\"Meu titulo\">Obter elemento com titulo: Meu titulo</a>",
                $(byTitle("Meu titulo")).toString());

    assertEquals("<form id=\"idForm\"></form>",
                $(By.id("idForm")).toString());
    assertEquals("<form id=\"idForm\"></form>",
                $(by("id", "idForm")).toString());
}

```



```
@Test  
public void testElementFail(){  
    open("http://localhost/sites/index.html");  
    assertEquals("<div id=\"texto\">Obter elemento com id: texto</div>",  
               $(By.id("texto")).toString());  
  
    assertEquals("<a title=\"Meu titulo\">Obter elemento com titulo: Meu titulo</a>",  
               $(by("title", "Meu tituloo")).toString());  
  
    assertEquals("<form id=\"idForm\"></form>",  
               $(By.id("idFormmm")).toString());  
}
```



SelenideElement

API fornece métodos adicionais facilitando o trabalho, alguns deles.

- ***should(Condição)***: Verifica se um determinado elemento atende a todas as condições dadas;
- ***shouldNot(Condição)***: Verifica se um determinado elemento não atende a todas as condições dadas;
- ***setValue(String)***: Define um valor no atributo value de algum elemento (HTML);
- ***val()***: Obtém o valor do atributo **value** de algum elemento (HTML);
- ***pressEnter()***: Pressiona o enter;
- ***isDisplayed()***: Retorna um boolean da verificação se algum elemento é visível;

SelenideElement

API fornece métodos adicionais facilitando o trabalho, alguns deles.

- **`exists()`**: Retorna um boolean da verificação se algum elemento existe;
- **`selectOptionByValue(String value)`**: Seleciona a opção pelo valor do elemento;
- **`getSelectedOption()`**: Retorna a opção selecionada do elemento;
- **`getSelectedValue()`**: Retorna o valor da opção.

Arquivo "element.html"

```
<html>
<head>
<meta charset="UTF-8">
<title>SelenideElement</title>
</head>
<body>
    <form id="formulario">
        <fieldset>
            <legend>Trabalhe Conosco</legend>
            <label for="input-1">Nome:</label>
            <input type="text" placeholder="Informe seu nome" name="nome" /> <br>
            <label>Sexo:</label>
            <input type="radio" name="sexo" value="Masculino">Masculino
            <input type="radio" name="sexo" value="Feminino">Feminino<br>
            <label>Idioma:</label>
            <input type="checkbox" name="idioma" value="Inglês">Inglês
            <input type="checkbox" name="idioma" value="Espanhol">Espanhol<br>
            <label>Vaga:</label>
            <select id="vaga">
                <option value="Desenvolvedor">Desenvolvedor</option>
                <option value="Gerente de TI">Gerente de TI</option>
                <option value="Suporte de TI">Suporte de TI</option>
            </select> <br>
            <input type="submit" value="Enviar" id="btnEnviar" />
            <input type="submit" style="visibility: hidden;" value="Cancelar"
                   id="btnCancelar" />
        </fieldset>
    </form>
</body>
</html>
```

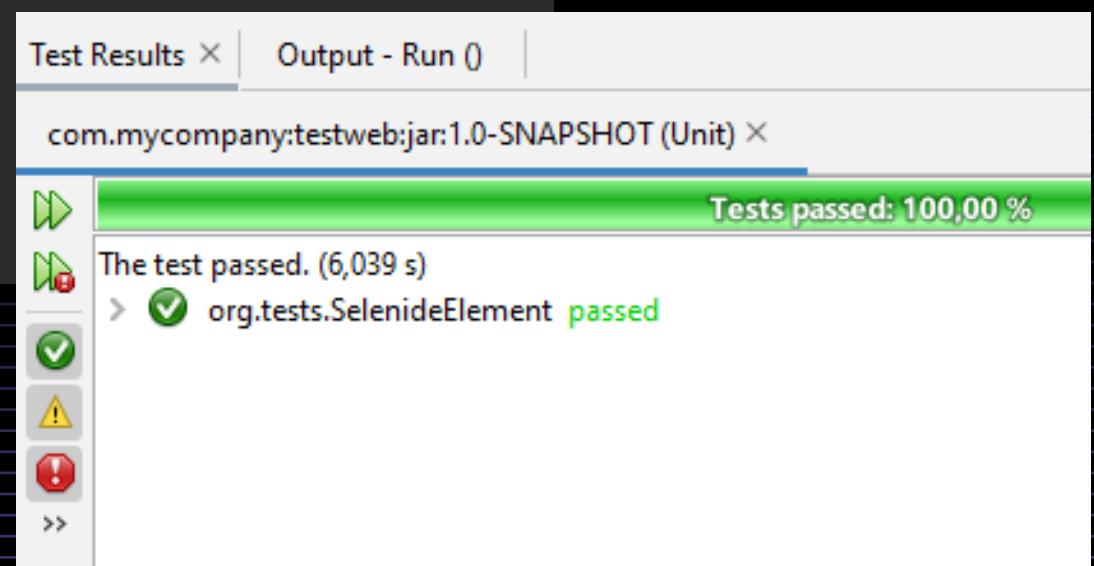
```
@Test
public void testSelenideElement() {
    open("http://localhost/sites/element.html");
    $("#btnEnviar").should(visible);

    assertEquals(true, $("#btnEnviar").exists());
    assertEquals("Inglês", $(By.name("idioma")).val());
    assertEquals("Desenvolvedor", $(By.id("vaga")).getSelectedValue());

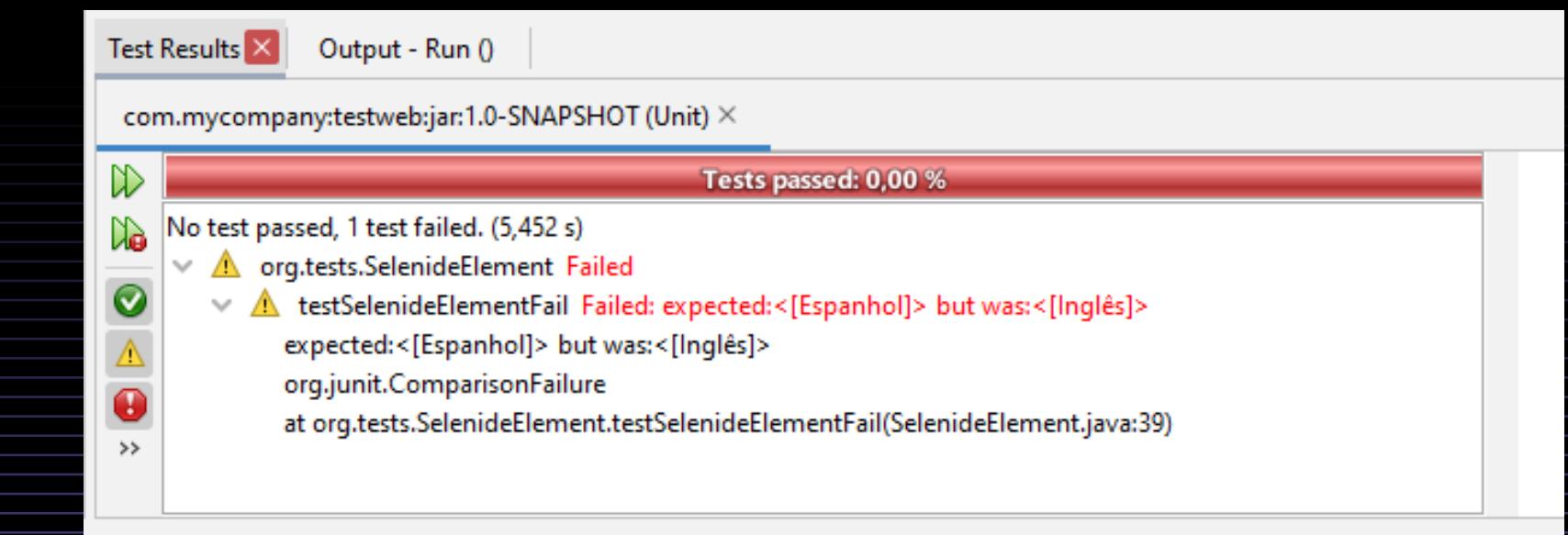
    $("#btnEnviar").pressEnter();

    $(By.name("nome")).setValue("Fabio");

    $(By.id("vaga")).selectOptionByValue("Suporte de TI");
}
```



```
@Test  
public void testSelenideElementFail() {  
    open("http://localhost/sites/element.html");  
    assertEquals("Espanhol", $(By.name("idioma")).val());  
    assertEquals("Suporte de TI", $(By.id("vaga")).getSelectedValue());  
}
```



Condições

Muitas aplicações web usam AJAX para mudar partes do site de forma dinâmica. Selenide fornece uma classe chamada Condition que, aguarda por algum tempo limite pré-definido a renderização dos elementos.

- **visible/appear**: Condição que indica que o elemento HTML é visível na página web;
- **present/exist**: Condição que indica que o elemento HTML existe na página web;
- **hidden/disappear/not(visible)**: Condição que indica que o elemento HTML não é visível na página web;
- **readonly**: Condição que indica que o elemento HTML permite somente a leitura na página web;
- **attribute**: Condição que indica que algum atributo está presente em um elemento HTML;
- **name**: Condição que indica que o elemento HTML possui um determinado valor de name;
- **value**: Condição que indica que o elemento HTML possui um determinado valor de value;

Condições

Por padrão, o tempo limite do Selenide para aguardar é de **4 segundos**, mas pode ser configurado.

`Configuration.timeout = 6000;`

- **type**: Condição que indica que o elemento HTML possui um determinado valor de type;
- **id**: Condição que indica que o elemento HTML possui um determinado valor de id;
- **empty**: Condição que indica que o elemento HTML está vazio;
- **cssClass**: Condição que indica que o elemento HTML possui um determinado valor de classe;
- **focused**: Condição que indica que o elemento HTML está em foco;
- **enabled**: Condição que indica que o elemento HTML está habilitado;
- **disabled**: Condição que indica que o elemento HTML está desabilitado;

Condições

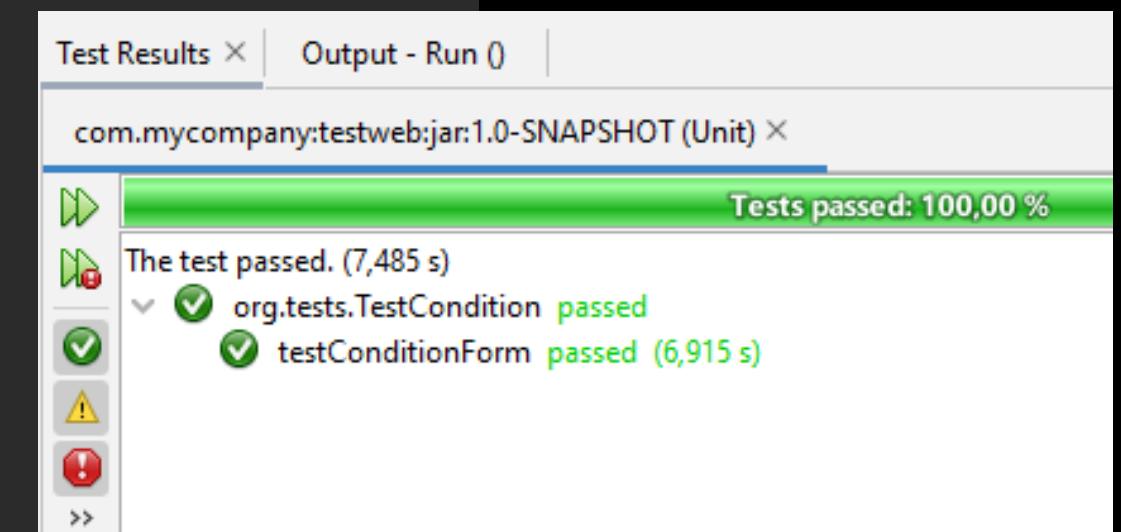
- **selected**: Condição que indica que o elemento HTML está selecionado;
- **text**: Condição que indica que o elemento HTML possui um determinado valor ou texto sem levar em consideração letras maiúsculas e minúsculas;
- **textCaseSensitive**: Condição que indica que o elemento HTML possui um determinado valor ou texto considerando letras maiúsculas e minúsculas.

Arquivo "*condition.html*"

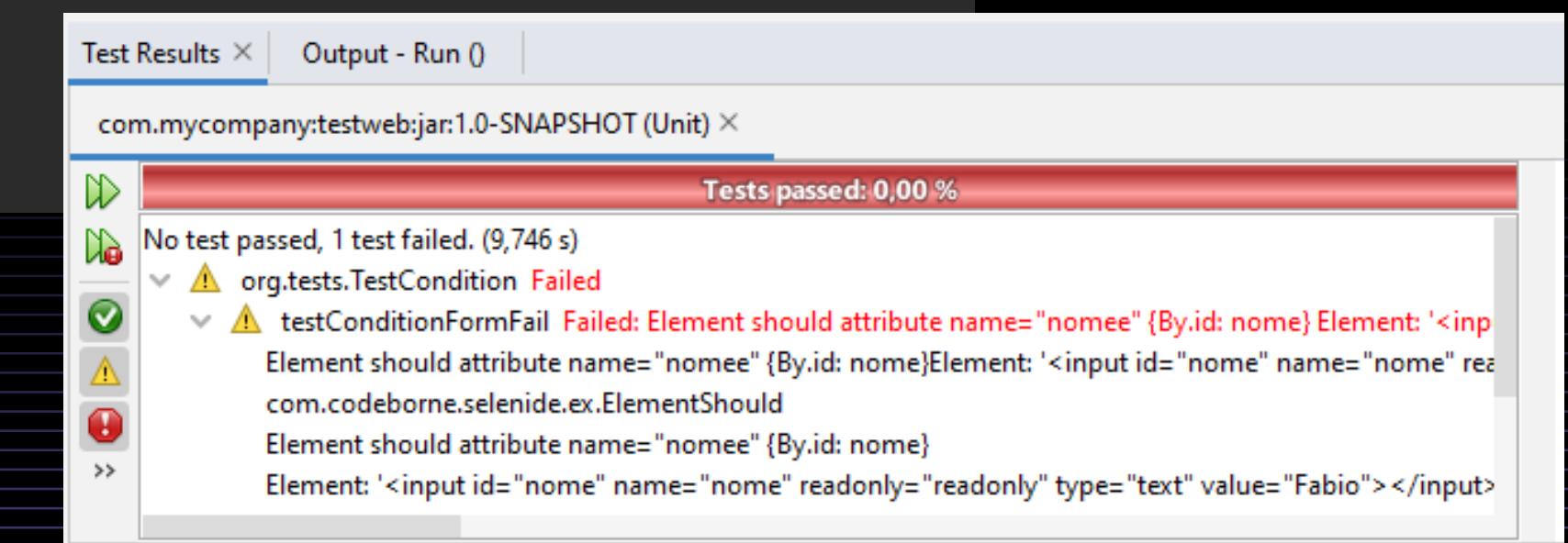
```
<html>
<head>
<meta charset="UTF-8">
<title>Condições</title>
</head>
<body>
<form id="meuForm">
    <label for="nome" >Nome:</label>
    <input type="text" id="nome" value="Fabio" readonly="readonly"
           name="nome"><br/>
    <label for="email" >Email:</label>
    <input type="email" autofocus="autofocus" id="email" >
</form>

    <div class="estilo_rodape" id="rodape"></div>
    <p id="meuParagrafo">Meu texto</p>
</body>
</html>
```

```
@Test  
public void testConditionForm() {  
    open("http://localhost/sites/condition.html");  
  
    $(By.id("meuForm")).should(visible);  
    $(By.id("meuForm")).should(exist);  
  
    $(By.id("nome")).should(name("nome"));  
    $(By.id("nome")).should(value("Fabio"));  
    $(By.id("nome")).should(type("text"));  
  
    $(By.id("email")).should(focused);  
  
    $(By.id("email")).should(enabled);  
  
    $(By.id("meuParagrafo")).should(textCaseSensitive("Meu texto"));  
}
```



```
@Test  
public void testConditionFormFail() {  
    open("http://localhost/sites/condition.html");  
  
    $(By.id("nome")).should(name("nomee"));  
    $(By.id("nome")).should(value("Maria"));  
    $(By.id("nome")).should(type("number"));  
  
    $(By.id("email")).should(disabled);  
}
```



Coleção de Elementos

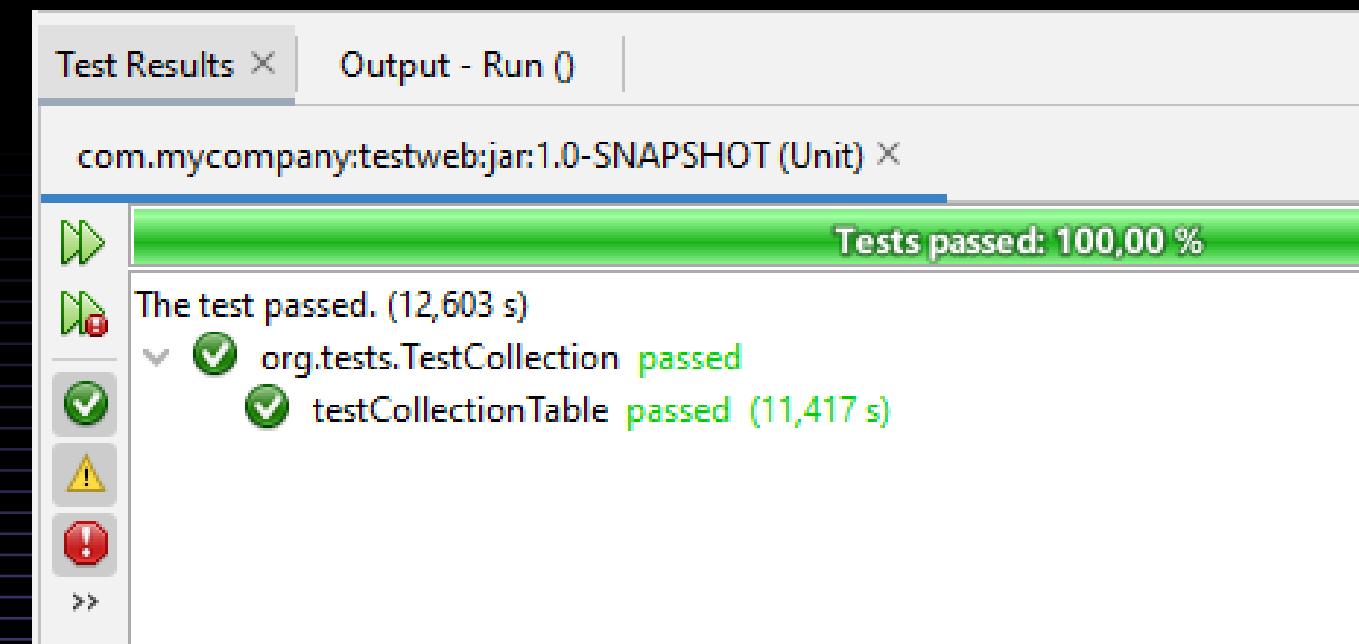
Como a pagina (html) é construida por varios elementos, selenide fornece maneiras úteis para trabalhar com coleções de elementos.

- **shouldBe**: Verifica o que o elemento HTML tem em determinada condição;
- **shouldHave**: Verifica o que o elemento html deve ter em determinada condição;
- **find**: Busca alguma informação no elemento;
- **filter**: Filtra alguma informação no elemento;
- **exclude**: Filtra todos os elementos menos determinados elementos de um condição.

Arquivo "*collection.html*"

```
<html>
<head>
<meta charset="UTF-8">
<title>Coleção de Elementos</title>
</head>
<body>
    <table border="1" id="tblGastos" style="width: 100%">
        <caption>Gastos mensais</caption>
        <tr>
            <th>Mês</th>
            <th>Gasto (R$)</th>
        </tr>
        <tr>
            <td>Janeiro</td>
            <td>R$250</td>
        </tr>
        <tr>
            <td>Fevereiro</td>
            <td>R$500</td>
        </tr>
        <tr>
            <td>Março</td>
            <td>R$5000</td>
        </tr>
    </table>
</body>
</html>
```

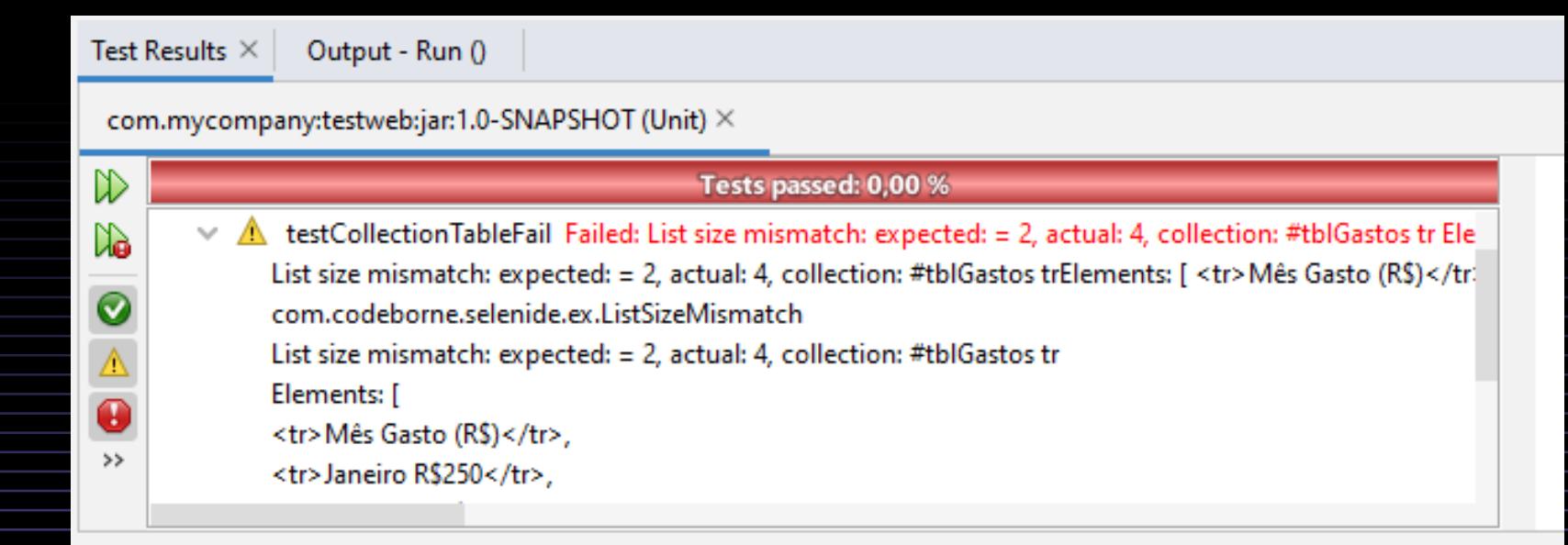
```
@Test  
public void testCollectionTable() {  
    open("http://localhost/sites/collection.html");  
    $$("#tblGastos tr").shouldHave(size(4));  
  
    assertEquals(true, $$("#tblGastos tr td").findBy(text("R$5000")).exists());  
    assertEquals(false, $$("#tblGastos tr td").filterBy(text("Janeiro")).isEmpty());  
  
}
```



```
@Test
public void testCollectionTableFail() {
    open("http://localhost/sites/collection.html");
    $$("#tblGastos tr").shouldHave(size(2));

    assertEquals(true, $$("#tblGastos tr td").findBy(text("R$10")).exists());
    assertEquals(true, $$("#tblGastos tr td").filterBy(text("Janeiro")).isEmpty());

}
```



Teste de fluxo!

```
@Test  
public void testPortalAluno() {  
    open("https://portal.ufsm.br/aluno/");  
  
    $(By.name("j_username")).setValue("201921050");  
    $(By.name("j_password")).setValue(senha);  
    $(By.name("enter")).pressEnter();  
  
    $(("#item_institucional")).click();  
    $$("#popup_institucional li a").findBy(text("Biblioteca")).click();  
  
    $(By.id("geral")).setValue("qualidade de software");  
    $(By.id("search-btn1")).pressEnter();  
  
    $$(".table tbody tr td").findBy(text("ABNT NBR ISO/IEC 14598-6 : engenharia de software : " + "avaliacao de produto : parte 6 : " + "documentacao de modulos de avaliacao / ")).should(exist).screenshot();  
}
```

Teste de fluxo!

```
@Test
public void testPortalAlunoFail(){
    open("https://portal.ufsm.br/aluno/");

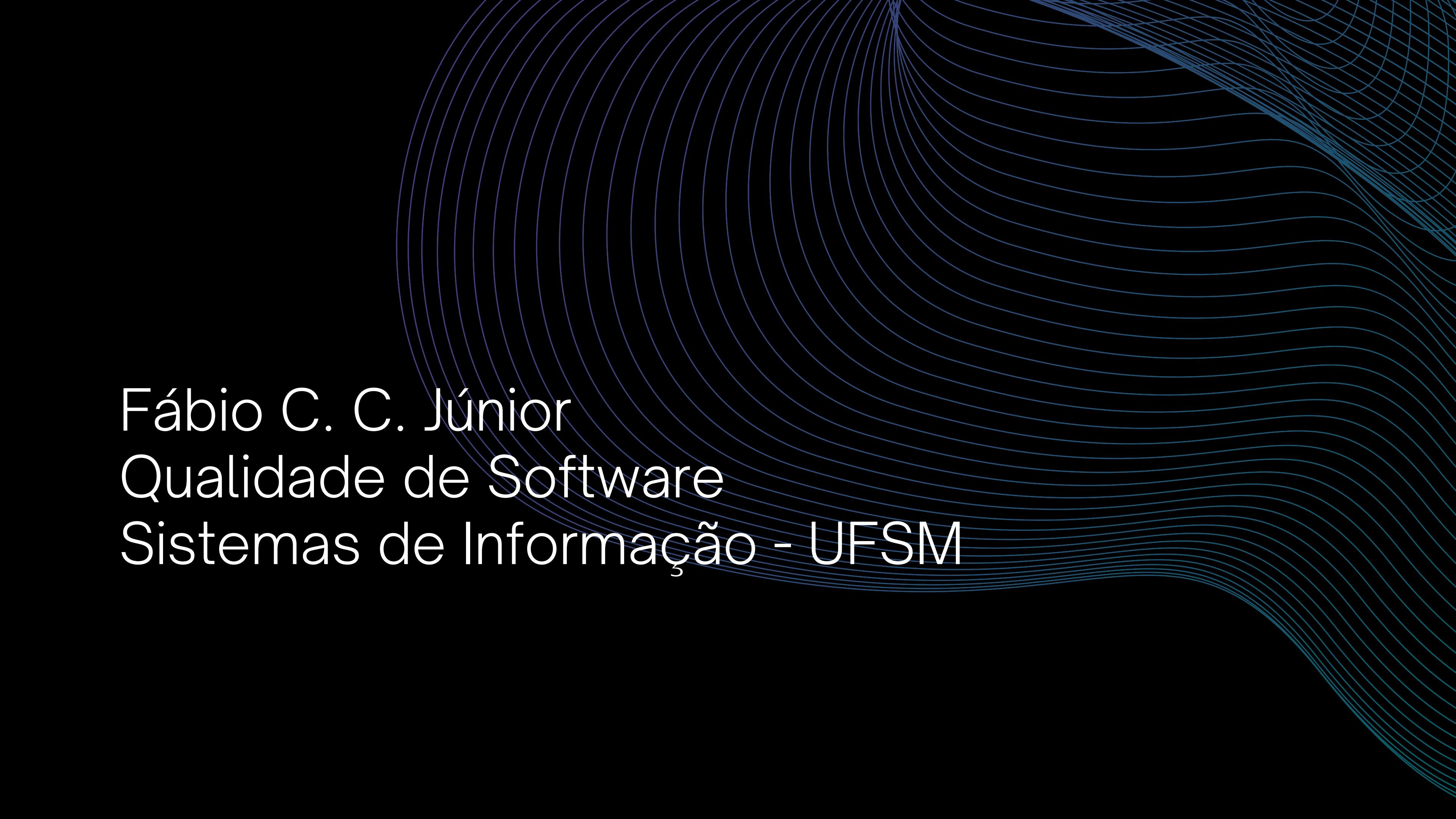
    $(By.name("j_username")).setValue("201921050");
    $(By.name("j_password")).setValue(senha);
    $(By.name("enter")).pressEnter();

    $(("#item_institucional")).click();
    $$("#popup_institucional li a").findBy(text("Biblioteca")).click();

    $(By.id("geral")).setValue("qualidade de software");
    $(By.id("search-btn1")).pressEnter();

    $$(".table tbody tr td").findBy(text("selenide")).should(exist).screenshot();
}

}
```



Fábio C. C. Júnior
Qualidade de Software
Sistemas de Informação - UFSM