03

Testando o cadastro de leilão

Transcrição

Agora que conseguimos montar o fluxo de navegação até o formulário de cadastro de um novo leilão, podemos continuar o desenvolvimento de teste para testarmos, de fato, se o cadastro está funcionando corretamente. Continuando nosso método, o próximo passo será preenchermos os campos do formulário ("nome", "valor inicial" e "data de abertura") e enviarmos o formulário. Por fim, queremos verificar se o leilão foi cadastrado com sucesso na página de listagem de leilões.

Tendo conseguido a `paginaDeCadastro`, podemos criar um método `cadastrarLeilao()` que receberá as informações do formulário por parâmetro. Como precisaremos fazer um *assert* utilizando esses valores, criaremos variáveis para guardá-los. A nossa aplicação já possui vários leilões cadastrados, portanto, para recuperarmos as informações corretamente e evitarmos conflitos, cadastraremos os leilões da seguinte forma:

- Leilão do dia 19/11/2020
- 500
- 19/11/2020

A data é a data atual da gravação do curso. Dessa forma, teremos uma descrição precisa das informações do teste. No método de teste, criaremos a variável `hoje` do tipo `String` recebendo a chamada de `LocalDate.now()`, de modo a recebermos a data atual. Formataremos a data recebida usando

`format(DateTimeFormatter.ofPattern())` , e passando como parâmetro a máscara desejada, `"dd/mm/yyyy"`

Em seguida, criaremos outra string `nome` que receberá `"Leilao do dia"` concatenado com o conteúdo da variável `hoje` . Por fim, teremos uma string `valor` recebendo `500.00` . Passaremos essas variáveis na chamada de `cadastrarLeilao()` .

```
public void deveriaCadastrarLeilao() {
    LoginPage paginaDeLogin = new LoginPage();
    paginaDeLogin.preencheFormularioDeLogin("fulano", "pass");
    this.paginaDeLeiloes = paginaDeLogin.efetuaLogin();
    CadastroLeilaoPage paginaDeCadastro = paginaDeLeiloes.carrega

    String hoje = LocalDate.now().format(DateTimeFormatter.ofPatt
    String nome = "Leilao do dia " + hoje;
    String valor = "500.00";
    paginaDeCadastro.cadastrarLeilao(nome, valor, hoje);

}
```

COPIAR CÓDIGO

Em `CadastroLeilaoPage` , criaremos o método `cadastrarLeilao()` recebendo esses três parâmetros, que renomearemos para `nome` , `valorInicial` e `dataAbertura` .

```
public void cadastrarLeilao(String nome, String valorInicial, Str

}
```

COPIAR CÓDIGO

Anteriormente nós aprendermos a preencher os campos de um formulário usando `this.broser.findElement(By.id())` para encontrarmos o elemento pelo seu ID e `sendKeys()` para passarmos o texto, começando pelo `nome`. Inspecionando a página de cadastro de leilões no navegador, faremos que o input referente a esse campo tem o ID `nome`, enquanto o "valor inicial" possui o id `valorInicial` e a "data abertura" o id `dataAbertura`. Com essas informações, preencheremos todos os nossos campos.

```
public void cadastrarLeilao(String nome, String valorInicial, Str
    this.browser.findElement(By.id("nome")).sendKeys(nome);
    this.browser.findElement(By.id("valorInicial")).sendKeys(valo
    this.browser.findElement(By.id("dataAbertura")).sendKeys(data
}
```

COPIAR CÓDIGO

Ao invés de só preenchermos o formulário e termos outro método para enviá-lo, faremos ambas essas ações no `cadastrarLeilao()`. Vimos que existem diversas maneiras de enviar um formulário, como clicar no botão, chamar o método `submit()` a partir do objeto `form` ou a partir de um input. Como nosso botão "Salvar" possui um ID, faremos da primeira forma.

Incluiremos no código mais um `findElement()`, dessa vez buscando pelo ID `button-submit`, e em seguida chamaremos o método `submit()`.

```
public void cadastrarLeilao(String nome, String valorInicial, Str
    this.browser.findElement(By.id("nome")).sendKeys(nome);
    this.browser.findElement(By.id("valorInicial")).sendKeys(valo
    this.browser.findElement(By.id("dataAbertura")).sendKeys(data
    this.browser.findElement(By.id("button-submit")).submit();
}
```

COPIAR CÓDIGO

Conseguimos finalizar um método que preenche os campos com as informações recebidas por parâmetro e envia o formulário. Quando chamarmos o método, seremos redirecionados para a lista de leilões. Portanto, o `cadastrarLeiloes()` pode ser alterado para devolver um objeto do tipo `LeiloesPage`, retornando uma nova instância desse objeto recebendo o mesmo `browser` (de modo a não abrirmos uma nova janela do navegador).

```
public LeiloesPage cadastrarLeilao(String nome, String valorInic
    this.browser.findElement(By.id("nome")).sendKeys(nome);
    this.browser.findElement(By.id("valorInicial")).sendKeys(valo
    this.browser.findElement(By.id("dataAbertura")).sendKeys(data
    this.browser.findElement(By.id("button-submit")).submit();

    return new LeiloesPage(browser);
}
```

COPIAR CÓDIGO

No teste `deveriaCadastrarLeilao()`, podemos reatribuir a variável `paginaDeLeiloes` recebendo o retorno do método `cadastrarLeilao()`.

```
public void deveriaCadastrarLeilao() {
    LoginPage paginaDeLogin = new LoginPage();
    paginaDeLogin.preencheFormularioDeLogin("fulano", "pass");
    this.paginaDeLeiloes = paginaDeLogin.efetuaLogin();
    CadastroLeilaoPage paginaDeCadastro = paginaDeLeiloes.carrega

    String hoje = LocalDate.now().format(DateTimeFormatter.ofPat
    String nome = "Leilao do dia " + hoje;
    String valor = "500.00";
```

```
        this.paginaDeLeiloes = paginaDeCadastro.cadastrarLeilao(nome,
    }
```

COPIAR CÓDIGO

Agora precisamos verificar se tudo ocorreu corretamente, o que pode ser feito verificando se a última linha da tabela possui as informações que foram cadastradas. Para isso, faremos um `Assert.assertTrue()` chamando um novo método `paginaDeLeiloes.isLeilaoCadastrado()`, que por sua vez receberá as variáveis `nome`, `valor` e `hoje`.

```
    public void deveriaCadastrarLeilao() {
        LoginPage paginaDeLogin = new LoginPage();
        paginaDeLogin.preencheFormularioDeLogin("fulano", "pass");
        this.paginaDeLeiloes = paginaDeLogin.efetuaLogin();
        CadastroLeilaoPage paginaDeCadastro = paginaDeLeiloes.carrega

        String hoje = LocalDate.now().format(DateTimeFormatter.ofPatt
        String nome = "Leilao do dia " + hoje;
        String valor = "500.00";

        this.paginaDeLeiloes = paginaDeCadastro.cadastrarLeilao(nome,
        Assert.assertTrue(paginaDeLeiloes.isLeilaoCadastrado(nome, va

    }
```

COPIAR CÓDIGO

Criaremos o método `isLeilaoCadastrado()` em `LeiloesPage`. Nele, precisaremos recuperar a última linha de uma tabela. As linhas não possuem IDs, mas a tabela por ser única, pode possuir, ainda que atualmente a nossa não possua. Sendo

assim, abriremos o arquivo `index.html` dentro do endereço `src/main/resources/templates/leilao` e adicionaremos o ID `tabela-leiloes` à tag `<table>`.

```
<table id="tabela-leiloes" class="table table-hover">
    <thead>
        <tr>
            <th scope="col">Nome</th>
            <th scope="col">Data de abertura</th>
            <th scope="col">Valor inicial</th>
            <th scope="col">Usuario</th>
            <th scope="col"></th>
            <th scope="col"></th>
        </tr>
    </thead>
```

COPIAR CÓDIGO

Para recuperarmos a última linha dessa tabela, podemos fazer uma navegação utilizando o seletor CSS. No método `isLeilaoCadastrado()`, chamaremos o `findElement()`, mas dessa vez usando `By.cssSelector()`. Passaremos para ele o parâmetro `"tabela-leiloes tbody tr:lastchilde"`, onde `#tabela-leiloes` é o ID da tabela, `tbody`, que representa as linhas do corpo, `tr`, que representa as linhas, e `last-child`, o pseudoelemento que nos retornará a última linha. Esses são conceitos de CSS, portanto conhecê-los pode facilitar esse tipo de trabalho.

```
public boolean isLeilaoCadastrado(String nome, String valor, Str
    this.browser.findElement(By.cssSelector("#tabela-leiloes tbo
    return false;
}
```

COPIAR CÓDIGO

Essa chamada devolverá um objeto `WebElement` que guardaremos em uma variável `linhaDaTabela` .

```
public boolean isLeilaoCadastrado(String nome, String valor, Str
    WebElement linhaDaTabela = this.browser.findElement(By.cssSe
    return false;
}
```

COPIAR CÓDIGO

Agora que temos a última linha, precisamos recuperar os `<td>` com cada informação (nome, data e valor). Para isso, a partir de `linhaDaTabela` , chamaremos um novo `findElement()` - ou seja, não é necessário buscar um elemento a partir do `browser` , sendo possível filtrar a partir de qual tag queremos realizar a busca.

Novamente utilizaremos a busca por um seletor CSS (`By.cssSelector()`) recuperando `td:nth-child(1)` , onde `1` representa a coluna que estamos buscando, e armazenaremos o retorno em uma variável `colunaNome` também do tipo `WebElement` . Repetiremos o processo para as outras colunas, `2` e `3` , armazenando seus conteúdos nas variáveis `colunaDataAbertura` e `colunaValorInicial` , respectivamente.

```
public boolean isLeilaoCadastrado(String nome, String valor, Str
    WebElement linhaDaTabela = this.browser.findElement(By.cssSe
    WebElement colunaNome = linhaDaTabela.findElement(By.cssSele
    WebElement colunaDataAbertura = linhaDaTabela.findElement(By.
    WebElement colunaValorInicial = linhaDaTabela.findElement(By.
    return false;
}
```

COPIAR CÓDIGO

Agora precisamos checar se tais elementos possuem as informações que foram passadas por parâmetro. Para isso, usaremos `colunaNome.getText()` para recuperarmos o texto do elemento e `equals()` para compararmos com o `nome` que foi passado por parâmetro. Usando `&&` , continuaremos as verificações para `colunaDataAbertura` e `colunaValorInicial` . Nesse ponto, também renomearemos a variável `hoje` para `data` .

```
public boolean isLeilaoCadastrado(String nome, String valor, Stri
    WebElement linhaDaTabela = this.browser.findElement(By.cssSel
    WebElement colunaNome = linhaDaTabela.findElement(By.cssSelec
    WebElement colunaDataAbertura = linhaDaTabela.findElement(By.
    WebElement colunaValorInicial = linhaDaTabela.findElement(By.

    return colunaNome.getText().equals(nome)
            && colunaDataAbertura.getText().equals(data)
            && colunaValorInicial.getText().equals(valor);
}
```

COPIAR CÓDIGO

A princípio é dessa maneira que verificaremos se um leilão foi cadastrado. Ao executarmos, nosso teste passará com sucesso. Esse teste foi um pouco mais trabalhoso, já que tivemos que passar por todo o processo de logar, carregar a página de leilões, carregar e preencher os campos do formulário, retornar à página de leilões, pegar as linhas e colunas da tabela e fazer as verificações com os valores encontrados.

Nesse processo, aprendemos alguns conceitos novos, como a recuperação de elementos por meio de um seletor CSS, e que é possível chamar o `findElement` a partir de objetos do tipo `WebElement` que não sejam o browser, filtrando e limitando a nossa consulta.