01

Esperando por Requisições Ajax

Transcrição

Começando deste ponto? Você pode fazer o DOWNLOAD (https://s3.amazonaws.com/caelum-online-public/PM-74/testes-de-sistema-cap4.zip) do projeto completo do capítulo anterior e continuar seus estudos a partir deste capítulo.

Vamos agora à página de detalhes de um leilão. Para isso, basta clicar em "Exibir" ao lado de qualquer leilão. Nessa página, o usuário pode ver os dados de um leilão e fazer lances nele. As imagens abaixo mostram um leilão sem nenhum lance, e com um lance adicionado:

Veja o que acontece quando damos um lance. A página não "pisca", mas é atualizada! Isso acontece porque nossa aplicação web fez uso do que chamamos de Ajax. Ou seja, a requisição aconteceu, mas ela foi por baixo dos panos, sem o usuário ver. Precisamos testar essa ação, e precisamos levar em conta que ela é executada via Ajax.

Nosso teste deve ser algo parecido com isso:

```
@Test
public void deveFazerUmLance() {

    leiloes.detalhes(1);

    lances.lance("José Alberto", 150);

    assertTrue(lances.existeLance("José Alberto", 150));
}
```

Não há segredo no método `lance()` do `DetalhesDoLeilaoPage`. Precisamos apenas selecionar um usuário no combo e preencher um valor. Nada de novo até então:

```
public class DetalhesDoLeilaoPage {

    private WebDriver driver;

    public DetalhesDoLeilaoPage(WebDriver driver) {
        this.driver = driver;
    }
```

```
    public void lance(String usuario, double valor) {
        WebElement txtValor = driver.findElement(By.name("lance.valor"));
        WebElement combo = driver.findElement(By.name("lance.usuario.id"));
        Select cbUsuario = new Select(combo);

        cbUsuario.selectByVisibleText(usuario);
        txtValor.sendKeys(String.valueOf(valor));

    }

}
```

Precisamos agora submeter o formulário. Mas dessa vez não vamos submeter pela caixa de texto. Veja o código-fonte da página! O botão não submete o formulário, ele é um simples "button" do HTML. Precisamos clicar nele. Para isso basta usar o método `click()`:

```
public class DetalhesDoLeilaoPage {

    private WebDriver driver;

    public DetalhesDoLeilaoPage(WebDriver driver) {
        this.driver = driver;
    }

    public void lance(String usuario, double valor) {
        WebElement txtValor = driver.findElement(By.name("lance.valor"));
        WebElement combo = driver.findElement(By.name("lance.usuario.id"));
        Select cbUsuario = new Select(combo);

        cbUsuario.selectByVisibleText(usuario);
        txtValor.sendKeys(String.valueOf(valor));

        driver.findElement(By.id("btnDarLance")).click();
    }

}
```

Precisamos agora verificar se o novo lance efetivamente apareceu na tela. Também da forma que já conhecemos:

```
    public boolean existeLance(String usuario, double valor) {
        return driver.getPageSource().contains(usuario)
                && driver.getPageSource().contains(String.valueOf(valor));
    }
```

Mas temos um problema. Uma requisição Ajax acontece por trás dos panos, e pode levar alguns segundos para terminar. Se invocarmos o método `existeLance()` e a requisição ainda não tiver voltado, o método retornará falso! Precisamos fazer com que esse método aguarde até a requisição terminar.

Para isso, pediremos ao Selenium para que espere alguns segundos até que a requisição termine. Isso é conhecido por explicit wait. Para usá-lo, precisamos fazer uso da classe `WebDriverWait`. No construtor, ela recebe o driver e a quantidade máxima de segundos a esperar. Em seguida, ela recebe a condição que faz esse tempo parar. No nosso caso,

a condição é se o texto aparecer. Para isso, faremos uso de uma outra classe, a `ExpectedConditions`, e passaremos a expectativa correta:

```
        Boolean temUsuario =
                new WebDriverWait(driver, 10)
                    .until(ExpectedConditions
                    .textToBePresentInElement(By.id("lancesDados"), usuario));
```

Vamos colocar o explicit wait dentro do método `existeLance()`, completando o código para verificar também se existe o valor:

```
    public boolean existeLance(String usuario, double valor) {
        Boolean temUsuario =
                new WebDriverWait(driver, 10)
                    .until(ExpectedConditions
                    .textToBePresentInElement(By.id("lancesDados"), usuario));

        if(temUsuario) return driver.getPageSource().contains(String.valueOf(valor));
        return false;
    }
```

Vamos agora ao teste. Repare que o cenário agora é maior: precisamos criar 2 usuários (um que é o dono do produto e outro para dar lance no produto) e um leilão. Vamos também limpar o cenário para facilitar nossa vida. Tudo isso no `@Before`:

```
public class LanceSystemTest {

    private WebDriver driver;
    private LeiloesPage leiloes;

    @Before
    public void inicializa() {
        this.driver = new FirefoxDriver();

                driver.get("http://localhost:8080/apenas-teste/limpa");

        UsuariosPage usuarios = new UsuariosPage(driver);
        usuarios.visita();
        usuarios.novo().cadastra("Paulo Henrique", "paulo@henrique.com");
        usuarios.novo().cadastra("José Alberto", "jose@alberto.com");

        leiloes = new LeiloesPage(driver);
        leiloes.visita();
        leiloes.novo().preenche("Geladeira", 100, "Paulo Henrique", false);
    }

    @Test
    public void deveFazerUmLance() {

        DetalhesDoLeilaoPage lances = leiloes.detalhes(1);

        lances.lance("José Alberto", 150);
```

```
        assertTrue(lances.existeLance("José Alberto", 150));
    }

}
```

Falta só um detalhe: o método `detalhes(1)` do LeiloesPage não está implementado. Precisamos fazer ele chegar na página de detalhes do produto para darmos os lances. Vamos pegar os detalhes do 1º item da lista. Precisamos pedir ao Selenium para achar todos os elementos da página com o texto Exibir, e clicar no primeiro deles. Vamos lá:

```
    public DetalhesDoLeilaoPage detalhes(int posicao) {
        List<WebElement> elementos = driver.findElements(By.linkText("exibir"));
        elementos.get(posicao - 1).click();

        return new DetalhesDoLeilaoPage(driver);
    }
```

Cuidado com maiúsculas e minúsculas! O link é "exibir" (tudo minúsculo). O Selenium é case-sensitive!

Pronto! Se executarmos o teste, ele passa! Acabamos de testar requisições feitas por Ajax!