

## Configurações do Selenium

### Transcrição

Agora nossos próximos passos são:

4) Navegar até a página desejada.

5) Interagir com a página.

Com isso fecharemos o entendimento dos nossos testes.

Escrutinando nossa classe `FiltraNegociacoesIT` vemos o método `verificaResultadosFiltradosNaTabela` anotado com `@Test` do JUnit. É este método que implementa nosso teste. Nele usamos métodos do webdriver para interagirmos com o browser, por exemplo:

```
driver.navigate().to(HTTP_LOCALHOST_888 + "/index.xhtml");
```

### Dependências no maven

Antes de configurarmos nosso Job, é importante vermos como foram feitas as configurações do Maven. Atualmente, ela foi organizada de uma forma exclusiva para entender apenas os testes de integração. Podemos verificar isso no arquivo `pom.xml` do nosso projeto.

No `pom.xml` temos a tag `profiles` que nada mais é do que uma amalgama de configurações que serão usadas apenas se o profile for especificado para ser usado. Há um profile chamado `testes-integracao`, que possui toda configuração específica para o teste de integração. Há dependências do Arquillian, do Tomcat embedded entre outros.

Na parte de `build`, `plugin`, há a dependência `maven-failsafe-plugin` voltado para testes de integração, diferente do `surefire` voltado para testes de unidade.

Você pode conferir a documentação do Maven sobre `build lifecycle` e conhecer outras fases de build do Maven

(<https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html>

(<https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html>)).