# Introdução ao Selenium IDE

**(por Elias Nogueira - Qualister)**

## Como instalar o Selenium IDE?

Como o Selenium IDE é um plugin para o Firefox só podemos instala-lo pelo Firefox.

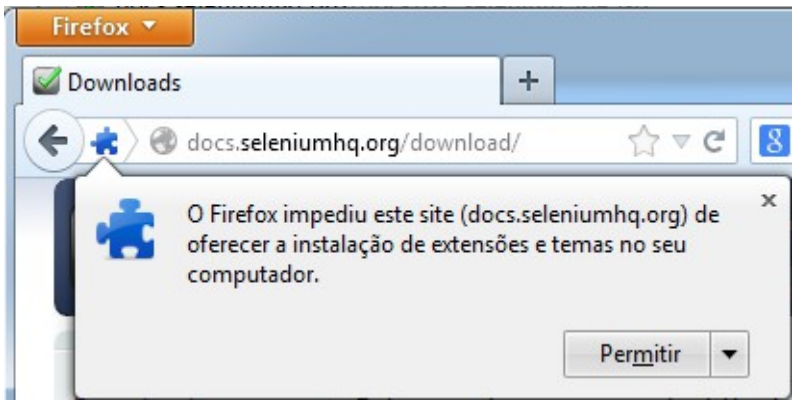
Utilize a página oficial de downloads do Selenium http://seleniumhq.org/download/

A primeira sessão chamada Selenium IDE conterá a descrição e um link referente a versão atual do Selenium IDE. Basta clicar neste link e seguir com a instalação do plugin.

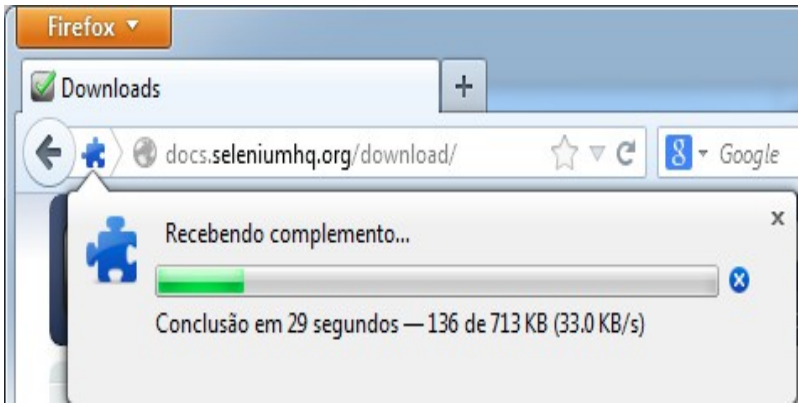
Observação

Sempre que há uma atualização do Firefox e o plugins instalado parar de funcionar, ou funcionar parcialmente, clique em um link chamada “unreleased”. Este link irá instalar a versão de desenvolvimento do plugin.


Uma vez lançado o plugin para a nova versão do Firefox, remova o plugin de desenvolvimento e instale o plugin suportado.
Uma mensagem do Firefox será apresentada perguntando se podemos permitir ou não a instalação do Selenium IDE. Clique em Permitir.




Aguarde enquanto o download do Selenium IDE é feito no Firefox

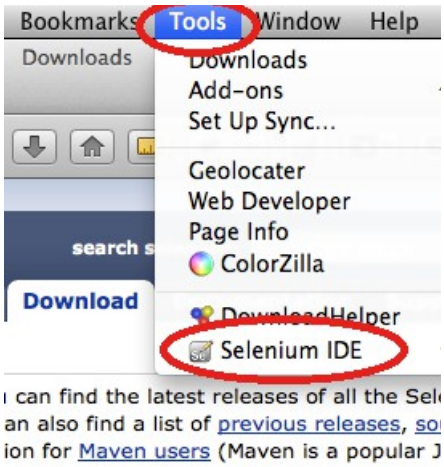



Uma janela contendo o plugin principal do Selenium IDE mais os formatadores (iremos aprender isso mais adiante) será apresentada. Clique em Instalar agora.


Após a instalação um pedido para reiniciar o Firefox será apresentado. Clique em Reiniciar.

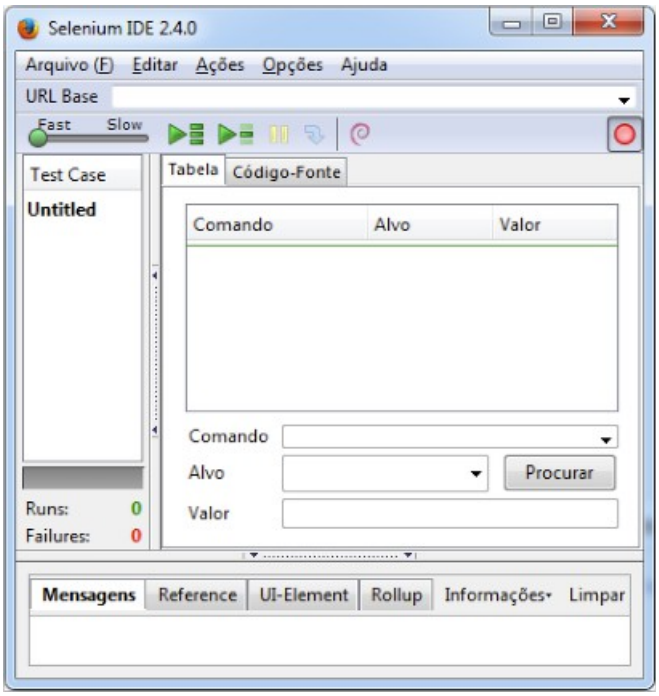
### Abrindo o Selenium IDE

Após a instalação podemos facilmente abri-lo pelo menu Firefox > Desenvolvedor web > Selenium IDE ou Ferramentas > Selenium IDE dependendo do sistema operacional que você está utilizando.




Exemplo do menu via Ferramentas > Selenium IDE no Firefox para MacOs

A janela do Selenium IDE será apresentada. No primeiro acesso o IDE abre uma página de Release Notes, que são as últimas modificações feitas nesta versão (adição de funcionalidades ou correção de bugs).

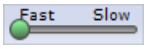


### Funcionalidades do Selenium IDE

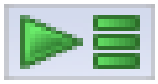
Como o plugin é uma IDE (Ambiente de Desenvolvimento Integrado) ele é dividido em quatro partes.

### Toolbar

A toolbar possui diversas funcionalidades importantes na utilização do Selenium IDE.

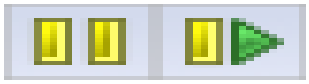

Speed Control: controla o quão rápido será a execução do script [2]



Run All: Executa uma suíte de teste que contenha múltiplos casos de teste contidos na suíte.

Run: Executa o caso de teste selecionado. Quando há apendas um caso de teste

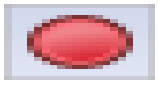
criado o botão Run All funciona do mesmo modo que o Run.



Pause/Resume: Permite parar e reiniciar a execução do caso de teste.

Step: Habilita a execução de cada passo (step by step) e deve ser utilizado apenas para questões de debug.

Apply Rollup Rules: Funcionalidade que permite a criação de tarefas repetitivas utilizando os comandos do Selenium em uma única ação



Record: Grava as ações do usuário no browser.

[2] Somente utilize o controle de velocidade para testar o seu script. Nunca utilize esta funcionalidade para executar o script de forma mais lenta, a fim de passar por testes que contenham requisições assíncronas (Ajax). Veremos a seguir qual funcionalidade é referente a requisições assíncronas.

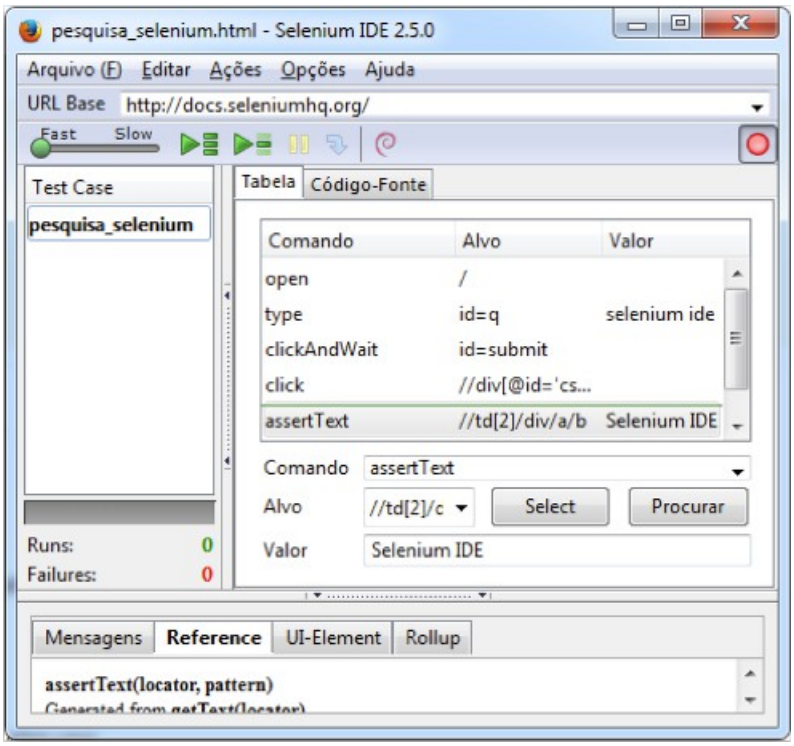
**Test Case**

O test Case é dividido em duas partes:

- Painel Test Case
- Abas Tabela e Código-fonte

O Painel Test Case exibe o caso de teste atual ou o conjunto de casos de teste (que irá configurar indiretamente uma suíte de teste).

Abas Tabela e Código-fonte irão exibir os comandos do Selenium e o código-fonte HTML dos comandos, respectivamente.




Há três colunas e três campos referentes a cada coluna:

- Comando: o comando executado pelo Selenium IDE
- Alvo: geralmente um elemento HTML
- Valor: quando o Alvo for uma elemento HTML o Valor pode ser o texto de comparação ou um texto para digitação, dependendo do comando

Podemos modificar qualquer um dos três itens a qualquer momento.

### Informações do script

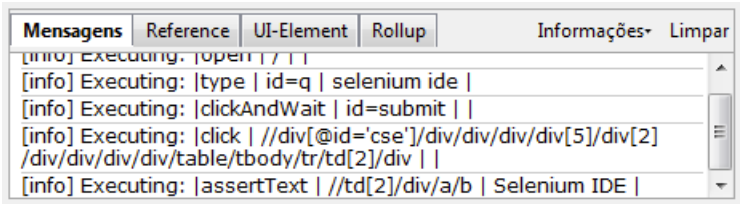
Esta sessão nos dá informações sobre diversos itens do script.

A aba Log (Mensagem) apresenta as informações de execução do script de teste e sobre erros.

A aba Reference apresenta a documentação de todos os comandos do Selenium, chamado de Selenese.

A aba UI-Element é uma funcionalidade onde podemos fazer o mapeamento entre nomes de elementos em uma página web.

A aba Rollup apresenta os comandos que foram agrupados através da funcionalidade Rollup Rules.




### Como gravar ações no Selenium IDE?

Quando iniciamos o Selenium IDE o botão Record ele já é apresentado pressionado, ou seja, ele já estará gravando todos os passos que executarmos no browser.

As ações gravadas são cliques em links, digitação de valores em caixas de texto, cliques em checkbox ou radio buttons e seleção de dados em uma combobox ou list.

Toda vez que desejarmos parar a gravação, iremos clicar novamente no botão Record.

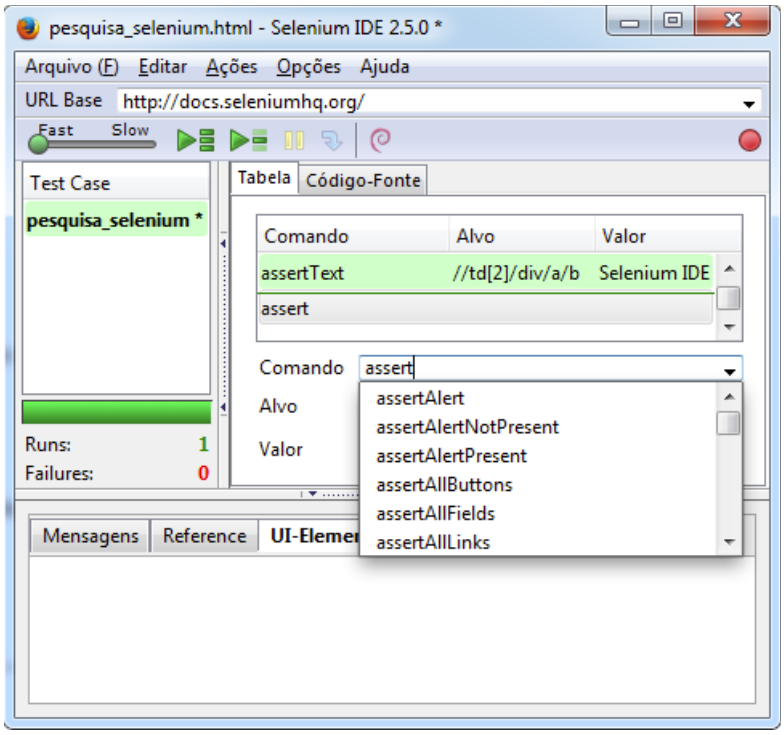
### Garantindo os resultados esperados

Após a gravação inicial do script é necessário colocar os pontos de controle, ou seja, os resultados esperados para garantir que o script tenha executado com sucesso.

No Selenium IDE há duas formas, quase idênticas, de garantir o(s) resultado(s) esperado(s):

- Verificação (verify): verifica se o resultado esperado está presente. Caso não esteja o Selenium IDE marca o passo com erro e continua a execução do script
- Asserção (asserts): garante que o resultado esperado está presente. Caso não esteja o Selenium IDE marca o passo com erro e para a execução do script.

Para descobrir todos os comandos referentes a verificação ou asserção basta digitar na caixa de texto **Comando** o comando iniciando com a palavra “assert” ou “verify”.



**Localizando Elementos HTML**

O Selenium IDE grava as ações executadas em uma página web através dos elementos HTML contidos na página.

No Selenium IDE há algumas formas de localizarmos os elementos HTML para uma interação, são eles:

**ID**

É provavelmente o tipo de localização mais comum e mais utilizado. Quando um elemento HTML possui o atributo id, o Selenium IDE logo utiliza o valor deste atributo para localizar o elemento.

Ex: Localizando um campo de usuário

```
<input type=”text” id=”username” />
```

Neste caso o Selenium IDE irá colocar na caixa de texto Alvo o valor id=username

**Name**

Quando o elemento HTML possui o atributo name o Selenium IDE utiliza o valor deste atributo para localizar o elemento.

Na grande maioria das situações o elemento possui os atributos id e name. Há poucos casos que iremos encontrar um elemento apenas com o atributo name.

Ex: Localizando um campo de comentários (text area)

```
<txtarea name=”comentarios” />
```

Neste caso o Selenium IDE irá colocar na caixa de texto Alvo o valor name=comentários

**Link**

Quando o elemento HTML é um link (um elemento tipo a) o Selenium IDE utiliza o texto do hyperlink para sua localização.

Ex: Link que apresenta o texto "Blog Qualister"

```
<a href=http://qualister.com.br/blog>Blog
Qualister</a>
```

Neste caso o Seleium IDE irá colocar na caixa de texto Alvo o valor link=Blog Qualister

**DOM**

O DOM – Document Oriented Model, representa um documento HTML e pode ser acessado através de Javascript.

Como a utilização de um DOM inicia com a palavra document não será necessário colocar um prefixo “ dom”, seguindo a lógica dos outros comandos.

Ex: Localizando um campo senha

```
<input type=”password” id=”passwd” />
```

Neste caso o Selenium IDE irá colocar na caixa de texto Alvo o valor document.getElementById('loginForm')

Para conhecer mais sobre o DOM, consulte este link http://www.w3schools.com/js/js_htmldom.asp (em inglês)

### CSS

CSS (Cascading Style Sheets) é uma linguagem para descrever e renderizar arquivos HTML ou XML. O CSS utiliza-se de seletores (selectors) para localizar, além de elementos a atributos, os mesmos através de seu estilo.

Ex: Localizando um campo através de seu estilo

```
<input
class="required" name="username" type="text"
/>
```

Neste caso o Selenium IDE irá colocar na caixa de texto Alvo o valor css=input.required

Para aprender mais sobre os tipos de seletores CSS, acesse a página http://www.w3.org/TR/css3-selectors/ (em inglês)

### XPATH

XPath é uma linguagem de consulta de nós em arquivos XML. Um arquivo HTML pode ser uma implementação do XML (XHTML) e também possui a mesma estrutura de um arquivo XML.

Ele possui algumas funções para facilitar a localização de elemento, que pode ser feita pelo elemento em si, atributos ou posição do elemento.

Todo comando de localização contendo XPATH deve começar com duas barras “//“

Ex: Localizando um campo por posição (localizando o Estado)

```
Cidade<input type=”text”
class=”required”>

Estado<input type=”text”
class=”required”>
```

Neste caso o Selenium IDE irá colocar na caixa de texto Alvo o valor //input[2]

### Comandos de Espera

Com a popularização das requisições assíncronas (XHTTP Requests) mais conhecidas como “Ajax” a execução do script de teste fica um pouco mais complexa. Quando temos que interagir com algum campo que, por exemplo, demora para ser apresentado em tela,

o script do Selenium IDE irá falhar, pois ele não efetua esperas quando gravamos um script.

Para resolver este problema o Selenium IDE disponibiliza uma série de comandos que iniciam com a palavra “waitFor” (esperar por).

Ex: se temos o seguinte código-fonte HTML de um elemento que demora alguns segundos para ser apresentado:

Para inserir um comando de espera primeiro analisamos como iremos criar esta abordagem. Neste caso iremos esperar que o elemento esteja presente na tela (apareça) para depois interagir.

Existe o comando chamado waitForElementPresent (esperar por um elemento presente) que necessita de um único parâmetro que é a forma de localização do elemento. As caixas de texto Comando e Alvo ficariam assim:

- Comand: waitForElementPresent
- Alvo: id=cep

Com isso O Selenium IDE irá esperar até 30 segundos (configurável acessando o Menu Opções > Opções na caixa de texto “Tempo de expiração padrão para os comandos gravados“) para passar para o próximo passo do script.

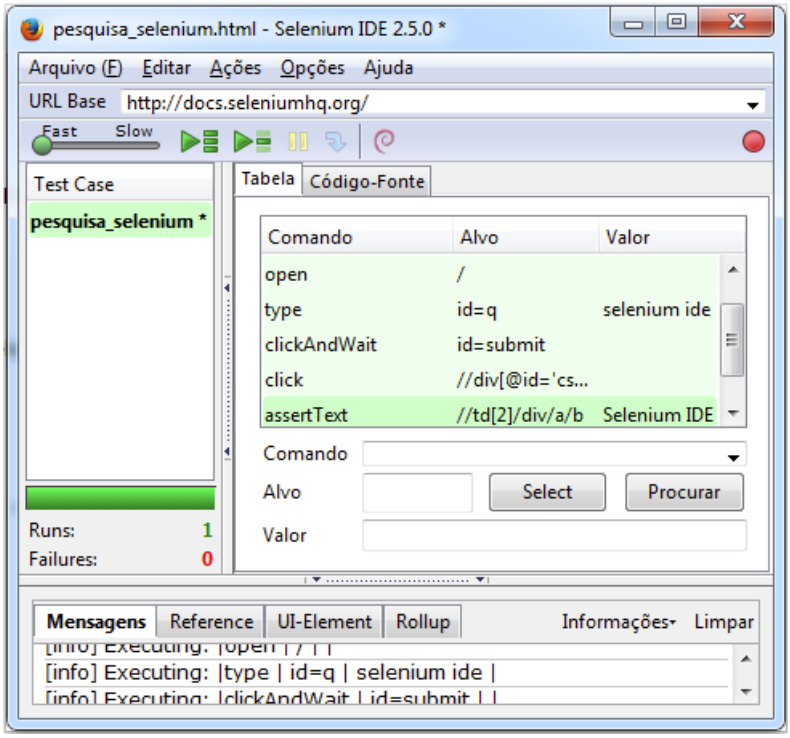
### Execução do script

Com o script pronto basta executar o script para garantir que o mesmo está de acordo com o teste que planejamos.

Para executar o script no Selenium IDE apenas clique no botão Run e todas as ações do script serão executadas no Firefox.

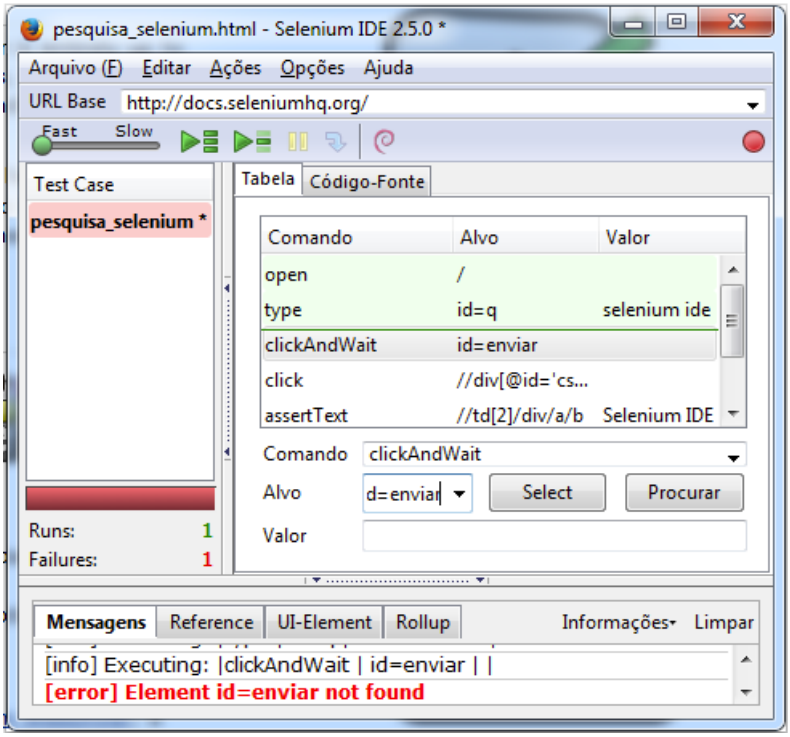
Ao final da execução o Selenium IDE irá identificar o script com sucesso ou falha.

No caso abaixo o script executado foi executado com sucesso:




A imagem abaixo apresenta um script com erro. Note que o passo que ocorreu o erro está com a cor vermelha, bem como o indicativo de qual problema ocorreu na aba Mensagens.

Neste caso ele não conseguiu encontrar o elemento de id=enviar



**Conclusão**

Neste tutorial foram apresentadas funcionalidades do Selenium IDE.

Fizemos a  instalação, a utilização e as principais ações que necessitamos fazer para utiliza-lo de forma mais assertiva.

# ATIVIDADE A SER DESENVOLVIDA

1. Instalar e abrir o Selenium IDE;
2. Gravar um script com os passos que você deseja automatizar na forma de um teste.
    1. Para executar este passo escolha um projeto do qual já fizemos em aulas anteriores, suba-o com o Tomcat dentro do eclipse.
    2. Para criar um teste na forma de um programa Java, você deve inicialmente capturar as ações relacionadas ao uso do trecho do sistema que você quer testar, para depois exportar para o formato Java. Por exemplo, para criar um Procedimento de Teste para o login no sistema, devo gravar as ações de login nesse sistema.
    3. A linguagem utilizada pelo Selenium é simples e intuitiva. Ao clicar no campo “comando” podemos ver a lista de comandos disponíveis.
3. Exportar o teste para Java
    1. Após criar o teste no Selenium, é necessário exporta-lo para Java. Com isso teremos as mesmas ações capturadas anteriormente, porém utilizando uma API Java para execução de comandos.
4. Criar um Procedimento de Teste Genérico
    1. Após termos o arquivo Java, gerado a partir do script com a captura das ações do teste, precisamos criar um Procedimento de Teste que possa ser utilizado para testar todas as situações envolvidas nessa parte do sistema. Isso incluir testar o “caminho feliz”, assim como todas as possíveis exceções que possam ser geradas a partir da execução da funcionalidade.
    2. Você pode por exemplo criar vários logins corretos, com o intuito de verificar o nível de acesso de cada usuário, assim como tentativas de login utilizando algum valor inválido, que causa a exibição de mensagens de erro.
5. Criar os casos de teste automatizados
    1. A partir desse procedimento de teste é possível fazer uma série de verificações. Conforme já mencionado, as verificações estão associadas ao “caminho feliz” da funcionalidade, assim como os prováveis casos excepcionais, e que normalmente causam a emissão de alguma mensagem ao usuário. Assim, faz parte do processo de teste pensar nesses casos, para em seguida automatizar sua execução.
    2. É interessante salientar que a execução dos testes via selenium exige que uma instância da classe DefaultSelenium seja criada.
        1. @BeforeClass
        2. public static void iniciaBrowser() throws Exception {
        3. browser = new DefaultSelenium("localhost",
        4. 4444, "*firefox", "http://localhost/**<<suaAPP>>**/");
        5. browser.start();

6. login = new LoginProcedimentosDeTeste(browser);
7. }

6. Execução do Teste
    1. Os testes criados com o Selenium são testes que utilizam o JUnit, de forma similar aos testes de unidade. Sua execução acontece da mesma forma que os testes de unidade, clicando com o botão direito do mouse em cima da classe e solicitando sua execução via JUnit. No entanto, existe uma diferença: para que o teste execute, é necessário que exista uma instância do Selenium RC executando na mesma máquina que contém o sistema a ser testado.
    2. O Selenium RC é a parte do selenium responsável por executar cada um dos comandos existentes nos procedimentos de teste. Assim, quando se solicita a digitação de um texto em um comando de uma tela, na verdade emitimos essa solicitação ao Selenium RC, que por sua vez se comunica com o navegador, via funções JavaScript, e executa o comando solicitado.
    3. O Selenium RC é uma extensão do selenium e precisa ser iniciado antes da execução do teste. A iniciação dele pode ser feita a partir da execução do jar com o servidor: java –jar selenium-server.jar. Caso o selenium rc não esteja ativo, a execução dos comandos não será executada com sucesso. O Selenium Rc pode ser obtido a partir do seguinte endereço:

        http://selenium-rc.openqa.org/download.jsp.
7. Chamar o instrutor e demonstrar o teste funcionando.