SELENIUM

28/03/2011

Márcio Delamaro
Harry Trinta

O que é a Selenium?

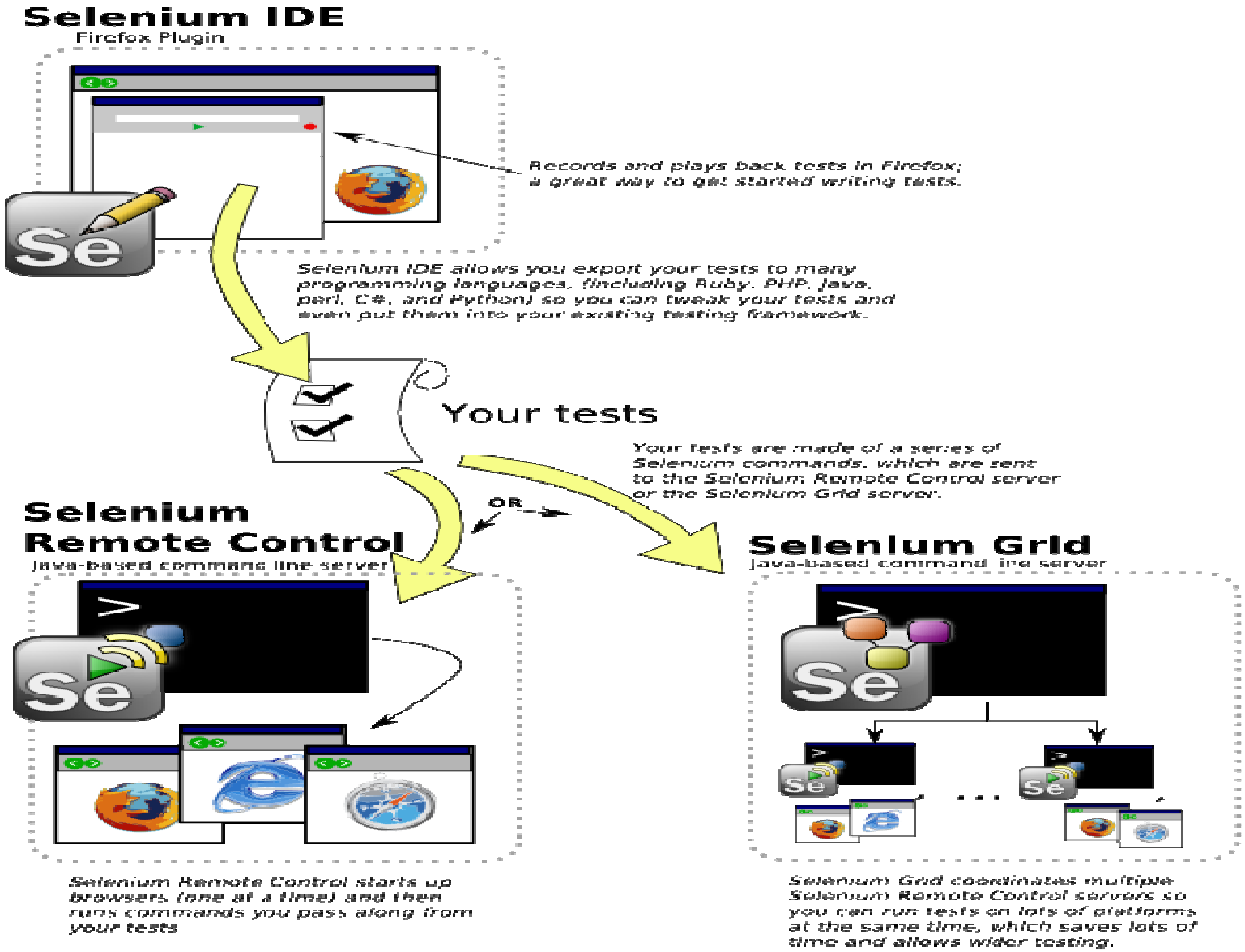
- Selenium é uma ferramenta desenvolvida para a automação de testes em aplicações Web
- Suporta diversos browsers:
  - Firefox
  - Internet Explorer
  - Safari
  - Opera
  - Chrome
- .. e diversos SOs
  - Windows, Linux, Solaris

O que é a Selenium?

- Tipos de Testes Possíveis:
  - Teste Funcional
  - Teste de Regressão
  - Teste de Desempenho

- Componentes:
  - Selenium IDE
  - Selenium RC (Remote Control)
  - Selenium Grid

- http://seleniumhq.org


Selenium IDE
Firefox Plugin
Records and plays back tests in Firefox; a great way to get started writing tests.
Selenium IDE allows you export your tests to many programming languages, (including Ruby, PHP, Java, perl, C#, and Python) so you can tweak your tests and even put them into your existing testing framework.
Your tests
Your tests are made of a series of Selenium commands, which are sent to the Selenium Remote Control server or the Selenium Grid server.
OR
Selenium Remote Control
Java-based command line server
Selenium Remote Control starts up browsers (one at a time) and then runs commands you pass along from your tests
Selenium Grid
Java-based command line server
Selenium Grid coordinates multiple Selenium Remote Control servers so you can run tests on lots of platforms at the same time, which saves lots of time and allows wider testing.

5

Selenium IDE

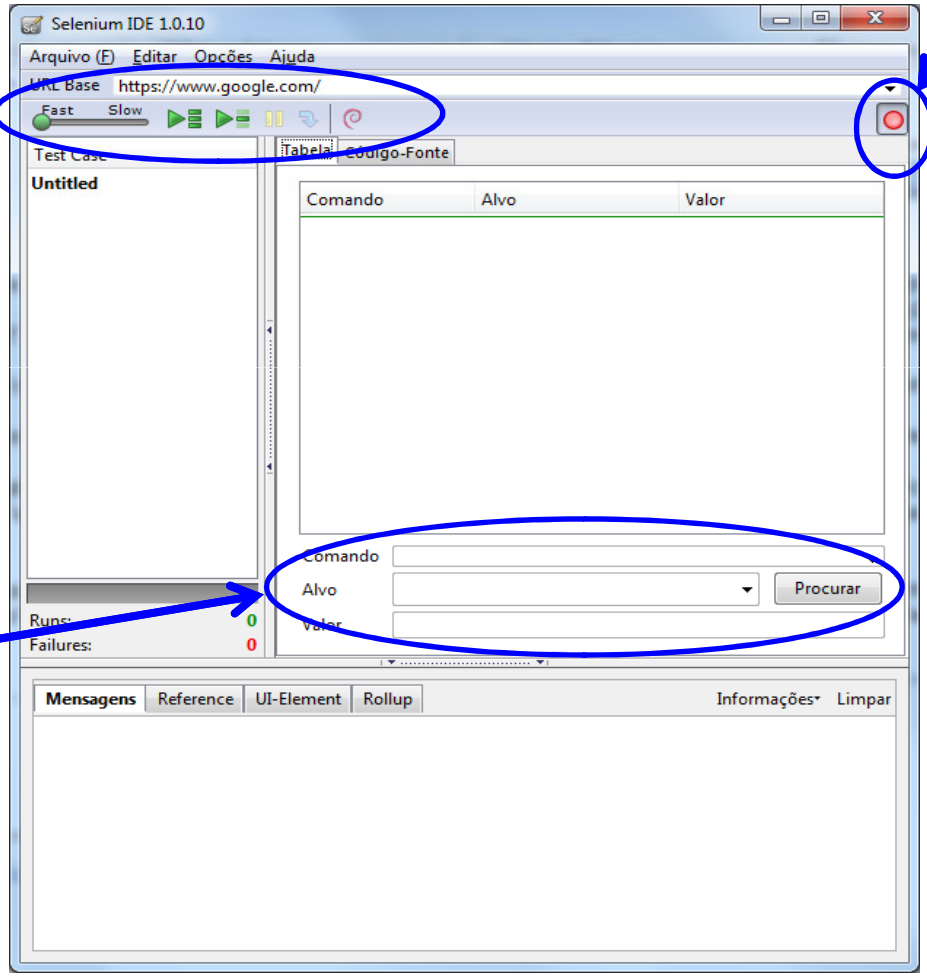
Selenium IDE

- Selenium IDE é um plug-in para o Firefox capaz de gravar e reproduzir testes
- Com ele é possível exportar o teste gravado para diversas linguagens, por exemplo, HTML, Java, .NET, Perl, Ruby, etc
- Os testes exportados podem ser executados em qualquer navegador e qualquer plataforma usando o componente Selenium RC

Selenium IDE

Execução dos comandos

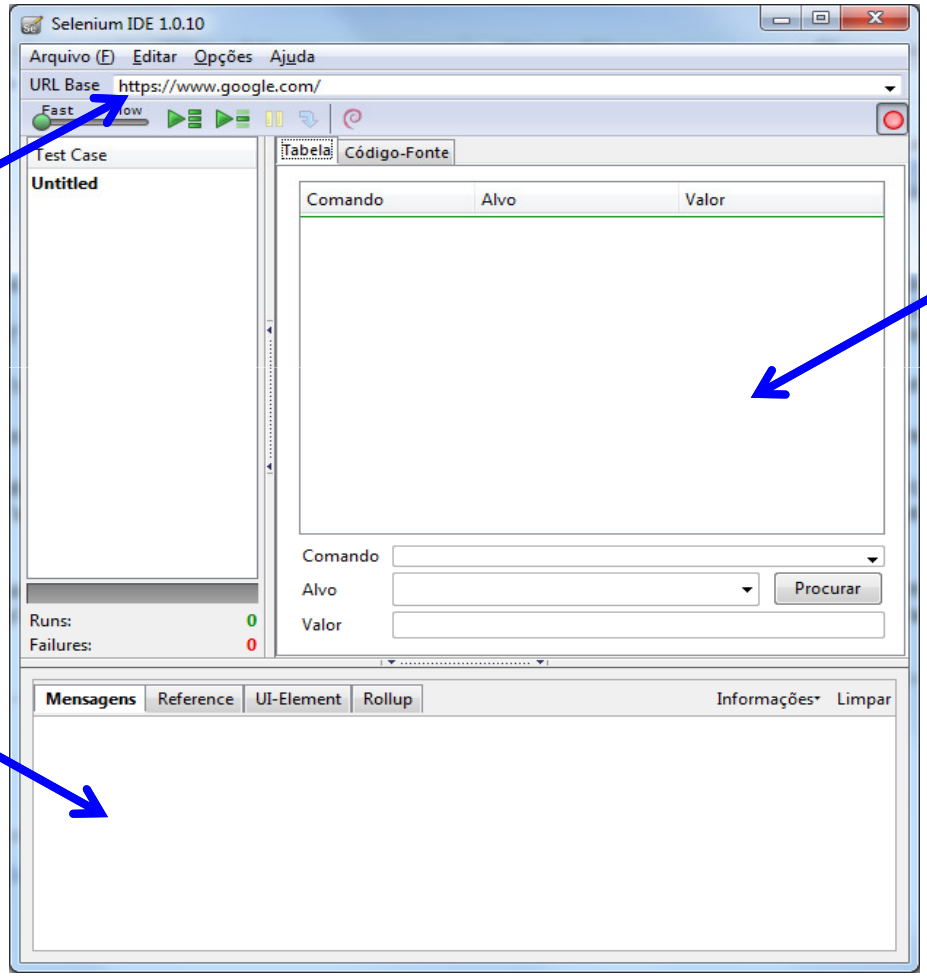
Grava as ações do usuário



Especificação dos comandos

Selenium IDE

Endereço do sistema que será testado



Lista das ações executadas

Registro dos eventos executados, incluindo eventuais erros

9

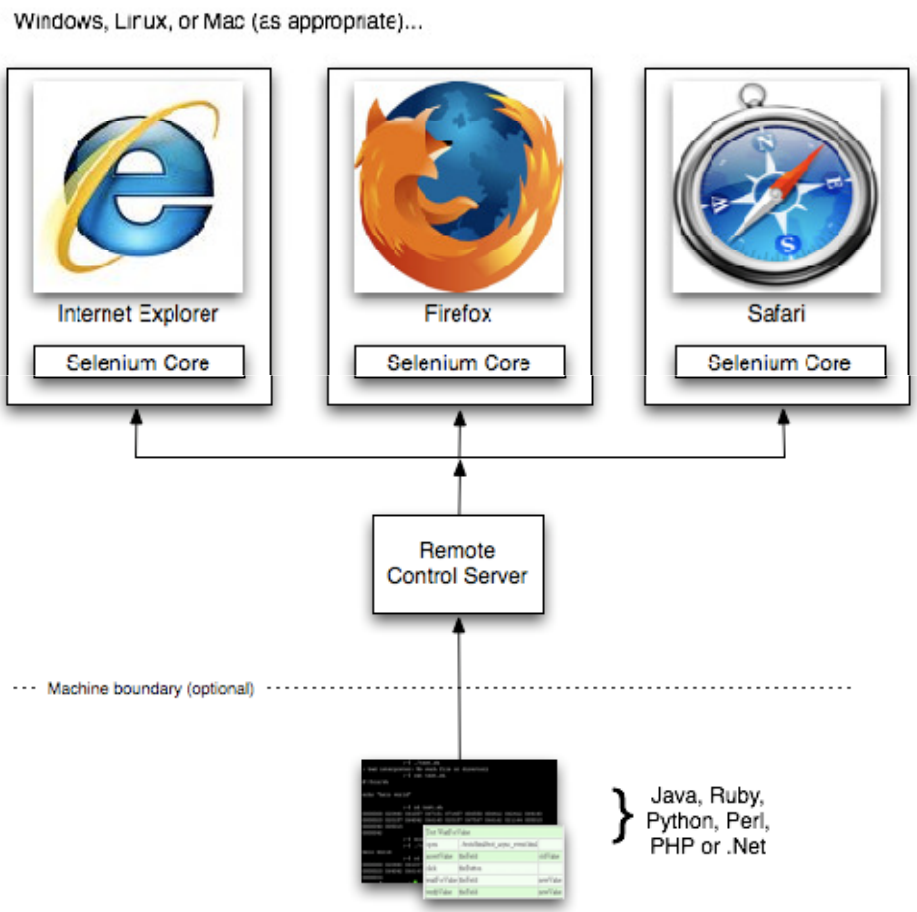
Selenium RC

Selenium-RC

- Selenium-RC oferece uma API (Application Programming Interface) e uma biblioteca para cada uma das linguagens suportadas: HTML, Java, C #, Perl, PHP, Python e Ruby.

- Possui um servidor que automaticamente abre e fecha browsers, e atua como um HTTP proxy para as requisições web.

Selenium-RC


Windows, Linux, or Mac (as appropriate)...
Internet Explorer
Selenium Core
Firefox
Selenium Core
Safari
Selenium Core
Remote Control Server
Machine boundary (optional)
Java, Ruby, Python, Perl, PHP or .Net

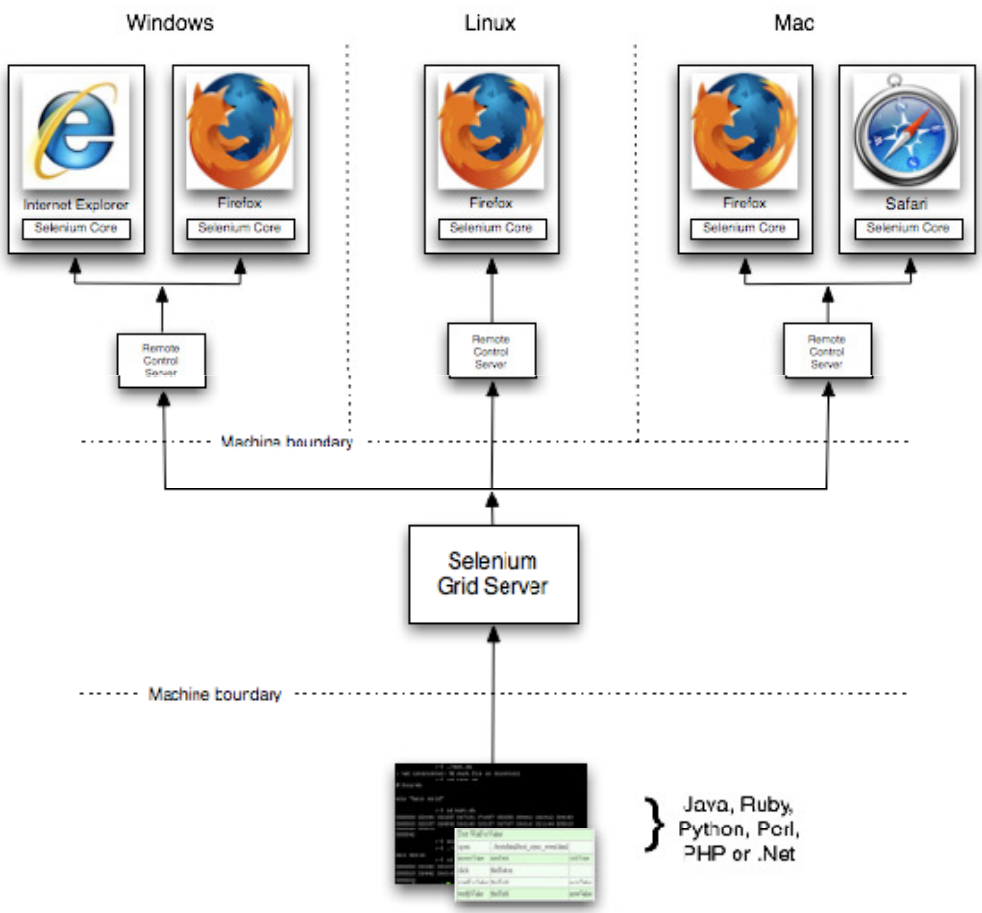
12

Selenium GRID

Selenium-Grid

- Esse componente é capaz de distribuir os testes em diversas máquinas para que possam ser executados em paralelo
  - Tempo de execução reduzido
  - Testes podem ser executados em diferentes combinações de navegador + sistema operacional
  - Gera um relatório agrupado de todas as máquinas de teste

Selenium-Grid



15

Comandos

Comandos

- *Actions:* são comandos que geralmente manipulam o estado da aplicação. Por exemplo, clicar em um link ou selecionar uma opção de uma comboBox.
  - Normalmente, os comandos que executam ações podem ter um sufixo "*AndWait*", por exemplo, "*clickAndWait*"
- *Accessors:* examinam o estado da aplicação e armazenam os resultados em variáveis, por exemplo, "storeTitle".
  - também são usados para gerar Assertions automaticamente.
- *Assertions:* são como Acessors, mas verificam se o estado da aplicação está em conformidade com o que se espera. Por exemplo, verifica se o título da página é X, verifica se uma determinada opção está assinalada.
- Todas as *Assertions* podem ser utilizadas em três modos: “assert", "verify" e "waitFor". Por exemplo, você pode utilizar "assertText", "verifyText“ e "waitForText".

Alguns comandos mais comuns

- open
  - Abre uma página URL.
- click/clickAndWait
  - Executa um clique e, opcionalmente, espera por uma nova página carregar.
- verifyTitle/assertTitle
  - Verifica o título da página.
- verifyTextPresent
  - Verifica se o texto esperado está na página.
- verifyElementPresent
  - Verifica se um elemento UI esperado, conforme definido pelo seu código HTML, está presente na página.
- verifyText
  - verifica se o texto esperado e sua tag HTML correspondente estão presentes na página.
- verifyTable
  - verifica o conteúdo esperado de uma tabela.
- waitForPageToLoad
  - interrompe a execução até que uma nova página seja carregada. Chamado automaticamente quando clickAndWait é usado.
- waitForElementPresent
  - interrompe a execução até que um elemento da interface do usuário esperado, conforme definido pelo seu código HTML, esteja presente na página.

Porque usar a Selenium?

Porque escolher a Selenium?

- As bibliotecas Selenium podem ser importadas nas mais populares IDEs
  - Eclipse, NetBeans, Visual Studio Net, etc.
  - Reutiliza os *frameworks* de testes de unidade
- Utilizada por equipes de teste funcional, teste de regressão e teste de aceitação do usuário
- A integração no processo de *build* tornou fácil de implementá-la em ambientes de Integração Contínua

Porque escolher a Selenium?

- Um forte apoio para as tecnologias AJAX
- Consegue realizar testes em diversos browsers e em diversos sistemas operacionais
- Executa testes em paralelo
  - Usando o componente Grid, os testes podem ser executados em paralelo em vários RC (agente) de diferentes máquinas
  - Economiza tempo e é mais rápido
- Selenium é GRÁTIS
  - Sem custos de licenciamento.
  - Pode ser implantada em quantas máquinas forem necessárias

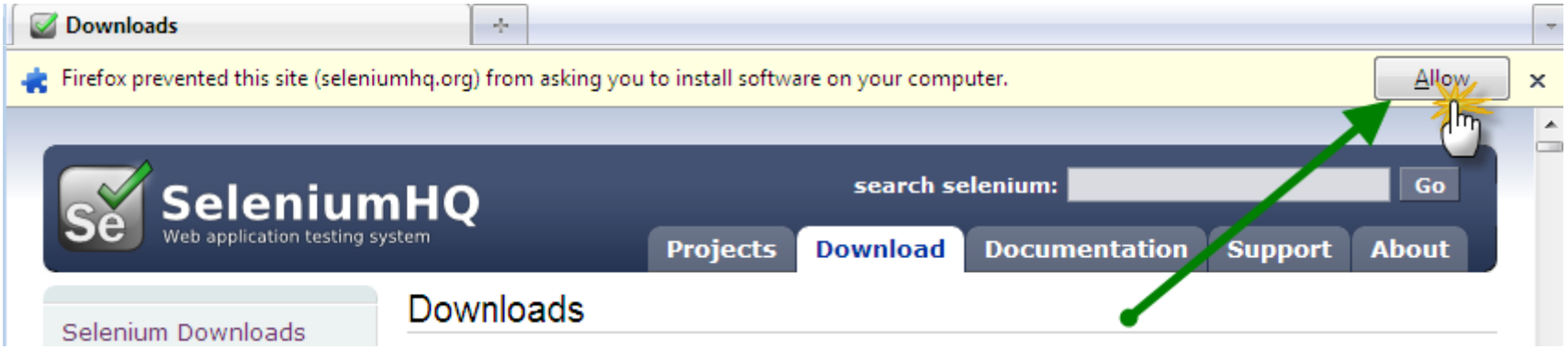
Instalando

Instalando Selenium IDE

- Abra a janela do navegador Firefox (obrigatoriamente)
- Abra http://seleniumhq.org/download/
- Em Downloads Escolha IDE Selenium

Instalando Selenium IDE

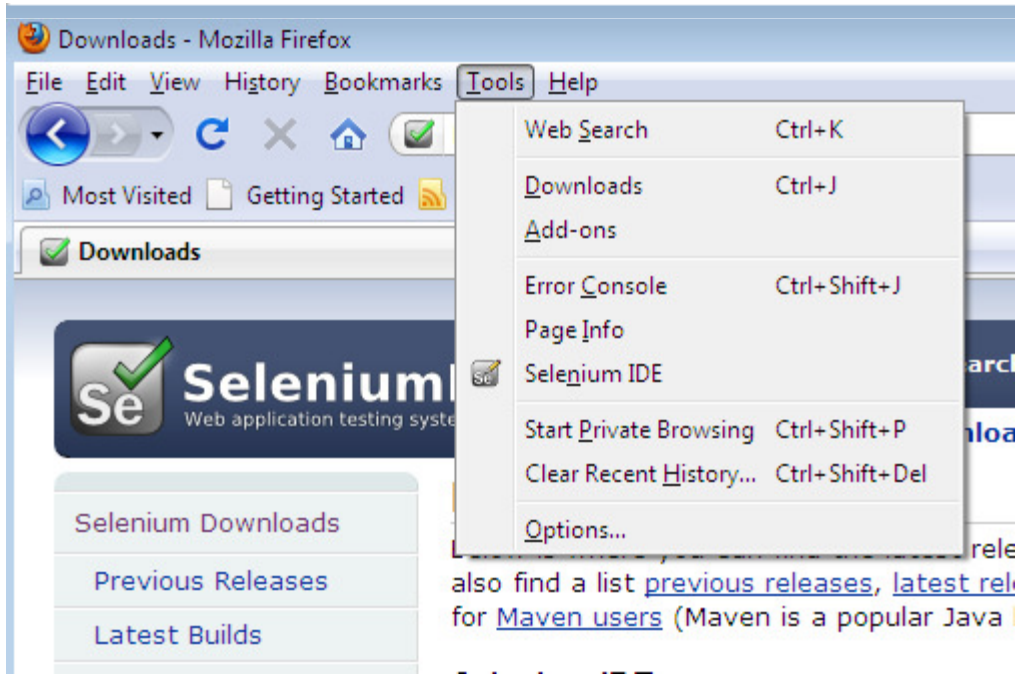
- Clique em *Allow* (Aceitar) para prosseguir com a instalação



- Reinicie o Firefox

Instalando Selenium IDE

- Agora, no Firefox, vá em Ferramentas e verifique se “Selenium IDE” é exibido
- Se sim, o Selenium IDE está instalado

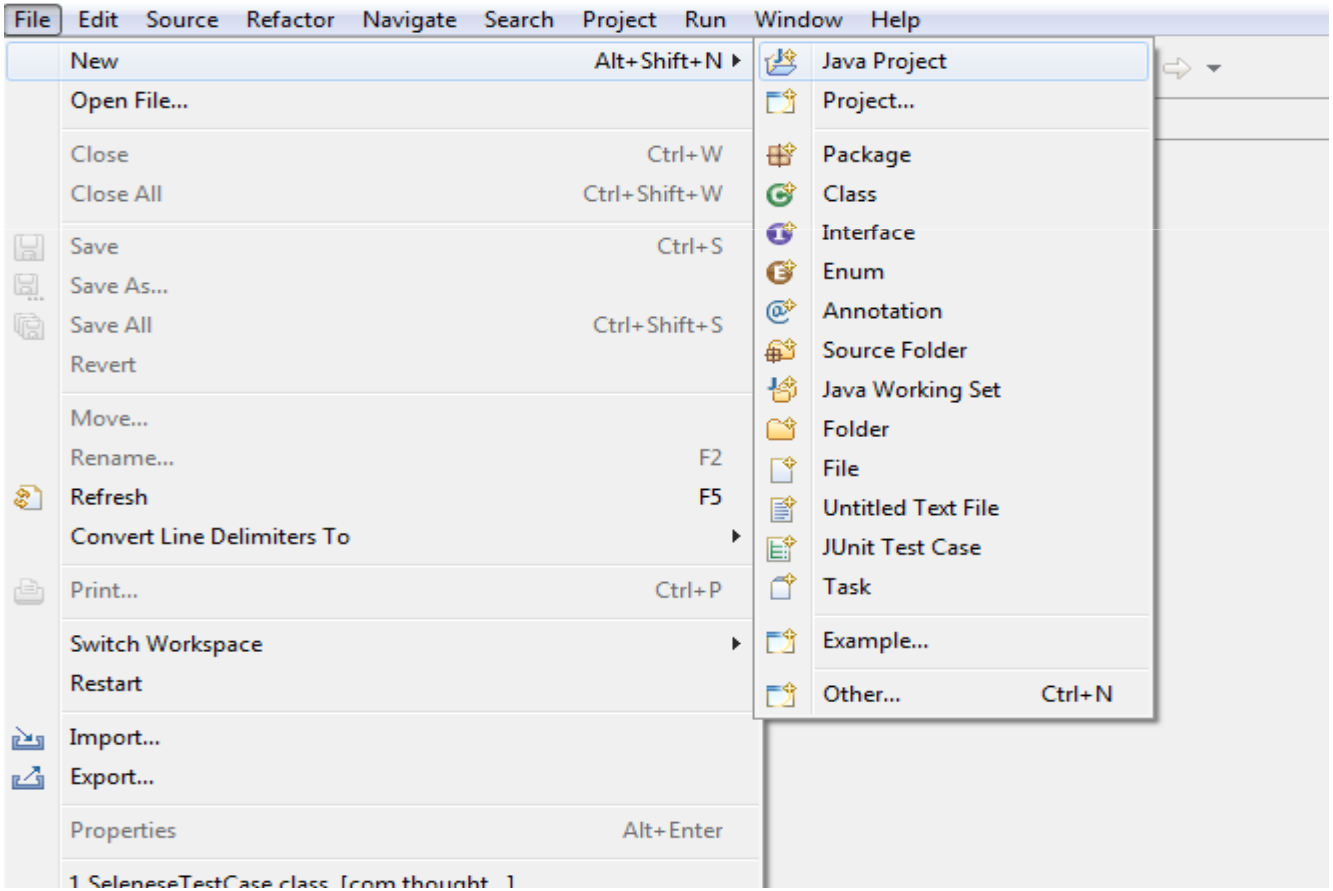


Instalando Selenium RC

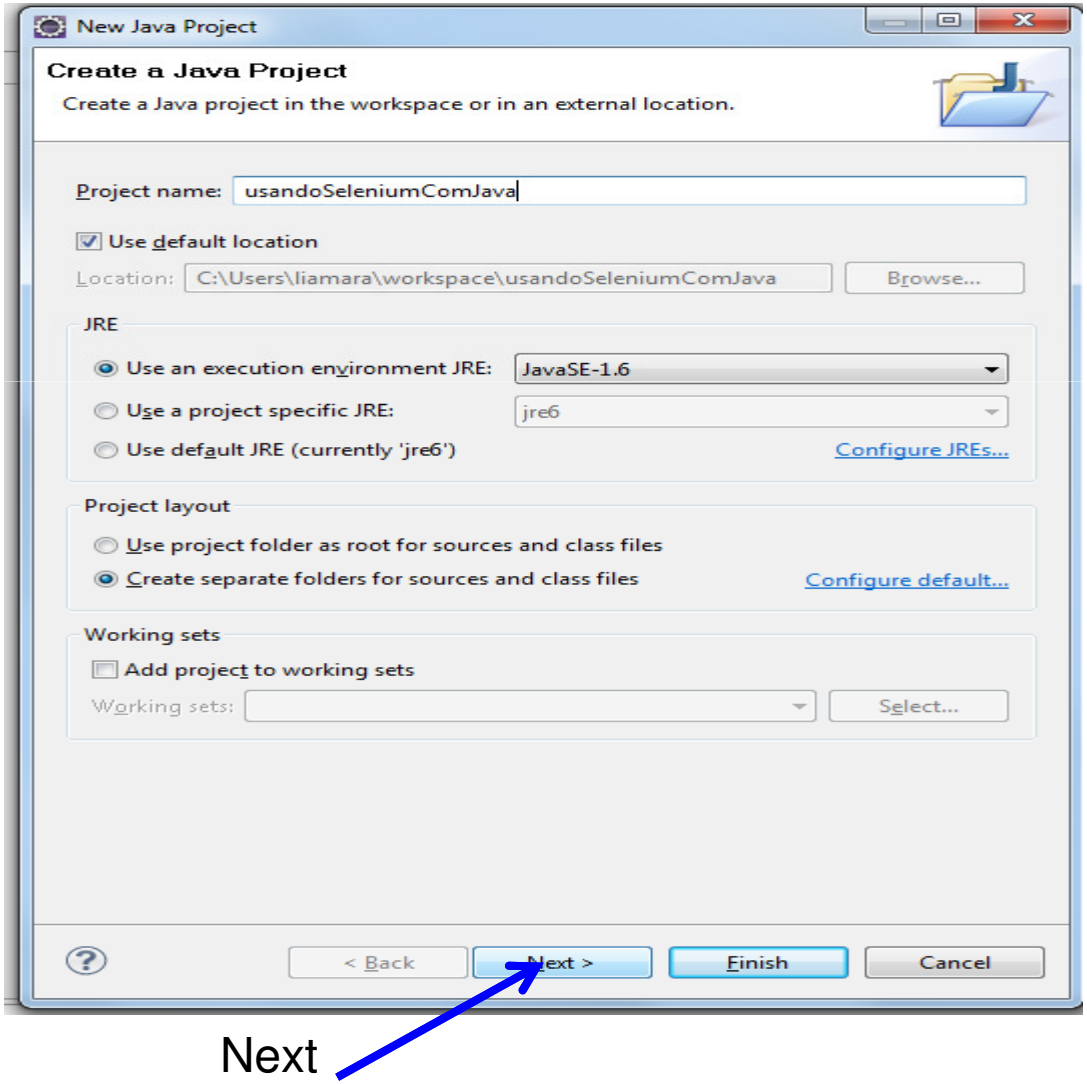
- Esse exemplo, será feito utilizando a linguagem Java
- A IDE utilizada será o Eclipse
- Para executar os testes, será usado o JUnit 4

Criação do projeto no Eclipse

- Criar um novo projeto



Criação do projeto no Eclipse



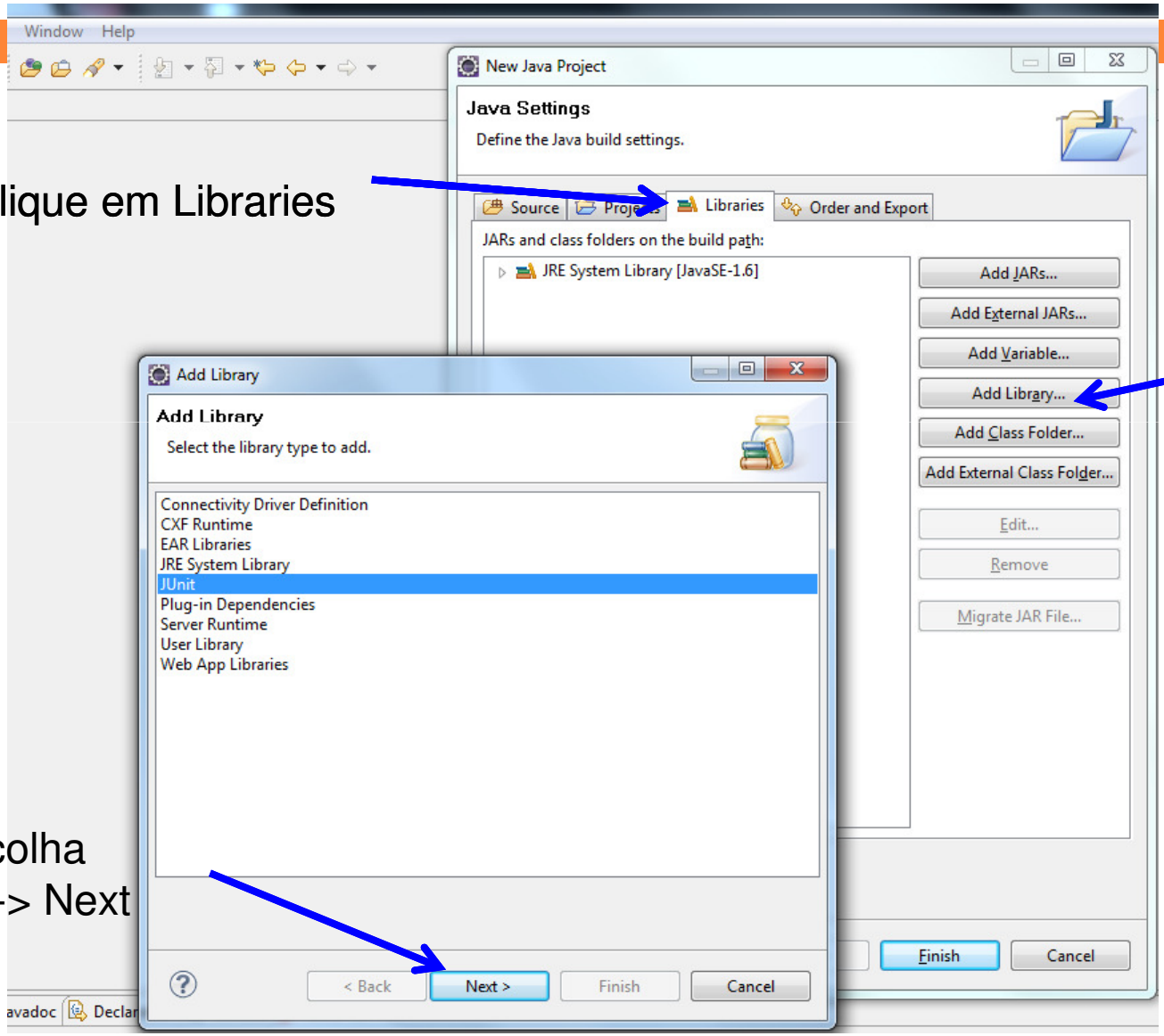
Next

Criação do projeto no Eclipse

- 1º Clique em Libraries

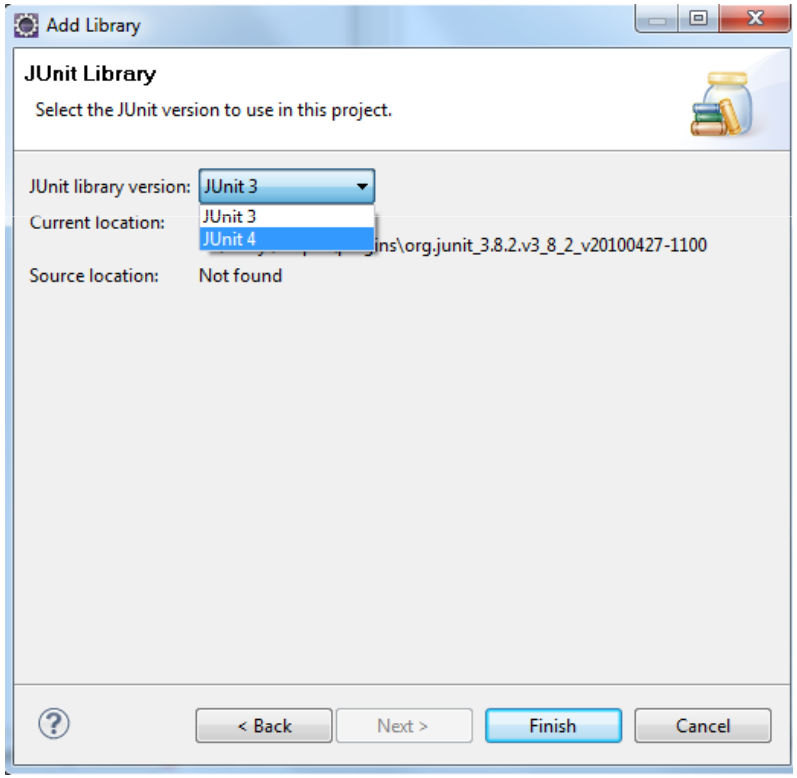
2º Clique em Add Library

3º Escolha JUnit -> Next

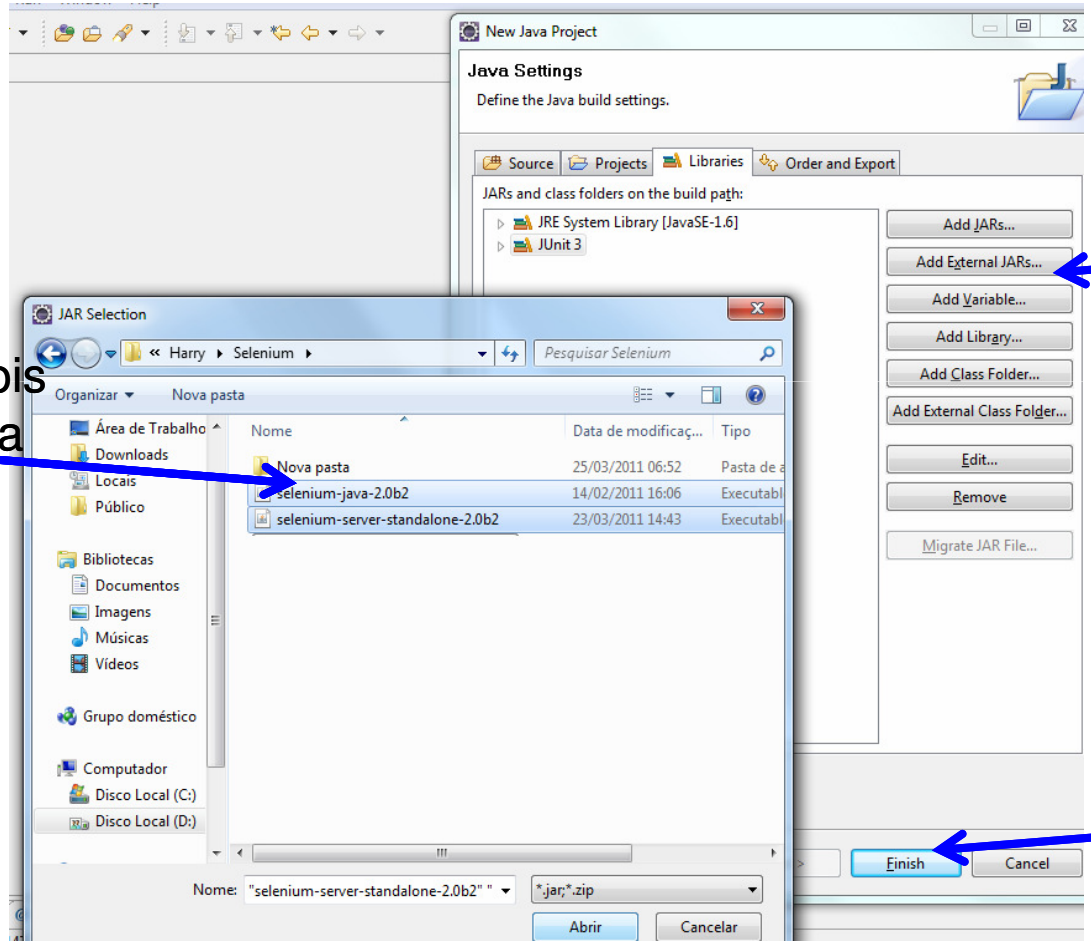


Criação do projeto no Eclipse

- Escolha JUnit 4 e clique em Finish



Criação do projeto no Eclipse



1º Clique em Add External Jars

2º Adicione os dois jars mostrados na Imagem

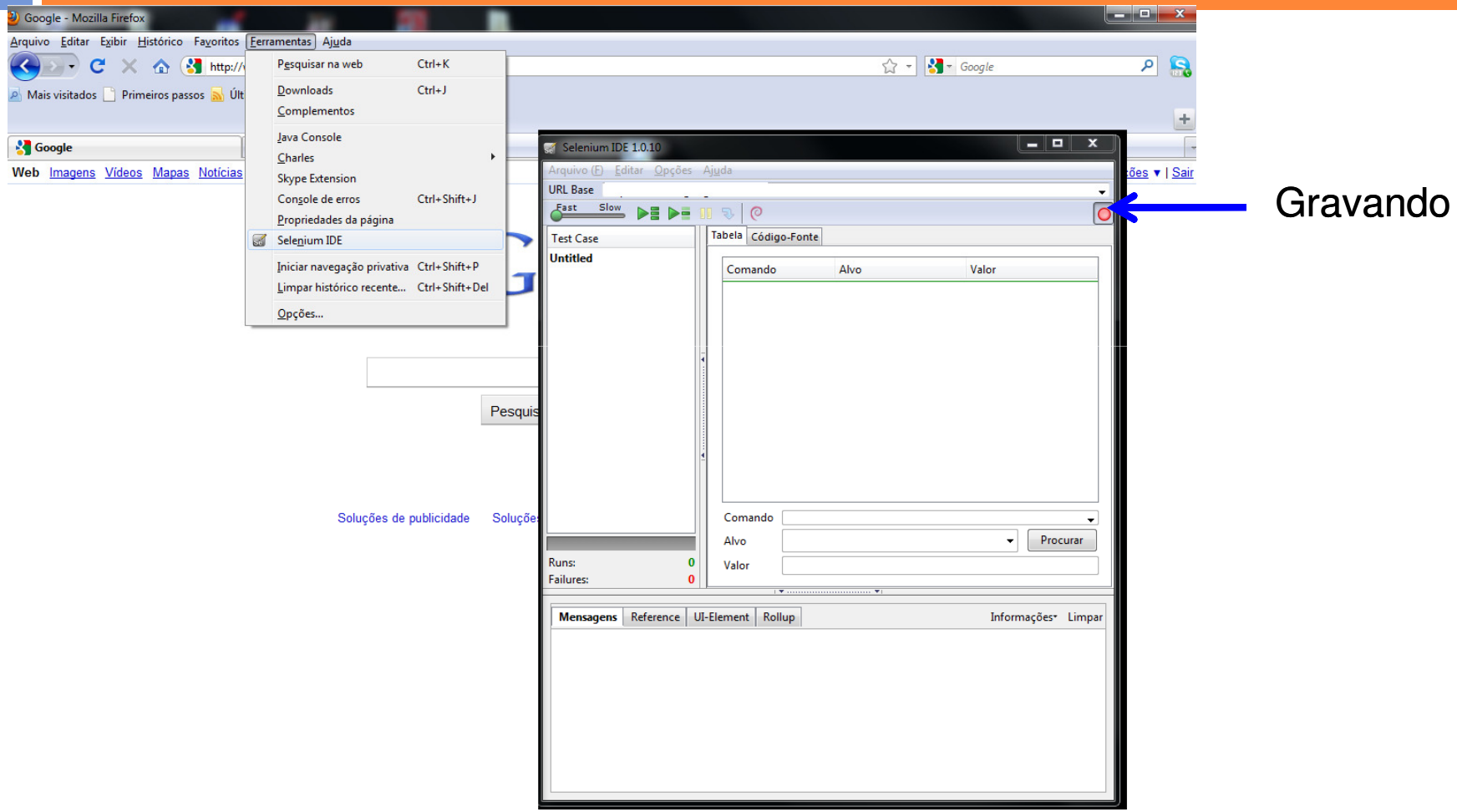
3º Clique em Finish

http://seleniumhq.org/download/

Ambiente pronto

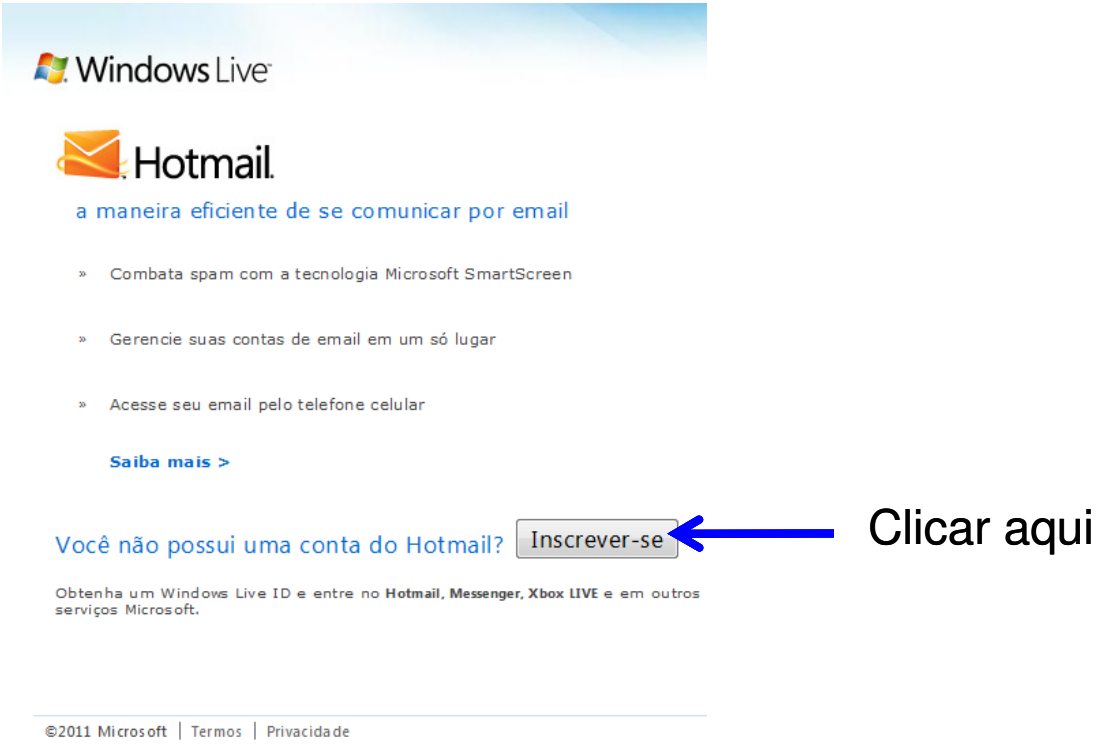
- Agora, já é possível realizar testes utilizando a ferramenta Selenium em projetos Java

Exemplo – Gravando um teste



Gravando utilização

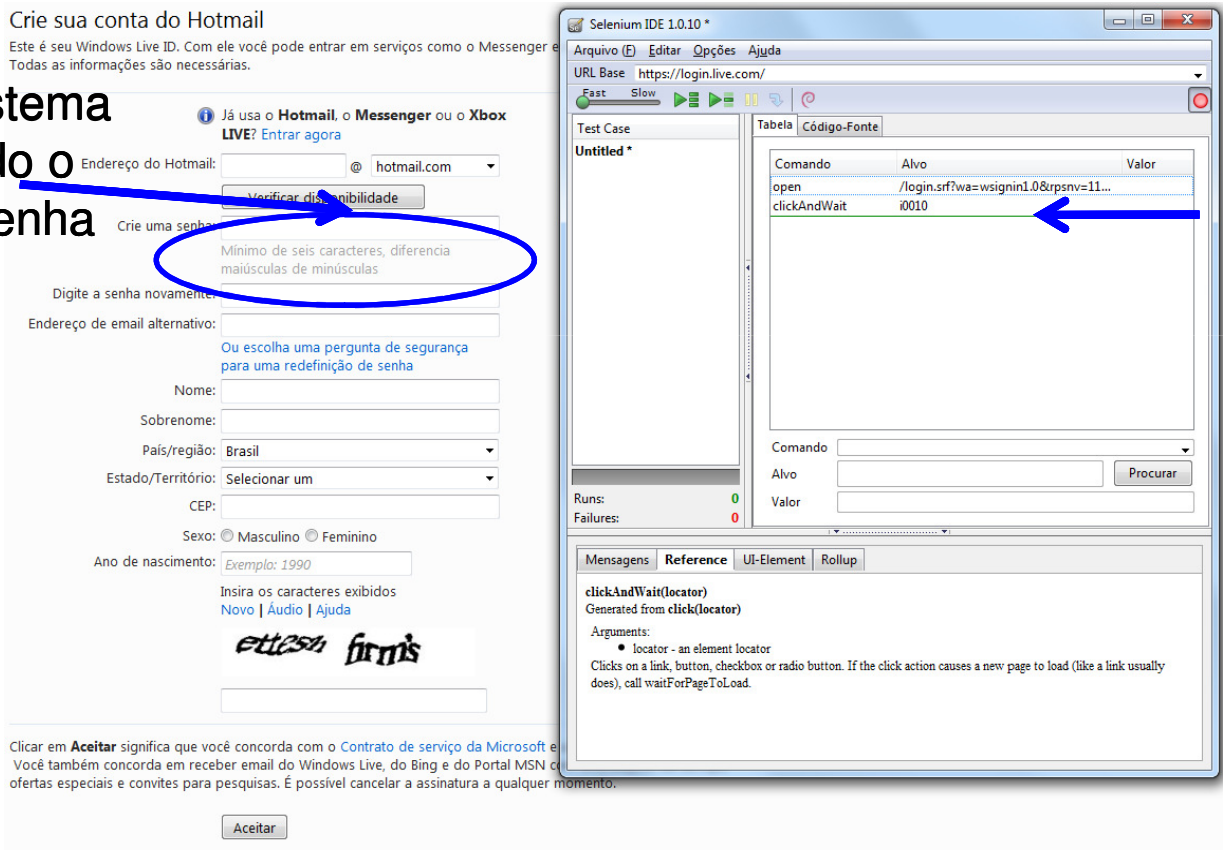
- Site escolhido: www.hotmail.com



Clicar aqui

Gravando utilização

Testar se o sistema está verificando o tamanho da senha



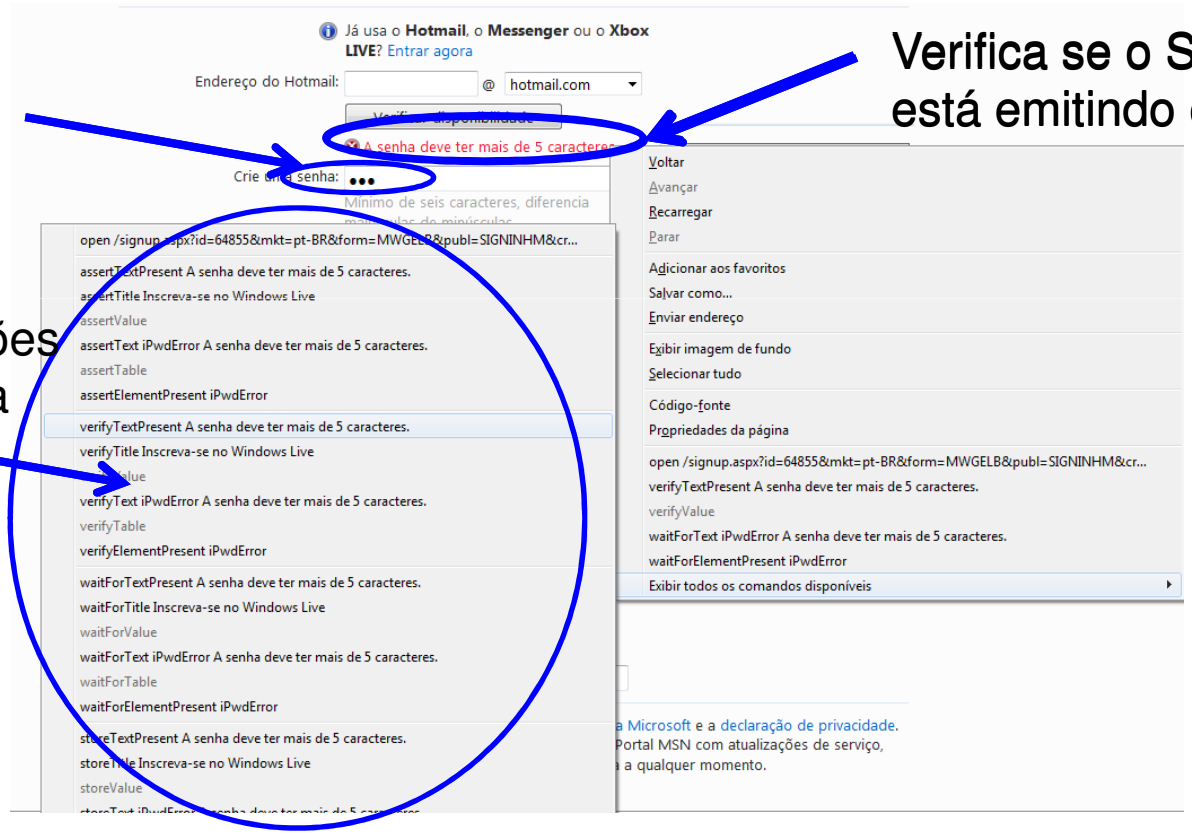
Passos já gravados

Gravando utilização

Digitado: 123

Verifica se o Sistema está emitindo o alerta

Comandos / Verificações disponíveis para a tela



Visualizando comandos

Selenium IDE 1.0.10 *

Arquivo (F) Editar Opções Ajuda

URL Base https://login.live.com/

Fast Slow

Test Case

Untitled *

Tabela Código-Fonte

Comando	Alvo	Valor
open	/login.srf?wa=wsignin1.0&rpsnv=11...	
clickAndWait	i0010	
type	iPwd	123
verifyTextPresent	A senha deve ter mais de 5 caracteres.	

Comando

Alvo

Valor

Procurar

Runs: 0

Failures: 0

Mensagens Reference UI-Element Rollup

clickAndWait(locator)

Generated from **click(locator)**

Arguments:

- locator - an element locator

Clicks on a link, button, checkbox or radio button. If the click action causes a new page to load (like a link usually does), call waitForPageToLoad.

Comandos capturados

Selenium IDE 1.0.10 *

Arquivo (F) Editar Opções Ajuda

Opções...

Formato

Formato da Área de Transferência

HTML

Ruby (Test::Unit)

RSpec

JUnit 3 (Remote Control)

JUnit 4 (Remote Control)

TestNG (Remote Control)

Groovy (JUnit)

Perl

PHP

Python 2 (Remote Control)

C# (Remote Control)

```
package com.example.tests;

import com.thoughtworks.selenium
import org.junit.After;
import org.junit.Before;
import org.junit.Test;
import java.util.regex.Pattern;

public class Untitled extends Se
    @Before
    public void setUp() throws Ex
        selenium = new Default
        selenium.start();
    }

    @Test
    public void testUntitled() throws Exception {
        selenium.open("/login.srf?wa=wsignin1.0&rpsnv=11&ct=1301047575&rver=6.1.6206.0&wp=MBI&wreply=
        selenium.click("i0010");
        selenium.waitForPageToLoad("30000");
        selenium.type("iPwd", "123");
        verifyTrue(selenium.isTextPresent("A senha deve ter mais de 5 caracteres."));
    }

    @After
    public void tearDown() throws Exception {
        selenium.stop();
    }
}
```

tps://login.live.com/");

Runs: 0

Failures: 0

Mensagens Reference UI-Element Rollup

clickAndWait(locator)

Generated from **click(locator)**

Código para ser utilizado no JUnit

Criando Classe de Teste

- O código gerado pela Selenium IDE pode ser exportado clicando em Arquivo -> Exportar Teste Como -> JUnit 4

  ou ...

  copiando diratamente o código mostrado e colando dentro de uma classe no Eclipse

Criando Classe de Teste

Java - usandoSeleniumComJava/src/com/example/tests/ClasseDeTeste.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

- testeSelenium
  - src
    - com.selenium.testes
      - teste.java
      - Untitled.java
  - JRE System Library [JavaSE-1.6
  - JUnit 4
  - Referenced Libraries
- usandoSeleniumComJava
  - src
    - com.example.tests
      - ClasseDeTeste.java
  - JRE System Library [JavaSE-1.6
  - JUnit 3
  - Referenced Libraries

ClasseDeTeste.java | teste.java

```
package com.example.tests;

import com.thoughtworks.selenium.*;

public class ClasseDeTeste extends SeleneseTestCase {
    @Before
    public void setUp() throws Exception {
        selenium = new DefaultSelenium("localhost", 4444, "*chrome", "https://login.l
        selenium.start();
    }

    @Test
    public void testUntitled() throws Exception {
        selenium.open("/login.srf?wa=wsignin1.0&rpsnv=11&ct=1301047575&rver=6.1.6206.
        selenium.click("i0010");
        selenium.waitForPageToLoad("30000");
        selenium.type("iPwd", "123");
        verifyTrue(selenium.isTextPresent("A senha deve ter mais de 5 caracteres."));
    }

    @After
    public void tearDown() throws Exception {
        selenium.stop();
    }
}
```

Problems | Javadoc | Declaration | Console

<terminated> teste [JUnit] C:\Program Files\Java\jre6\bin\javaw.exe (24/03/2011 16:13:27)

Writable | Smart Insert | 28 : 6

Servidor

- Para executar os casos de teste, é necessário inicializar o servidor Selenium. É possível fazer isso de duas maneiras:
  - Via prompt, ir até a pasta onde se encontra o .jar do servidor e iniciá-lo com o comando:

    ```
    java -jar selenium-server-standalone-2.0b2.jar
    ```

  - Incluir trechos de códigos na classe de teste

    ```
    SeleniumServer servidor;
    servidor = new SeleniumServer();
    servidor.start();
    ```

Alteração no código

- É importante ressaltar que, muita das vezes, o código gerado pela Selenium IDE precisa de **ajustes** do testador

```
public class ClasseDeTeste extends SeleneseTestCase {
Selenium selenium;
SeleniumServer servidor;

@Before
public void setUp() throws Exception {
    servidor = new SeleniumServer();
    servidor.start();
    selenium = new DefaultSelenium("localhost", 4444, "*chrome", "https://login.live.com/");
    selenium.start();
}

@Test
public void testTeste() throws Exception {

selenium.open("/login.srf?wa=wsignin1.0&rpsnv=11&ct=1300939240&rver=6.1.6206.0&wp=MBI&wreply=http:%2F%2Fmail.live.com%2Fdefault.aspx&lc=1046&id=64855&mk
t=pt-br&cbcxt=mai&snsc=1");
    Thread.sleep(3000);
    selenium.click("i0010");
    selenium.waitForPageToLoad("30000");
    selenium.type("iPwd", "123");
    assertEquals(selenium.isTextPresent("A senha deve ter mais de 5 caracteres."), true);
}

@Test
public void testTeste2() throws Exception {
selenium.open("/login.srf?wa=wsignin1.0&rpsnv=11&ct=1300939240&rver=6.1.6206.0&wp=MBI&wreply=http:%2F%2Fmail.live.com%2Fdefault.aspx&lc=1046&id=64855&mk
t=pt-br&cbcxt=mai&snsc=1");
    Thread.sleep(3000);
    selenium.click("i0010");
    selenium.waitForPageToLoad("30000");
    selenium.type("iPwd", "1234");
    assertEquals(selenium.isTextPresent("A senha deve ter mais de 5 caracteres."), true);
}

@Test
public void testTeste3() throws Exception {
selenium.open("/login.srf?wa=wsignin1.0&rpsnv=11&ct=1300939240&rver=6.1.6206.0&wp=MBI&wreply=http:%2F%2Fmail.live.com%2Fdefault.aspx&lc=1046&id=64855&mk
t=pt-br&cbcxt=mai&snsc=1");
    Thread.sleep(3000);
    selenium.click("i0010");
    selenium.waitForPageToLoad("30000");
    selenium.type("iPwd", "12345");
    assertEquals(selenium.isTextPresent("A senha deve ter mais de 5 caracteres."), true);
}

@Test
public void testTeste4() throws Exception {
selenium.open("/login.srf?wa=wsignin1.0&rpsnv=11&ct=1300939240&rver=6.1.6206.0&wp=MBI&wreply=http:%2F%2Fmail.live.com%2Fdefault.aspx&lc=1046&id=64855&mk
t=pt-br&cbcxt=mai&snsc=1");
    Thread.sleep(3000);
    selenium.click("i0010");
    selenium.waitForPageToLoad("30000");
    selenium.type("iPwd", "123456");
    assertEquals(selenium.isTextPresent("A senha deve ter mais de 5 caracteres."), true);
}

@After
public void tearDown() throws Exception {
    selenium.stop();
    servidor.stop();
}
}
```

Servidor

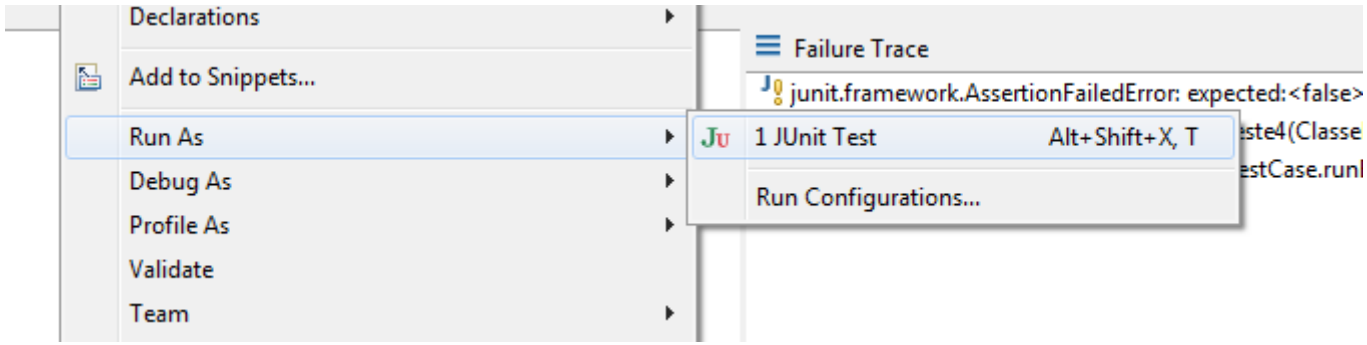
Necessário para que a página seja carregada completamente

Verifica se o texto foi encontrado na página

Esse teste deve falhar, pois será digitado 6 caracteres para a senha e, no comando abaixo, espera-se que seja emitido a mensagem de erro (não será).

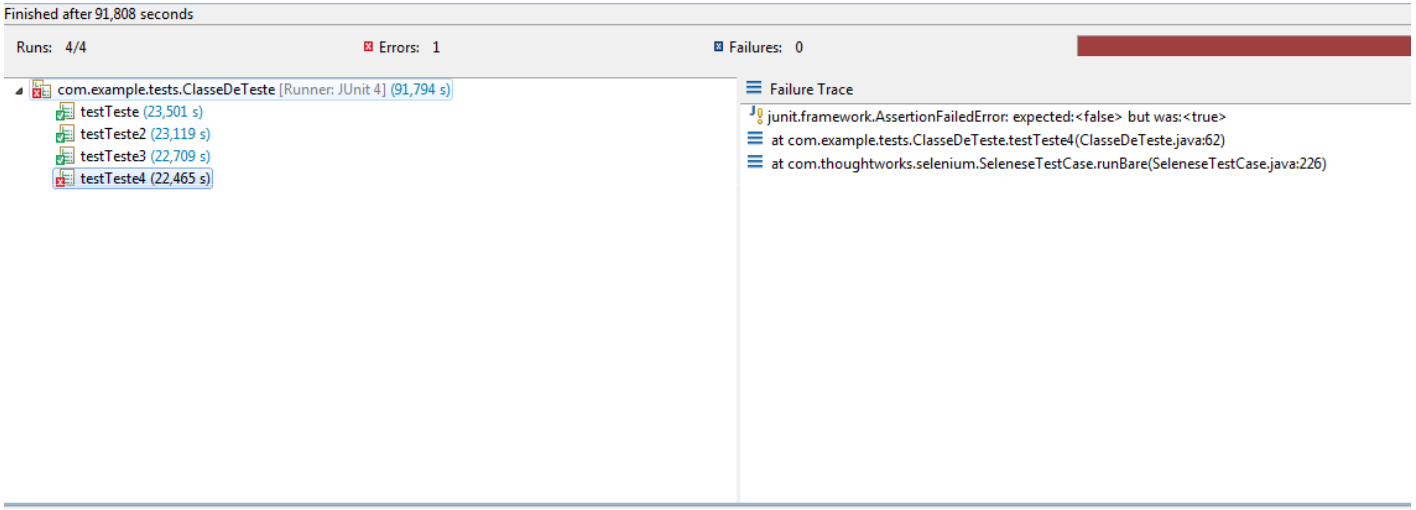
Executando os Testes

- Clicar com o Botão direito -> Run As -> JUnit Teste
  - Todos os Testes da classe serão executados



Executando os Testes

- Como esperado, o último teste falhou



Exercício