Documentación de integración de pruebas Web con Selenium

Contents

[Pasos para ejecutar pruebas con Selenium en una aplicación nueva 2](#_Toc342660305)

[Supuestos 2](#_Toc342660306)

[Requisitos 2](#_Toc342660307)

[Pasos 2](#_Toc342660308)

[Correr pruebas de Selenium usando archivos HTML 3](#_Toc342660309)

[Extender la clase DextraTestBase 5](#_Toc342660310)

[Errores comunes 5](#_Toc342660311)

[El servidor Selenium está corriendo 5](#_Toc342660312)

# Pasos para ejecutar pruebas con Selenium en una aplicación nueva

## Supuestos

* La aplicación a probar se encuentra en ejecución en un servidor de aplicaciones
* La aplicación es accesible desde otras IP, si no se probará la aplicación en e localhost.

## Requisitos

* Java 1.5 o superior, preferentemente el 6
* Tomcat 5.5 o superior, preferentemente el 6
* Maven 2.2.x o superior, preferentemente el 3.x.x
* Un cliente de SVN, preferentemente si es un plugin para el IDE, como Subclipse para Eclipse.

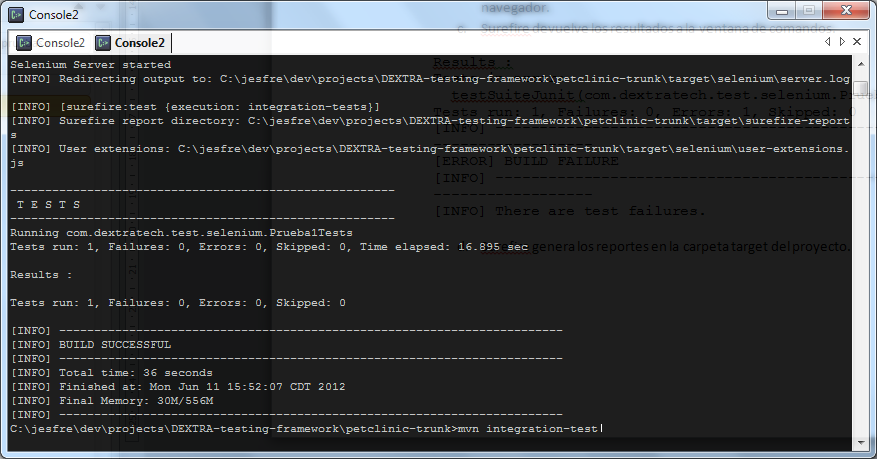
## Pasos

### Creación del proyecto de pruebas

1. Descargar el arquetipo Maven del repositorio :pserver:[user]@192.168.10.150:/opt/cvshome/TESTING/dtf-project-archetype/target/generated-sources/archetype/target/dtf-project-archetype-0.0.1-SNAPSHOT.jar o de alguna de las revisiones liberadas.
2. Instalar el arquetipo en el repositorio local de Maven. Esto se ejecuta fácilmente corriendo el archivo por lotes **create-archetype.bat**.
3. Copiar el archivo por lotes **generate-project.bat** a una nueva ubicación, preferentemente donde se quiera generar el nuevo proyecto de pruebas.
4. Editar el archivo **generate-project.bat** cambiando el parámetro artifactId por el nombre deseado para el proyecto de ptuebas.
5. Correr **generate-project.bat**. Esto generará un proyecto de pruebas. Este contiene carpetas donde serán generados casos y datos de pruebas, además de recursos para generar reportes.

### Creación de pruebas y ejecución

1. Dentro del nuevo proyecto de pruebas.
2. Poner los script de pruebas HTML en la carpeta <PROYECTO\_PRUEBAS>/src/test/resources/base-layouts/. Estos script pudieron ser generados desde el Selenium IDE o manualmente. Solo serán usados para generar plantillas.
3. Modificar el archivo de configuración en <PROYECTO\_PRUEBAS>/dtf-project/src/test/resources/config/globalConfiguration.xml con los datos necesarios.
4. Ejecutar el comando de Maven**>mvn dtf:build-layout**.
5. Esto habrá generado <PROYECTO\_PRUEBAS>/src/test/resources/generated-datasets/[casos\_de\_prueba.xls] y <PROYECTO\_PRUEBAS>/src/test/resources/generated-layouts/[plantilla\_para\_cada\_caso\_de\_prueba.html].
6. Agregar los datos a probar para cada batería de pruebas. Hay un archivo Excel por cada HTML encontrado en base-layouts. Cada archivo de Excel contiene una batería de pruebas, cada renglón representa un escenario de pruebas.
7. Para correr las pruebas, ejecutar el comando **>mvn clean integration-test**. Este correrá cada prueba de acuerdo a las configuraciones establecidas en globalConfiguration.xml.
   1. Ejecutará los comandos >mvn dtf:generate-tests y >mvn dtf:html2java para generar los casos de prueba en base a la plantilla HTML y conjunto de datos generados en el paso anterior
   2. Levanta el servidor Selenium
   3. Abre una ventana del navegador (Firefox) y ejecuta las pruebas de Selenium en el navegador.
   4. Surefire devuelve los resultados a la ventana de comandos.



### Creación de pruebas y ejecución

1. Para generar los reportes, correr **>mvn post-site.**
2. Este comando abrirá el reporte en el navegador especificado por la propiedad de Maven: report.browser.
3. El reporte se encontrará en <PROYECTO\_PRUEBAS>/target/site/surefire-report.html.

# Correr pruebas de Selenium usando archivos HTML

Dadas las configuraciones del primer documento. Se siguen los siguientes pasos:

1. Agregar la siguiente ejecución al pluginselenium-maven-plugin

<properties>

…

<selenium.browser>\*firefox</selenium.browser>

<selenium.suite>src/test/selenium/testSuite1.html</selenium.suite>

<selenium.startURL>http://google.com.mx</selenium.startURL>

</properties>

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>selenium-maven-plugin</artifactId>

<version>2.3</version>

**<configuration>**

**<browser>${selenium.browser}</browser>**

**<suite>${selenium.suite}</suite>**

**<startURL>${selenium.startURL}</startURL>**

**</configuration>**

<executions>

<execution>

<id>start</id>

<phase>pre-integration-test</phase>

<goals>

<goal>start-server</goal>

</goals>

<configuration>

**<skip>true</skip>**

<background>true</background>

<logOutput>true</logOutput>

<multiWindow>true</multiWindow>

</configuration>

</execution>

<execution>

<id>stop</id>

<phase>post-integration-test</phase>

<goals>

<goal>stop-server</goal>

</goals>

<configuration>

<skip>true</skip>

</configuration>

</execution>

**<execution>**

**<id>test-html</id>**

**<phase>integration-test</phase>**

**<goals>**

**<goal>selenese</goal>**

**</goals>**

**<configuration>**

**<skip>false</skip>**

**</configuration>**

**</execution>**

</executions>

</plugin>

**Esta nueva ejecución espera encontrar el archivo de suite de pruebas definido en src/test/selenium/testSuite1.html.**

1. Correr**>mvn integration-test** o **>mvnselenium:selenese**. El primero causa el problema mencionado en la sección de Errores comunes.

# Extender la clase DextraTestBase

Actualmente, el IDE de Selenium genera las clases extendiendo de la clase SeleneseTestCase, pero esta clase está deprecada. En lugar de extender de esa clase se sugiere extenderé de SeleneseTestBase, esto es para los casos de prueba solamente, ya que **para los suites de pruebas es necesario que extienda de DextraTestBase**.

Además de extender de la clase DextraTestBase, lo que hice fue extender (mejorar) la implementación construyendo una superclase para las pruebas que extiende de esa clase, y agregando las implementaciones de mejora. Desde ahí se pueden agregar más mejoras.

# Errores comunes

## El servidor Selenium está corriendo

**Causa**

Este error ocurrió porque se estaba ejecutando la tarea definida en la ejecución start-server porque se está ejecutando directamente la fase **integration-test**. Para evitar que la tarea inicie el servidor Selenium, basta con agregar la configuración **<skip>true</skip>**a la ejecución.

**Solución**

Entrar a la dirección: <http://localhost:4444/selenium-server/driver/?cmd=shutDownSeleniumServer>

La respuesta debe ser **OKOK**, en caso de que el servidor realmente no esté levantado entonces marcará que no se puede conectar.

## No puede arrancar una sesión del navegador

Cuando intenta ejecutar las pruebas aparece el siguiente error:

Could not start Selenium session: Failed to start new browser session: java.lang.RuntimeException: Firefox 3 could not be found in the path!

Please add the directory containing ''firefox.exe'' to your PATH environment

variable, or explicitly specify a path to Firefox 3 like this:

\*firefox3c:\blah\firefox.exe

**Causa**

Esto es debido a que no el driver de Selenium no conoce la ruta del .exe del navegador en la maquina local. Por lo tanto no puede ejecutar el navegador.

**Solución**

En la configuración del navegador a usar (globalConfiguration.xml), en blowser se deberá poner la ruta del navegador. Por ejemplo, para Chrome en Windows:

\*googlechrome C:\\Program Files (x86)\\Google\\Chrome\\Application\\chrome.exe

## Error de compilación 1

Un error de compilación al momento de intentar correr el comando **integration-test** puede ser el siguiente:

Compilation failure: Compilation failure:

[ERROR]ruta\a\Scenario1.java:[47,29] unclosed string literal

**Causa**

Puede ser ocasionado por que alguna frase puede tener saltos de línea en el HTML, por ejemplo en un texto esperado en un alert, provocando que el archivo Java generado tenga ese salto de línea y no compile correctamente.

**Solución**

Asegurarse al generar la plantilla HTML para la prueba que no existen saltos de línea.

# Lista de navegadores soportados

\*firefox

\*chrome

\*iexploreproxy

\*googlechrome

\*mock

\*firefoxproxy

\*pifirefox

\*iexplore

\*firefox3

\*safariproxy

\*konqueror

\*firefox2

\*safari

\*piiexplore

\*firefoxchrome

\*opera

\*iehta

\*custom