

GUIA PRUEBA AUTOMATIZADA CON SELENIUM A UN LOGIN Y REGISTRO

Medellín, abril 2020

Por: Johana Catalina Ríos Torres
Ingeniería Sistema UdeA
Cod 41960845
Calidad de Software

HERRAMIENTAS:

- Eclipse JEE 2019-09 preferiblemente con JDK 11
- Google Versión 81.0.4044.122 (64 bits)
- GitHub
- Jenkins
- SonarQube → JDK 11

En esta guía, mostrare cómo usar Jenkins para ejecutar automáticamente las pruebas de Selenium escritas en formato JUnit, y cómo los resultados de estas pruebas se pueden informar directamente a Jenkins. Para lograr esto, debemos completar los siguientes pasos:

- Escribir algunas las pruebas de Selenium en formato JUnit que queremos ejecutar como son un Login y un registro.
- Cree un archivo de compilación que ejecute estas pruebas y escriba los informes en el disco
- Configure un trabajo de Jenkins que ejecute estas pruebas e interprete los resultados.

Creación de pruebas de Selenium para ejecutar

Primero, necesitamos tener algunas pruebas para ejecutar. He creado dos pruebas cortas en formato JUnit en eclipse:

Login a una patina web

```
package back.ing.procesos.app.automated.test;

import org.junit.Before;
import org.junit.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.junit.Assert;
```

```

public class LoginTest {

    private WebDriver driver;

    @Before
    public void setUp() {
        System.setProperty("webdriver.chrome.driver",
            "./src/test/resources/chromedriver2/chromedriver.exe");
        driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("http://localhost:8080");
    }

    @Test
    public void comprobarFlujoCorrectoTransferencia() {
        final String tituloProductos = "Listado de Productos";
        //Login
        WebElement username = driver.findElement(By.id("username"));
        username.sendKeys("admin");
        WebElement password = driver.findElement(By.id("password"));
        password.sendKeys("12345");
        WebElement btnSignIn = driver.findElement(By.name("signIn"));
        btnSignIn.click();

        //Assert
        WebElement titulo = driver.findElement(By.name("titulo"));
        Assert.assertEquals(tituloProductos, titulo.getText());
    }
}

```

Registrar un nuevo usuario:

```

package back.ing.procesos.app.automated.test;

import org.junit.Assert;
import org.junit.Before;
import org.junit.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class formUsuarioTest {

    private WebDriver driver;

    @Before
    public void setUp() {
        System.setProperty("webdriver.chrome.driver",
            "./src/test/resources/chromedriver2/chromedriver.exe");
        driver = new ChromeDriver();
    }
}

```

```

        driver.manage().window().maximize();
        driver.get("http://localhost:8080");
        //Login
        WebElement username = driver.findElement(By.id("username"));
        username.sendKeys("admin");
        WebElement password = driver.findElement(By.id("password"));
        password.sendKeys("12345");
        WebElement btnSignIn = driver.findElement(By.name("signIn"));
        btnSignIn.click();
    }

    @Test
    public void comprobarFlujoCorrectoTransferencia() {
        driver.get("http://localhost:8080/usuario/usuarioListar");
        // List
        int idMayor = 1;

        while (true) {
            int i = 0;
            try {
                WebElement idUsuario =
driver.findElement(By.name(String.valueOf(idMayor+i)));
                if (idUsuario != null) {
                    if (Integer.parseInt(idUsuario.getText()) >
idMayor) {
                        idMayor =
Integer.parseInt(idUsuario.getText());
                        i++;
                    } else if
(Integer.parseInt(idUsuario.getText()) == idMayor) {
                        idMayor =
Integer.parseInt(idUsuario.getText())+1 ;
                        i++;
                    }
                }
            } catch (Exception e) {
                break;
            }
        }
        WebElement btnCrearProductoListar =
driver.findElement(By.name("crearUsuario"));
        btnCrearProductoListar.click();

        // Form
        WebElement nombres = driver.findElement(By.id("nombres"));
        nombres.sendKeys("Nombre Auto");

        WebElement apellidos = driver.findElement(By.id("apellidos"));
        apellidos.sendKeys("Matized");

        WebElement edad = driver.findElement(By.id("edad"));
        edad.sendKeys("66");

        WebElement sexo = driver.findElement(By.id("sexo"));
        sexo.sendKeys("M");
    }

```

```

        WebElement correoElectronico =
driver.findElement(By.id("correoElectronico"));
        correoElectronico.sendKeys("auto@matized.com");

        WebElement departamento =
driver.findElement(By.id("departamento"));
        departamento.sendKeys("Antioquia");

        WebElement ciudad = driver.findElement(By.id("ciudad"));
        ciudad.sendKeys("Medellin");

        WebElement direccion = driver.findElement(By.id("direccion"));
        direccion.sendKeys("Direccion Virtual");

        WebElement telefono = driver.findElement(By.id("telefono"));
        telefono.sendKeys("666 77 88");

        WebElement rol = driver.findElement(By.id("rol"));
        rol.sendKeys("Admin");

        WebElement cmdAceptar =
driver.findElement(By.name("crearUsuario"));
        cmdAceptar.click();

        // Assert

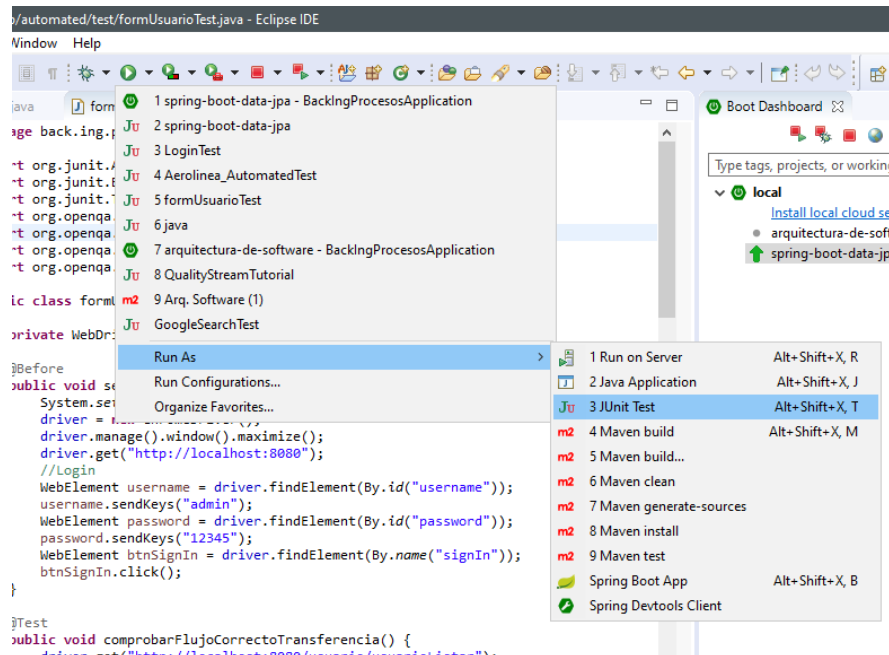
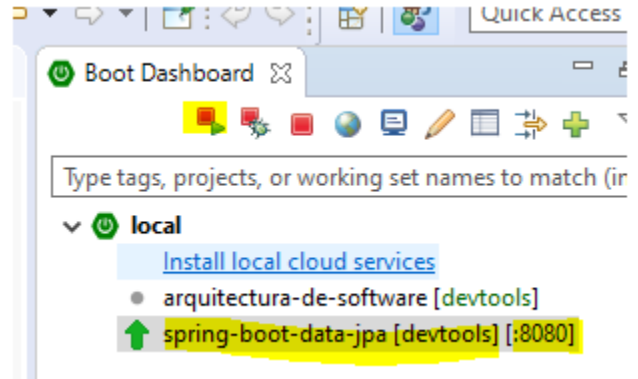
        boolean testExitoso = false;
        try {
            WebElement idProducto =
driver.findElement(By.name(String.valueOf(idMayor)));
            if (idProducto != null) {
                testExitoso = true;
            }
        } catch (Exception e) {
        }

        Assert.assertTrue(testExitoso);
    }
}

```

Antes de correr la prueba como Junit:

1. Revisar puerto de la página el proyecto que se va a testear en este caso corre <http://localhost:8080>
2. Instalar el plugin Spring Dashboard en eclipse para poder iniciar rápidamente la aplicación de arranque.
3. Revisar la versión de Google y descargar el Chromedriver correcto. Y guardarlo en la carpeta llamada resources test
4. Se ejecuta la prueba



El software de prueba esta utilizando Chrome y se automatiza el acceso al Login y el registro de un nuevo usuario.

Centro de Acopio Virtual

Administración

Registrarse

Por favor Iniciar sesión!

admin

.....

Registrarse

Listado de Usuarios

localhost:8080/usuario/usuarioListar

Un software de prueba automatizado está controlando Chrome.

Centro de Acopio Virtual

Administración

Admin

Listado de Usuarios

→

Crear Usuario

id	nombres	apellidos	edad	sexo	correoElectronico	departamento	ciudad	direccion	telefono	rol		
1	RAMIRO	BUILES	50	M	correo@rmo.com	ANTIOQUIA	MEDELLIN	En la ptm	3131111111	ADMIN	editar	eliminar
2	Nombre Auto	Matized	66	M	auto@matized.com	Antioquia	Medellin	Direccion Virtual	666 77 88	Admin	editar	eliminar
3	Nombre Auto	Matized	66	M	auto@matized.com	Antioquia	Medellin	Direccion Virtual	666 77 88	Admin	editar	eliminar
4	Nombre Auto	Matized	66	M	auto@matized.com	Antioquia	Medellin	Direccion Virtual	666 77 88	Admin	editar	eliminar

Package Explorer

JUnit

Finished after 26,371 seconds

Runs: 3/3

Errors: 1

Failures: 0

back.ing.procesos.app.automated.test.formProductoTest [Runner: JUnit 4] (0,104 s)

comprobarFlujoCorrectoTransferencia (0,104 s)

Failure Trace

java.lang.IllegalStateException: The driver executable does not exist: C:\Users\catalina\g

at com.google.common.base.Preconditions.checkState(Preconditions.java:585)

at org.openqa.selenium.remote.service.DriverService.checkExecutable(DriverService.java:100)

at org.openqa.selenium.remote.service.DriverService.findExecutable(DriverService.java:115)

at org.openqa.selenium.chrome.ChromeDriverService.access\$000(ChromeDriverService.java:16)

at org.openqa.selenium.chrome.ChromeDriverService\$Builder.findDefaultExecutable(ChromeDriverService.java:100)

at org.openqa.selenium.remote.service.DriverService\$Builder.build(DriverService.java:33)

at org.openqa.selenium.chrome.ChromeDriverService.createDefaultService(ChromeDriverService.java:44)

at org.openqa.selenium.chrome.ChromeDriver.<init>(ChromeDriver.java:123)

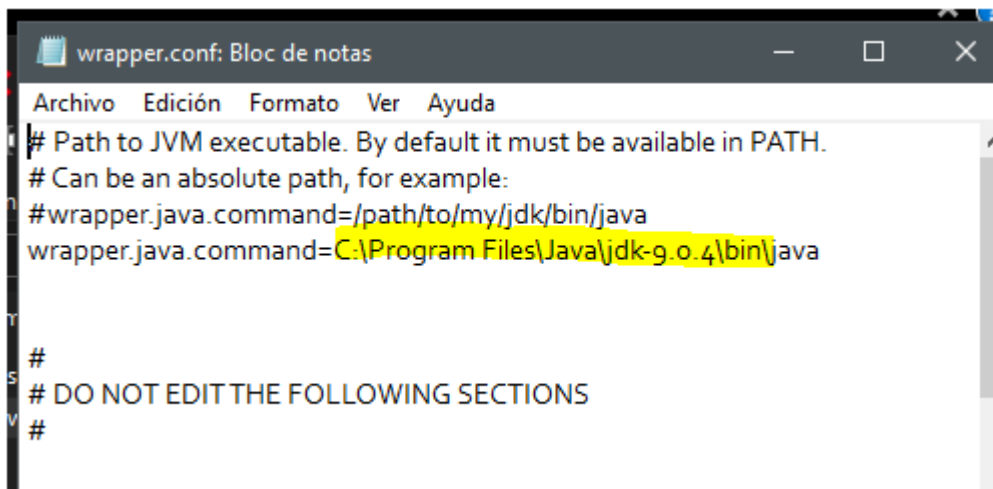
at back.ing.procesos.app.automated.test.formProductoTest.setUp(formProductoTest.java:10)

El test presenta un error en la ejecución.

Se carga el proyecto a GitHub, anexo tutorial que utilice de YouTube <https://youtu.be/8EVG7RmrP-o>

Descargar SonarQube

- Hay que tener en cuenta que la mayoría de las versiones de SonarQube necesitan Java 11.
- En caso contrario se debe editar el archivo wrapper.conf de la carpeta de descarga (C:\Users\catalina\Downloads\sonarqube-8.1.0.31237\conf) y Agregar la ubicacion de ejecución de java, guardar.
- Ejecutar el cmd en la dirección → C:\Users\catalina\Downloads\sonarqube-8.1.0.31237\bin\windows-x86-64 se escribe el comando "StartSonar.bat" para abrir el servidor de Sonar.



```
wrapper.conf: Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
# Path to JVM executable. By default it must be available in PATH.
# Can be an absolute path, for example:
# wrapper.java.command=/path/to/my/jdk/bin/java
wrapper.java.command=C:\Program Files\Java\jdk-9.0.4\bin\java
#
# DO NOT EDIT THE FOLLOWING SECTIONS
#
```

Ejecución de sus pruebas a través de Jenkins

El paso final es integrar Jenkins con el trabajo cargado a Github: después de que se hayan ejecutado las pruebas, Jenkins debe recoger los resultados de la prueba JUnit de la carpeta especificada en la tarea de *informe de junit*, si todo está configurado correctamente, ahora debería poder ejecutar sus pruebas a través de Jenkins y mostrar los resultados: También puede ver los detalles de los resultados de las pruebas individuales haciendo clic en el mensaje de error. Pero sin antes configuramos:

las rutas de las herramientas de algunos componentes e instalar o actualizar plugins Jenkins



Global Tool Configuration

Maven Configuration

Default settings provider	Use default maven settings
Default global settings provider	Use default maven global settings

JDK

instalaciones de JDK	<div>Añadir JDK</div> <div><div>JDK</div><div>Nombre</div><div>Java SE 8</div><div>JAVA_HOME</div><div>C:\Program Files\Java\jdk1.8.0_221</div><div><input type="checkbox"/> Instalar automáticamente</div></div>
----------------------	---

Git

Git installations	<div><div>Git</div><div>Name</div><div>gitJenkins</div><div>Path to Git executable</div><div>C:\Program Files\Git\bin\git.exe</div><div><input type="checkbox"/> Instalar automáticamente</div></div>
-------------------	---

Add Git ▼

Listado de instalaciones de Ant en este sistema

Maven

instalaciones de Maven	<div>Añadir Maven</div> <div><div>Maven</div><div>Nombre</div><div>maven-3.6.0</div><div>MAVEN_HOME</div><div>C:\MAVEN\apache-maven-3.6.0</div><div><input type="checkbox"/> Instalar automáticamente</div></div>
------------------------	---

SonarQube servers

Environment variables	<input type="checkbox"/> Enable injection of SonarQube server configuration as build environmen
Instalaciones de SonarQube	<div>If checked, job administrators will be able to inject a SonarQube server configuration as environ</div> <div><div>Name</div><div>sonar 2.6.1</div><div>URL del servidor</div><div>http://localhost:9000</div><div>Por defecto es http://localhost:9000</div><div>Server authentication token</div><div><div>- none -</div><div>Add ▼</div></div><div>SonarQube authentication token. Mandatory when anony</div></div>

● Compilar #3 (22-abr-2020 5:25:11)

 [añadir descripción](#)



Changes

1. se edita archivo HELP.md para prueba ([commit: 430e5b4](#)) ([details](#) / [githubweb](#))



Iniciado por el usuario [johana c rios](#)



Revision: 430e5b41f965cf5846394a672deafca608fedb17

- refs/remotes/origin/master

● Salida de consola

```
Lanzada por el usuario johana c rios
Running as SYSTEM
Ejecutando en el espacio de trabajo C:\Program Files (x86)\Jenkins\workspace\loginUsuarios
No credentials specified
> C:\Program Files\Git\bin\git.exe rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/jcriostorres/LoginUsuarios\_Test.git #
timeout=10
Fetching upstream changes from https://github.com/jcriostorres/LoginUsuarios\_Test.git
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> C:\Program Files\Git\bin\git.exe fetch --tags --progress -- https://github.com/jcriostorres/LoginUsuarios\_Test.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/origin/master^{commit}" # timeout=10
Checking out Revision 430e5b41f965cf5846394a672deafca608fedb17 (refs/remotes/origin/master)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f 430e5b41f965cf5846394a672deafca608fedb17 # timeout=10
Commit message: "se edita archivo HELP.md para prueba"
> C:\Program Files\Git\bin\git.exe rev-list --no-walk 10fc09942287226ab4e8ccff6eed59f3486fb5d # timeout=10
Finished: SUCCESS
```