Groovy script for invoking a UI from SoapUI using Groovy

script enabled by Selenium Jars:

import com.eviware.soapui.model.\*

import com.eviware.soapui.model.testsuite.Assertable

import com.eviware.soapui.support.XmlHolder

import java.io.File;

def regLogger = org.apache.log4j.Logger.getLogger("RegressionTestLog

def groovyUtils = new com.eviware.soapui.support.GroovyUtils( context

def properties = new java.util.Properties();

import org.openqa.selenium.remote.CapabilityType;

import org.openqa.selenium.remote.DesiredCapabilities;

import com.thoughtworks.selenium.Selenium;

WebDriver driver = new FirefoxDriver();

driver.manage().deleteAllCookies();

driver.manage().window().maximize();

WebDriver driver = new FirefoxDriver();

driver.get("http://www.google.com ");

**Just for reference, the following is a sample script for UI automation using the**

**Chrome driver:**

import org.openqa.selenium.\*;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.interactions.Actions;

import org.openqa.selenium.ie.InternetExplorerDriver;

import com.eviware.soapui.model.\*

import org.openqa.selenium.chrome.ChromeDriver;

import com.eviware.soapui.model.testsuite.Assertable

import com.eviware.soapui.support.XmlHolder

import java.io.File;

//import java.util.\*;

//import jxl.write.\*

//import jxl.\*

def regLogger =

org.apache.log4j.Logger.getLogger("RegressionTestLoger");

def groovyUtils = new com.eviware.soapui.support.GroovyUtils(

context )

def properties = new java.util.Properties();

import org.openqa.selenium.remote.CapabilityType;

import org.openqa.selenium.remote.DesiredCapabilities;

import com.thoughtworks.selenium.Selenium;

//System.setProperty("webdriver.chrome.driver", "C:\\Program

Files\\SmartBear\\SoapUI-5.0.0\\bin\\ext\\chromedriver.exe");

//WebDriver driver = new ChromeDriver();

driver.manage().deleteAllCookies();

driver.get("http://www.google.com ");

try{

synchronized(log)

{

UniqueID = (testRunner.testCase.testSuite.

getPropertyValue("UniqueID")).toLong()+1000;

testRunner.testCase.testSuite.setPropertyValue("UniqueID",Unique

ID.toString ());

testRunner.testCase.setPropertyValue("UniqueID",UniqueID.toString

());

regLogger.info("\* UniqueID: " + UniqueID + " , Case: " +

testRunner.testCase.name);

}

}catch(Exception e){

log.info(Error+e);

}

Following is a sample script for test automation on the IE browser:

import org.openqa.selenium.\*;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.interactions.Actions;

import org.openqa.selenium.ie.InternetExplorerDriver;

import com.eviware.soapui.model.\*

import org.openqa.selenium.chrome.ChromeDriver;

import com.eviware.soapui.model.testsuite.Assertable

import com.eviware.soapui.support.XmlHolder

import java.io.File;

**Multilayer Test Automation Using SoapUI and Selenium**

def regLogger =

org.apache.log4j.Logger.getLogger("RegressionTestLoger");

def groovyUtils = new com.eviware.soapui.support.GroovyUtils( context

)

def properties = new java.util.Properties();

import org.openqa.selenium.remote.CapabilityType;

import org.openqa.selenium.remote.DesiredCapabilities;

import com.thoughtworks.selenium.Selenium;

File file = new File("C:\\Program Files\\SmartBear\\SoapUI-5.0.0\\

bin\\ext\\IEDriverServer\_x64\_2.39.0\\IEDriverServer.exe");

System.setProperty("webdriver.ie.driver", file.getAbsolutePath());

WebDriver driver = new InternetExplorerDriver()

better. We should also note that XPath will remain the same for

WebElement LoginName =

driver.findElement(By.xpath("//\*[@id='loginname']"));

LoginName.click();

LoginName.sendKeys("admin");

WebElement Password =

driver.findElement(By.xpath("//\*[@id='passwd']"));

Password.click();

Password.sendKeys("admin");

WebElement Login =

driver.findElement(By.xpath("//\*[@id='loginBtn']"));

Login.click();

WebElement Cardno = driver.findElement(By.xpath("//\*[@id='Credit

card Number']"));

Cardno.click();

Cardno.sendKeys("1098987654");

[ 13 ]

Multilayer Test Automation Using SoapUI and Selenium

This way we can enter data in any field on the UI page. Once we have

logged into the application and submitted our card details on the

screen, we need to click on the Submit button:

WebElement SUBMIT=

driver.findElement(By.xpath("//\*[@id='SBTBtn']"));

SUBMIT.click();

As per our scenario, we will receive a unique number in the UI. Let's name it UID.

Now we need to fetch the UID and pass it to the properties file so that we can re-

utilize it in the next step which is a test request.

Here is sample code on how to achieve it:

def UID = driver.findElement(By.xpath("//\*[@Name='UniqueID']")).get

testRunner.testCase.setPropertyValue("UID",UID.toString());

Now you can use this UID throughout your test case. Once this is done, we can

parameterize our test request and run it. This will complete step 3 of our scenario.

1. We need to run a service request; we can do that easily using SoapUI.

2. We again need to run the service but we need to save the response in the

properties of SoapUI which can be done with Groovy script or property

transfer with the help of XPath as we have learnt in the previous chapters.

We will save the property name as Token.

3. Now we will return to the browser and get the property details from the

soap UI properties and enter the details in the token field. See the following

def Token = (testRunner.testCase.getPropertyValue("Token"))

Token.sendKeys(TOKEN);