Using Ant: We have in previous chapters seen how can we create a build.xml file

to execute and generate reports for our Test automation suites. We can use the same

build.xml file and the same shall work for us.

<arg line= "-j -f 'c:\project\SoapUITest ' ' like

c:/project/SoapUITest like c:/project/SoapUITest

/Production.xml'"/>

</exec>

</target>

<target name = "testreport" depends ="soapui">

<junitreport todir=

"C:/Users/pnandan/Desktop/Analysis/NewFolder">

<fileset dir="C:/Users/pnandan/Desktop/Analysis/NewFolder">

<include name="TEST-TestSuite\_1.xml"/>

</fileset>

<report todir="C:/Users/Admin / Analysis/Reports /HTML"

styledir="C:/Testing/apache-ant-1.9.6/etc"

format="frames">

</report>

</junitreport>

</target>

</project>

SoapUI Integration with Jenkins and HP QC

The second option we have is to invoke a batch file. We can use the command line

functionality of SoapUI to create a batch file which executes the project:

Commandline utility of soap UI

Sample Batch file:

cd C:\Program Files\SmartBear\soapUI-4.0.1\bin testrunner.bat

-ehttps://192.90.190.19:7001/test?wsdl -sTestSuite -r -a -fC:\Users\

pnandan\Desktop\batchsoap -I "C:\Users\pnandan\Desktop\batsoap\

soapuitestproject.xml"

Excel integration via Groovy script

import com.eviware.soapui.model.\*

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();

def s2

def s3=(testRunner.testCase.getPropertyValue("RUN"))

regLogger.info(s3);

if (s3 !=1 & s2 != 1&s3 !=3);

{ testRunner.testCase.setPropertyValue("RUN");

s3=(testRunner.testCase.getPropertyValue("RUN"));

}

Workbook workbook = Workbook.getWorkbook(new

File(D:\\testdata.xls)

End-to-End Test Automation

for (count in 2..< 11)

{

Sheet sheet = workbook.getSheet(1)

Cell a1 = sheet.getCell(0,count) // getCell(row,column) —

place some values in myfile.xlsCell b2 =

sheet.getCell(s3.toInteger(),count) // then those values

will be acessed using a1, b2 &amp; c3 Cell.

String s1 = a1.getContents();

s2 = b2.getContents();

//Cell c2 = sheet.getCell(2,1)

testRunner.testCase.setPropertyValue(s1,s2);

}

workbook.close()

The following is a sample script for Selenium integration:

import org.openqa.selenium.By

import org.openqa.selenium.WebDriver

import org.openqa.selenium.WebElement

import org.openqa.selenium.firefox.FirefoxDriver

import org.openqa.selenium.support.ui.ExpectedCondition

import org.openqa.selenium.support.ui.WebDriverWait

WebDriver driver = new FirefoxDriver()

// And now use this to visit facebook

driver.get("http://www.Facebook.com") // Find the

text input element by its name

WebElement element =

driver.findElement(By.name("Login")); // Enter something

to search for

element.sendKeys("pranai!")

//Close the browser

driver.quit()

Alternatively we can also use the following Groovy script to connect to

queues and topics:

import com.eviware.soapui.impl.wsdl.submit.transports.jms.

JMSConnectionHolder

import com.eviware.soapui.impl.wsdl.submit.transports.jms.util.

HermesUtils

import com.eviware.soapui.impl.wsdl.submit.transports.jms.

JMSEndpoint

[ 6 ]

import hermes.Hermes

import javax.jms.\*

def jmsEndpoint = new JMSEndpoint("jms://activeMQSession::queue::

queueQ1");

def hermes = HermesUtils.getHermes( context.testCase.testSuite.

project, jmsEndpoint.sessionName)

def jmsConnectionHolder = new JMSConnectionHolder( jmsEndpoint,

hermes, false, null ,null ,null);

Session queueSession = jmsConnectionHolder.getSession();

Queue queueSend = jmsConnectionHolder.getQueue(

jmsConnectionHolder.getJmsEndpoint().getSend() );

Queue queueBrowse = jmsConnectionHolder.getQueue(

jmsConnectionHolder.getJmsEndpoint().getReceive() );

MessageProducer messageProducer =queueSession.createProducer(

queueSend );

TextMessage textMessageSend = queueSession.createTextMessage();

textMessageSend.setText( "jms message from groovy");

messageProducer.send( textMessageSend );

textMessageSend.setText( "another jms message from groovy");

messageProducer.send( textMessageSend );

QueueBrowser qb = queueSession.createBrowser(queueBrowse);

Enumeration en= qb.getEnumeration();

while(en.hasMoreElements()){

TextMessage tm = (TextMessage)en.nextElement();

log.info tm.getText()

}

jmsConnectionHolder.closeAll()

Here is the build.xml for generating reports.

arg line= "-j -f 'C:/Users/pnandan/Desktop/Analysis/NewFolder'

'C:/Users/pnandan/Desktop/Test/Pranai/Production.xml'"/>

</exec>

</target>

<target name = "testreport" depends ="soapui">

<junitreport todir="C:/Users/pnandan/Desktop/Analysis/NewFolder">

<fileset dir="C:/Users/pnandan/Desktop/Analysis/NewFolder">

<include name="TEST-TestSuite\_1.xml"/>

</fileset>

<report todir="C:/Users/pnandan/Desktop/Analysis/NewFolder/HTML"

styledir="C:/Testing/apache-ant-1.9.6/etc"

format="frames">

</report>

</junitreport>

</target>

</project>

here is the build.xml file for sending e-mails:

<property name="line2" value="Message"/>

<echo message="${line2}"/>

<mail mailhost="pranainandan08@gmail.com" mailport="25"

subject="Test build" charset="utf-8">

<from address=" pranainandan08@gmail.com "/>

<to address="all@xyz.com"/>

<message>Test Message</message>

</mail>