==============================================================================

Creating Reports in Selenium

----------------------------

WD is not provided any library to generate test execution reports

whereas we can use some 3rd party APIs to generate reports

1. using TestNg framework

2. using log4j.properties

3. ExtentReports v4

1. using TestNg framework:

------------------------

by default TestNg will generate html reports for each suite execution

disadvantage:

it is not possible to customize

2. \*\*Apache Log4j (log4j.properties)

----------------------------------

-Log4j is an open source framework provided by Apache for Java projects

-Log4j is a fast, flexible and reliable logging framework

-\*\*With Log4j, it is possible to store the flow details of our Selenium

Automation execution result in console as well as in File (.log)

-Log4j is used for large as well as small projects

-In Log4j, we use log statements rather than SOPL statements in the

code to know the status of a project while it is executing

Log4j has three main components:

i. Appender: Responsible for publishing logging information to various

preferred destinations like Console/File/DB/SMTP (mail)

ii. Layouts: Responsible for formatting logging information in different

styles

iii.Logger: It is a class provided in Log4j

Responsible for capturing logging information

we have to initialize one Logger object for each Java class

syntax: {To create Logger Object}

Logger obj= Logger.getLogger(CurrentClassName.class);

we use Logger methods/levels to generate log statements like trace, debug,

info,warn, error, fatal...etc

ex:

logger.info(msg);

logger.error(msg);

Note:

Appender and Layouts are maintained in "log4j.properties" file, whereas Logger

class we use in the script

PropertyConfigurator:

this class used to initialize log4j.properties file

syntax:

PropertyConfigurator.configure(property file name);

Working with Log4j:

--------------------

Step 1: Create new java project

Step 2: Configure Log4j library to java project

Navigation:

Download log4j.Jar file

http://www-us.apache.org/dist/logging/log4j/1.2.17/log4j-1.2.17.zip

extract zip file

configure "log4j-1.2.17" jar file to java project

Note:

based on requirement we can also configure WD jars and TestNg jars

Step 3:create log4j.properties file in project Home directory

navigation:

Right click on "java project"

New

File

enter file name as "log4j.properties"

Click on Finish

Step 4: write following code into log4j.properties file

log4j properties file configuration script

----------------------------------------

# Here we have defined root logger

log4j.rootLogger=DEBUG,CONSOLE,FILE

#Console configuration

log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender

log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout

log4j.appender.CONSOLE.layout.ConversionPattern= %d{dd-MM-yyyy hh:mm:ss} %c %M -%m%n

#File configuration

log4j.appender.FILE=org.apache.log4j.FileAppender

log4j.appender.FILE.layout=org.apache.log4j.PatternLayout

log4j.appender.FILE.file=./TestResults/AppLogs.log

log4j.appender.FILE.layout.ConversionPattern= %d{dd-MM-yyyy hh:mm:ss} %c %M -%m%n

log4j.appender.FILE.Append=true

=============================================================================

Step 5: create TestNg class and use Logger class to pass messages into log file

and console

package testng.framework;

import org.apache.log4j.Logger;

import org.apache.log4j.PropertyConfigurator;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.Test;

public class Log4jDemo {

private Logger logger;

public WebDriver driver;

@Test

public void setUp() {

logger=Logger.getLogger("Log4jDemo.class");

PropertyConfigurator.configure("log4j.properties");

System.setProperty("webdriver.chrome.driver", "D:\\SelResource\\chromedriver.exe");

logger.info("Browser executable file running");

driver=new ChromeDriver();

logger.info("Browser launched");

driver.get("https://opensource-demo.orangehrmlive.com/");

logger.info("Application opened");

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

}

}

==============================================================

Ex: Log4j properties

package reports;

import org.apache.log4j.Logger;

import org.apache.log4j.PropertyConfigurator;

import org.testng.annotations.Test;

public class DemoLog4j {

private Logger logger;

@Test

public void userLogin() {

logger=Logger.getLogger("DemoLog4j.class");

PropertyConfigurator.configure("./log4j.properties");

System.out.println("\n Printing Logger messages \n");

logger.trace("to trace the execution");

logger.info("Browser launched");

logger.error("it in not printed");

logger.warn("printing warning messages");

logger.fatal("It is a fatal defect");

System.out.println("\n Printing Logger messages are completed \n");

}

}

================================================================

3. Extent Reports v4 in Selenium Webdriver

--------------------------------------------

Extent Reports(Tool) is also used as Selenium Reporting Tool

We can create interactive and detailed reports using Extent framework

such as events,pie charts, screenshots, tags, devices, author...etc

Steps To Generate Extent Reports in Selenium:

Step 1: create a TestNG project in eclipse

Step 2: Download Extent reports Version4 JAR files

url: http://extentreports.com/community/

Step 3: Add the downloaded library (jar) files to your project

(external jar file and Dependency jars)

\*\*\*Classes in Extent Reports jars:

i. ExtentHtmlReporter:

using this class we can customize UI of the Reports like

Report id,Report name, Theme...etc

ii. ExtentReports:

to entry in reports

iii. ExtentTest:

used to update reports

Step 4: Create Class using Extent Report classes to verify OrangeHRM page

title and also provide screenshot of page with date stamp in Reports

package sample.pack;

import java.io.File;

import java.io.IOException;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.Date;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.io.FileHandler;

import org.testng.Assert;

import org.testng.annotations.AfterClass;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentHtmlReporter;

import com.aventstack.extentreports.reporter.configuration.Theme;

public class ExtentDemo {

ExtentHtmlReporter htmlReporter;

ExtentReports extent;

ExtentTest test;

WebDriver driver;

@BeforeClass

public void extentInit() {

//to specify the location

htmlReporter=new ExtentHtmlReporter("./TestResults//ExtentReporter.html");

// Title of the report

htmlReporter.config().setDocumentTitle("Automation Report");

//Name of the report

htmlReporter.config().setReportName("Functional Report");

//To Specify Theme like STANDARD/DARK

htmlReporter.config().setTheme(Theme.STANDARD);

//create ExtentReports object and attach Reports

extent=new ExtentReports();

extent.attachReporter(htmlReporter);

extent.setSystemInfo("Hostname", "LocalHost");

extent.setSystemInfo("OS", "Windows10");

extent.setSystemInfo("Tester Name", "Madhukar");

extent.setSystemInfo("Browser", "Chrome");

//create ExtentTest object and specify Report Title

test=extent.createTest("OrangeHRM\_TitleTest");

}

@BeforeMethod

public void setUp()

{

System.setProperty("webdriver.chrome.driver", "D:\\SelResource\\chromedriver.exe");

driver = new ChromeDriver();

test.log(Status.INFO,"Chrome browser launched");

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

driver.get("https://opensource-demo.orangehrmlive.com/");

test.log(Status.INFO,"Navigated Orange HRM Application");

}

@Test

public void titleTest() throws IOException {

String pgTitle=driver.getTitle();

Assert.assertEquals("OrangeHRM", pgTitle);

test.log(Status.PASS, "Expected title exist");

File f=((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);

//to get Date stamping with file

Date dt= new Date(); //it will return system date

DateFormat df1= new SimpleDateFormat("dd\_MM\_yyyy\_HH\_MM\_ss");//to convert into required format

//to save screenshot with date stamping

FileHandler.copy(f,new File("D:\\titleScreenshot"+df1.format(dt)+".png")) ;

test.addScreenCaptureFromPath("D:\\titleScreenshot"+df1.format(dt)+".png");// adding screen shot

}

@AfterMethod

public void tearDown() {

driver.close();

}

@AfterClass

public void endReport()

{

extent.flush();

}

}

---------------------------------------------------------------------------

Maven

-----

-it is a build execution tool

-it will compile all the programs and create build

-other similar tools which are support java are ANT, Gradle

Maven Configuration:

Step 1:Download Maven

open URL: https://maven.apache.org/download.cgi

click on "apache-maven-3.6.3-bin" under "File" section

Extract zip file

Step 2: Specify maven path to System Environment variable

MAVEN\_HOME: D:\SelResource\apache-maven-3.6.3

path: D:\SelResource\apache-maven-3.6.3\bin

Note:

To check maven version:

go to command prompt

type: mvn -version

click on Enter

Working with Maven in Selenium

-----------------------------

Step 1: create Maven project in Eclipse

Navigation:

File menu in Eclipse

New

Select "Others"

Maven

select "Maven Project"

Select "create simple project"

Enter group id (package name)

enter Artifact id (project name)

click on "Finish"

Note:

Maven Project structure:

-->src/test/java

in this folder we can

-create packages

-create automation programs for testing

-->src/main/resources

in this folder we maintain

-create testng.xml file

-testdata excel files

-properties files...etc

Step 2: work with POM.xml file (Project Object Model)

we can add all the dependency and Plugin details in pom.xml file

dependencies:used to download jars like Webdriver jars, TestNg jars,

apache poi...etc

plugins:used to run your Maven project

maven-compiler-plugin: to compile java programs

maven-surefire-plugin : to run Testng xml files

navigation:

Go to mvn repository website

search for selenium jars, poi jars, testng jars, extentreports jars...etc

those are copy and paste in the pom.xml file

or you can Copy and paste the following code in pom.xml file

==============================================

<properties>

<suiteXmlFile>src/main/resources/testng.xml</suiteXmlFile>

</properties>

<dependencies>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.9.9</version>

</dependency>

<!-- Adding Selenium dependency -->

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>3.141.59</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.0</version>

<configuration>

<compilerVersion>1.8</compilerVersion>

<source>1.6</source>

<target>1.6</target>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.17</version>

<configuration>

<suiteXmlFiles>

<suiteXmlFile>${suiteXmlFile}</suiteXmlFile>

</suiteXmlFiles>

</configuration>

</plugin>

</plugins>

</build>

=========================================================

Note:

After copy and paste dependencies and plugins pom.xml file then save file

if it is showing any error then update the maven project

Navigation:

Right click on Maven project

Maven

select "update project"

Note:

the dependencies added in pom.xml, those jar files are created in our localsystem

C:\Users\HP\.m2\repository

Step 3: Create package in "src/test/java" folder

Step 4: Create TestNg class

public class TestMvn {

@Test

public void sampleTest() {

System.setProperty("webdriver.chrome.driver", "D:\\SelResource\\chromedriver.exe");

WebDriver driver=new ChromeDriver();

driver.get("https://opensource-demo.orangehrmlive.com/");

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

}

}

Step 5: Convert TestNg class into Testng.xml file

Navigation:

select class

right click on mouse

select "Convert to Testng"

click on "Finish"

Note:

move the Testng.xml file into "src/main/Resource" folder

========================================

Executing test cases:

we can execute test cases in 4 ways

1. Run test cases using pom.xml within eclipse

Navigation:

select pom.xml file

right click on mouse

Run As

select "Maven Test"

Note:

Note:

[ERROR] Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:3.5.1:testCompile (default-testCompile) on project MavenPro: Compilation failure

[ERROR] No compiler is provided in this environment. Perhaps you are running on a JRE rather than a JDK?

Navigation:

window menu in Eclipse

Preferences

Enter Installed JREs

Click on ADD

Click on Next

Click on Directory

Browse JDK folder path( C:\Program Files\Java\jdk1.8.0\_144)

Click on Finish

Select JDK checkbox

Click on OK

2. Run test cases from command prompt(without using eclipse)

Navigation:

Goto project directory

Shift+right click on mouse

select "open powerShell window"

enter "mvn clean install"

click on Enter

3. Run test cases using batch file:

Creating batch file:

open notepad

write code "call mvn test -f <location of pom.xml file>"

(Ex:call mvn test -f E:\Sel\_LAB\11amJuly19\MavenProject\pom.xml)

saveas with ".bat" extension

click on that file to run batch file

4. Run test cases using Jenkins:

Continuous Integration (CI) Tool

---------------------------------

These tools are used to continuously integrated the scripts and to execute

as per schedules or whenever code changes are done

Jenkins:

It is a continuous integration tool (CI)

It is a Web based tool, configured in the server

working with jenkins:

it is a web based application we need url

open any one browser IE/Chrome/FF

Enter url: http://localhost:8080 (in general which is installed at

DevOps Enviornment)

enter login credentials and click on Login

uid: MadhukarQAIT

pwd: Mercury

Adding project into Jenkins:

click on "New Item"

Enter item name (i.e. project name)

click on "Free style" project

click on "OK"

select "Build Trigger Event"

Select "Build Periodically" under

"Build Triggers"

enter schedule timings

ex:H/15 \* \* \* \* (to execute every 15 min. in hour)

Select "Build" tab

click on "add build step"

select "Execute Windows batch command"

enter path of u r batch file/maven project pom.xml file

click on apply &Save

click on "Build now"

back to "Dashboard"

============================================================================

Git --> is a local repository

Github --> is a Remote repository