Q.1 What are the Different types of framework in selenium?

* Key driven framework
* TestNG Framework
* Datadriven Framework
* Hybrid framework
* Cucumber Framework
* Modular Framework

Q.2 How to implement Framework and How to start framework implementation from basics?

Q.3 What is collection Framwork?

We have a predefined group of classes and interfaces in collection framework..we are uses this implementation classes and interfaces as a user.

Q.4 Why we are using framework ? what are its advantages?

We are implement our own framework to automate an application

Advantages

1. Reusability of code increases- we are going to reuse that code utility

2. We are avoid duplication of the code

3. We maintain the code

4. we build security of the code

Q.5 As an Automation test Engineer what points we have to focus when building framework?

1. Implementation

2. Execution

3. Mainenance

## Q What is Maven?

Maven is a project management and comprehension tool that provides developers a complete build lifecycle framework. Development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.

In case of multiple development teams environment, Maven can set-up the way to work as per standards in a very short time. As most of the project setups are simple and reusable, Maven makes life of developer easy while creating reports, checks, build and testing automation setups.

Maven provides developers ways to manage the following −

* Builds
* Documentation
* Reporting
* Dependencies
* SCMs
* Releases
* Distribution
* Mailing list

Q.6 How to install Maven plugin?

Open Eclipse

Go to help

Open Eclipse Marketplace

Search Maven and Install it

Q.7 How to create Maven Project?

Go to file

New

Other

See there Maven , In that Maven there is maven project click next next without changing anything

Write Group id and Arifact Id..Group id is nothing but Package name and Arifact id is like project name

Click on finish.

We see M for maven project at RHS

In that folder we will get src/main/java

And src/test/java inside that there is hybrid framework package

We will get JRE System library

We will get maven Dependencies and Junit 3.8 jar files

We see pom.xml and this is imp

Q.7 What is pom.xml in Maven

We can download automatically all required jars using pom.xml from maven repository.

Whatever jars we are downloading from maven remote repository that going to attached or downloaded in maven local repository. In pom.xml file there is dependency tag in that we are going to add required dependencies. By default there are dependency jars we simply delete it.

Then we are deleting default created packages at RHS under src/main/java and src/test/java

Here we are creating our own packages.

Q.8 why we are using dependencies in maven project? Or What is meant by dependencies.

Whatever jars we are going to add like Selenium webdriver jars and Test Ng jars, poi xml jars, poi API jars for advance reporting extent report jars, for login purpose log file jars.. we no need to download it , simply we should go to the maven repository then we should update those jars by using dependencies.

Go to the any browser and search maven repository. copy and paste in poi xml under dependencies . format and save it then all dependencies get downloaded into maven repository RHS.

As a Automation test Engineer we have to work with src/test/java folder RHS

Q.9 What is the standard way to create packages?

Com. Your company name, which project appln and project name

e.g com.visionit.orangeHRM.login

Q.10 Write a sample framework

1. Create a java class TestBase:- Test Base is helps us to load the browser and perform some activity..within this test base we are writing some common code so that we can reuse all code like load the browser.. , Navigate to the application.., close the browser., Report we are going to generate, capturing the screen shot. Again and again what utility we are going to use we will keep in Testbase

// write the set up method ..Test case we are writing @BeforeMethod

2. In PageObject we are going to design the Page..How we are going to design patterns.

Create a LoginPage .java class

**File name-LoginPage.java**

**package** com.visionit.orangehrm.pageObject;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.support.FindBy;

**import** org.openqa.selenium.support.PageFactory;

**public** **class** LoginPage {

WebDriver driver;

// find a webelement using findby methods

@FindBy(name = "txtUsername")

WebElement username;

@FindBy(name = "txtPassword")

WebElement userpass;

@FindBy(id = "btnLogin")

WebElement loginBtn;

// create a web Element repository at page level

// create a login constructor..constructor name and login name must be same.

**public** LoginPage(WebDriver driver) {

**this**.driver = driver;

// By using page factory class we are going to initialise all webelements

PageFactory.*initElements*(driver, **this**);

}

// This is the action method

**public** **void** loginOrangeHrm() {

username.sendKeys("Admin");

userpass.sendKeys("admin123");

loginBtn.click();

}

}

**File name-TestBase.java**

**package** com.visionit.orangehrm.testBase;

**import** java.util.concurrent.TimeUnit;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.testng.annotations.AfterMethod;

**import** org.testng.annotations.BeforeMethod;

**public** **class** TestBase {

**public** **static** WebDriver *driver* = **null**;

@BeforeMethod

**public** **void** setUp() {

// set the System variable path

// if you dont want to use system variable path you can download driver manager

// dependencies

// it will automatically check browser version.

// set up the System variable path

System.*setProperty*("webdriver.chrome.driver", "./Drivers/chromedriver.exe");

// create the driver instance of chromedriver.

*driver* = **new** ChromeDriver();

// navigate to the application

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

*driver*.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

}

@AfterMethod

**public** **void** tearDown() {

// quit the browser

*driver*.quit();

}

}

FileName- Login\_Tc\_001.java

**package** com.visionit.orangehrm.testcases;

**import** org.testng.annotations.Test;

**import** com.visionit.orangehrm.pageObject.LoginPage;

**import** com.visionit.orangehrm.testBase.TestBase;

**public** **class** Login\_Tc\_001 **extends** TestBase {

@Test

**public** **void** loginOrangeHrmTest() {

LoginPage login = **new** LoginPage(*driver*);

login.loginOrangeHrm();

}

}

**New Session:** 3**Selenium Web**Driver | Hybrid Framework part03

Here username and password is visible we are going to do one thing we are going to hide this..

Q.1 How we hide this?

By using config Data Provider.. Whatever configurable data is there we will get from external file.

Q.2 How to change or update jdk version in maven?

Add in Properties tag.

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

The moment we will create instance of Driver, it will launch the browser use following commands

Driver = new ChromeDriver();

Driver = newFirefoxDriver();

Q.3 What is Properties class in maven?

Properties class help us to store key value ..it is a legacy class..

**package** com.visionit.orangehrm.pageObject;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.support.FindBy;

**import** org.openqa.selenium.support.PageFactory;

1.. LoginPage.java

**public** **class** LoginPage {

WebDriver driver;

// find a webelement using findby methods

@FindBy(name = "txtUsername")

WebElement username;

@FindBy(name = "txtPassword")

WebElement userpass;

@FindBy(id = "btnLogin")

WebElement loginBtn;

// create a web Element repository at page level

// create a login constructor..constructor name and login name must be same.

**public** LoginPage(WebDriver driver) {

**this**.driver = driver;

// By using page factory class we are going to initialise all webelements

PageFactory.*initElements*(driver, **this**);

}

// This is the action method

**public** **void** loginOrangeHrm(String uname,String upass) {

//username.sendKeys("Admin");

//userpass.sendKeys("admin123");

username.sendKeys(uname);

userpass.sendKeys(upass);

loginBtn.click();

}

}

2. TestBase.java

**package** com.visionit.orangehrm.testBase;

**import** java.util.concurrent.TimeUnit;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.openqa.selenium.ie.InternetExplorerDriver;

**import** org.testng.annotations.AfterMethod;

**import** org.testng.annotations.BeforeMethod;

**import** org.testng.annotations.BeforeSuite;

**import** org.testng.annotations.Optional;

**import** org.testng.annotations.Parameters;

**import** com.visionit.orangehrm.utilities.ConfigDataProvider;

**public** **class** TestBase {

**public** **static** WebDriver *driver* = **null**;

**public** ConfigDataProvider configData;

@BeforeSuite

**public** **void** setupSuite() {

configData = **new** ConfigDataProvider();

}

@Parameters("Browser")

@BeforeMethod

**public** **void** setUp(@Optional("Chrome")String browserName) {

**if**(browserName.equals("Chrome")) {

// if we want to execute test cases on different browser

// set up the System variable path

// set the System variable path

// if you dont want to use system variable path you can download driver manager

// dependencies

// it will automatically check browser version.

System.*setProperty*("webdriver.chrome.driver", "./Drivers/chromedriver.exe");

// create the driver instance of chromedriver.

*driver* = **new** ChromeDriver();

// navigate to the application

} **else** **if**(browserName.equals("Firefox")) {

System.*setProperty*("webdriver.gecko.driver", "./Drivers/geckodriver.exe");

*driver* = **new** FirefoxDriver();

}**else** **if**(browserName.equals("IE")) {

System.*setProperty*("webdriver.ie.driver", "./Drivers/IEDriverServer.exe");

*driver* = **new** InternetExplorerDriver();

}

// set the System variable path

// if you dont want to use system variable path you can download driver manager

// dependencies

// it will automatically check browser version.

//driver.get(url);

*driver*.get(configData.getAppUrl());

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

*driver*.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

}

@AfterMethod

**public** **void** tearDown() {

// quit the browser

*driver*.quit();

}

}

3. Login\_Tc\_001 **.java**

**package** com.visionit.orangehrm.testcases;

**import** org.testng.annotations.Test;

**import** com.visionit.orangehrm.pageObject.LoginPage;

**import** com.visionit.orangehrm.testBase.TestBase;

**public** **class** Login\_Tc\_001 **extends** TestBase {

@Test

**public** **void** loginOrangeHrmTest() {

LoginPage login = **new** LoginPage(*driver*);

login.loginOrangeHrm(configData.getUserName(),configData.getUserPassword());

}

}

4. ConfigDataProvider.java

**package** com.visionit.orangehrm.utilities;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.util.Properties;

**public** **class** ConfigDataProvider {

**public** Properties prop;

// constructor

**public** ConfigDataProvider() {

**try** {

File fs = **new** File("./Config/config.properties");

FileInputStream fins = **new** FileInputStream(fs);

prop = **new** Properties();

prop.load(fins);

} **catch** (Exception e) {

e.printStackTrace();

}

}

**public** String getKeyValue(String searchKey) {

**return** prop.getProperty("searchKey");

}

**public** String getUserName() {

**return** prop.getProperty("username");

}

**public** String getUserPassword() {

**return** prop.getProperty("password");

}

**public** String getAppUrl() {

**return** prop.getProperty("url");

}

}

5..

6… login.Test.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name=*"Suite"*>

<test name=*"Test"*>

<parameter name = *"Browser"* value = *"Chrome"*></parameter>

<classes>

<class name=*"com.visionit.orangehrm.testcases.Login\_Tc\_001"*/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

7. .. config.properties

# login credentials of an application

username = Admin

password = admin123

url = https://opensource-demo.orangehrmlive.com/

#key = value

## New Session: 3Selenium WebDriver | Hybrid Framework part04

## Create a new java class under utility package XLXSdataProvider

## 1. XLXSDataProvider

package com.visionit.orangehrm.utilities;

import java.io.File;

import java.io.FileInputStream;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class XLXSDataProvider {

// load the workbook

XSSFWorkbook wb;

// create a constructor of a class

public XLXSDataProvider() {

try {

// load the file and create a reference of that object

File fs =new File("./TestData/testData.xlsx");

// read excel data

FileInputStream fins= new FileInputStream(fs);

// load the workbook

wb = new XSSFWorkbook(fins);

}

catch(Exception e) {

System.out.println("Excel file not found >>" +e.getMessage());

}

}

// create action data and read the string data from excel sheet

public String getStringCellData(String sheetname,int row,int col) {

return wb.getSheet(sheetname).getRow(row).getCell(col).getStringCellValue();

}

//read the integer data from excel sheet

public int getNumericCellData(String sheetname,int row,int col) {

return (int)wb.getSheet(sheetname).getRow(row).getCell(col).getNumericCellValue();

}

public String getStringCellData(int sheetIndex,int row,int col) {

return wb.getSheetAt(sheetIndex).getRow(row).getCell(col).getStringCellValue();

}

public int getNumericCellData(int sheetIndex,int row,int col) {

return (int)wb.getSheetAt(sheetIndex).getRow(row).getCell(col).getNumericCellValue();

}

}

2. Login\_Tc\_001

**package** com.visionit.orangehrm.testcases;

**import** org.testng.annotations.Test;

**import** com.visionit.orangehrm.pageObject.LoginPage;

**import** com.visionit.orangehrm.testBase.TestBase;

**public** **class** Login\_Tc\_001 **extends** TestBase {

@Test

**public** **void** loginOrangeHrmTest() {

LoginPage login = **new** LoginPage(*driver*);

// login.loginOrangeHrm(configData.getUserName(),configData.getUserPassword());

login.loginOrangeHrm(xlsxData.getStringCellData("login", 0, 0),

xlsxData.getStringCellData("login", 0, 1));

}

}

**3. TestBase**

package com.visionit.orangehrm.testBase;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.ie.InternetExplorerDriver;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.Optional;

import org.testng.annotations.Parameters;

import com.visionit.orangehrm.utilities.ConfigDataProvider;

import com.visionit.orangehrm.utilities.XLXSDataProvider;

public class TestBase {

public static WebDriver driver = null;

public ConfigDataProvider configData;

//create instance at class level

public XLXSDataProvider xlsxData;

@BeforeSuite

public void setupSuite() {

configData = new ConfigDataProvider();

xlsxData = new XLXSDataProvider();

}

@Parameters("Browser")

@BeforeMethod

public void setUp(@Optional("Chrome")String browserName) {

if(browserName.equals("Chrome")) {

// if we want to execute test cases on different browser

// set up the System variable path

// set the System variable path

// if you dont want to use system variable path you can download driver manager

// dependencies

// it will automatically check browser version.

System.setProperty("webdriver.chrome.driver", "./Drivers/chromedriver.exe");

// create the driver instance of chromedriver.

driver = new ChromeDriver();

// navigate to the application

} else if(browserName.equals("Firefox")) {

System.setProperty("webdriver.gecko.driver", "./Drivers/geckodriver.exe");

driver = new FirefoxDriver();

}else if(browserName.equals("IE")) {

System.setProperty("webdriver.ie.driver", "./Drivers/IEDriverServer.exe");

driver = new InternetExplorerDriver();

}

// set the System variable path

// if you dont want to use system variable path you can download driver manager

// dependencies

// it will automatically check browser version.

//driver.get(url);

driver.get(configData.getAppUrl());

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

}

@AfterMethod

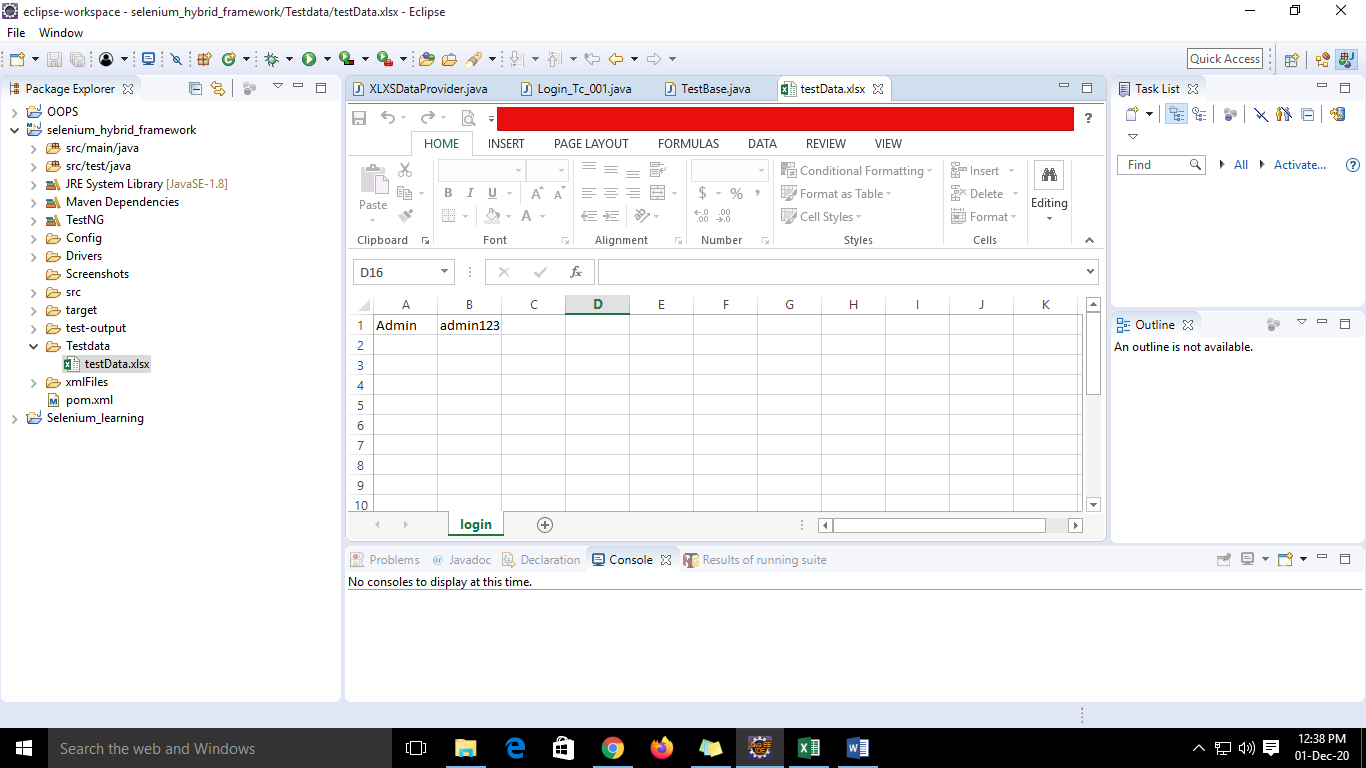
public void tearDown() {

// quit the browser

driver.quit();

}

}

4. testData.xlsx

5. LoginPage.java

package com.visionit.orangehrm.pageObject;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class LoginPage {

WebDriver driver;

// find a webelement using findby methods

@FindBy(name = "txtUsername")

WebElement username;

@FindBy(name = "txtPassword")

WebElement userpass;

@FindBy(id = "btnLogin")

WebElement loginBtn;

// create a web Element repository at page level

// create a login constructor..constructor name and login name must be same.

public LoginPage(WebDriver driver) {

this.driver = driver;

// By using page factory class we are going to initialise all webelements

PageFactory.initElements(driver, this);

}

// This is the action method

public void loginOrangeHrm(String uname,String upass) {

//username.sendKeys("Admin");

//userpass.sendKeys("admin123");

username.sendKeys(uname);

userpass.sendKeys(upass);

loginBtn.click();

}

}

6. ConfigDataProvider

package com.visionit.orangehrm.utilities;

import java.io.File;

import java.io.FileInputStream;

import java.util.Properties;

public class ConfigDataProvider {

public Properties prop;

// constructor

public ConfigDataProvider() {

try {

File fs = new File("./Config/config.properties");

FileInputStream fins = new FileInputStream(fs);

prop = new Properties();

prop.load(fins);

} catch (Exception e) {

e.printStackTrace();

}

}

public String getKeyValue(String searchKey) {

return prop.getProperty("searchKey");

}

public String getUserName() {

return prop.getProperty("username");

}

public String getUserPassword() {

return prop.getProperty("password");

}

public String getAppUrl() {

return prop.getProperty("url");

}

}

## New Session: 3Selenium WebDriver | Hybrid Framework part05

## Q.What is Data Driven Testing ?

## In data driven testing we keep data in excel sheet that is in external resources and fed to the application this will reduces the repetitive work and time and we can modify the data whenever required.

## In excel sheet we are storing the data in the form of frame, horizontally there are rows and vertical cell called columns and one row and one column is together called one frame.

## File name: OrangeHrmTest

**package** com.demoExtentReports;

**import** java.io.File;

**import** java.io.IOException;

**import** java.text.SimpleDateFormat;

**import** java.util.Date;

**import** java.util.concurrent.TimeUnit;

**import** org.openqa.selenium.By;

**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.ITestResult;

**import** org.testng.annotations.AfterMethod;

**import** org.testng.annotations.AfterTest;

**import** org.testng.annotations.BeforeMethod;

**import** org.testng.annotations.BeforeTest;

**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** OrangeHrmTest {

**public** ExtentHtmlReporter htmlReporter ;

**public** ExtentReports report;

**public** ExtentTest test;

**public** **static** WebDriver *driver*;

@BeforeTest

**public** **void** setupExtent() {

File fs =**new** File("./Reports/orange\_hrm.html");

htmlReporter = **new** ExtentHtmlReporter(fs);

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

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

htmlReporter.config().setTheme(Theme.***DARK***);

report =**new** ExtentReports();

report.attachReporter(htmlReporter);

report.setSystemInfo("HostName", "LocalHost");

report.setSystemInfo("OS", "Windows");

report.setSystemInfo("Tester Name", "Vasant");

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

}

@AfterTest

**public** **void** endReport() {

report.flush();

}

@BeforeMethod

**public** **void** setup() {

System.*setProperty*("webdriver.chrome.driver", "./Drivers/chromedriver.exe");

*driver*=**new** ChromeDriver();

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

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

*driver*.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

}

@Test

**public** **void** verifyTitleTest() {

test =report.createTest("verifyTitleTest");

String actualTitle =*driver*.getTitle();

String expectedTitle= "OrangeHRM";

Assert.*assertEquals*(actualTitle, expectedTitle);

}

@Test

**public** **void** verifyLogoTest() {

test =report.createTest("verifyLogoTest");

**boolean** status = *driver*.findElement(By.*xpath*("//div[@id='divLogo']/img")).isDisplayed();

Assert.*assertFalse*(status);

}

@AfterMethod

**public** **void** tearDown(ITestResult result) **throws** IOException {

**if** (result.getStatus()==ITestResult.***FAILURE*** ) {

test.log(Status.***FAIL***, "Test Case failed"+result.getName());

test.log(Status.***FAIL***, "Test Case failed"+result.getThrowable());

String screenshotPath= *captureScreenshot*(result.getName());

test.addScreenCaptureFromPath(screenshotPath);

}

**else** **if** (result.getStatus()==ITestResult.***SUCCESS***) {

test.log(Status.***PASS***, "Test Case Passed"+result.getName());

}

**else** **if** (result.getStatus()==ITestResult.***SKIP***) {

test.log(Status.***SKIP***, "Test Case Skipped"+result.getName());

}

*driver*.quit();

}

**public** **static** String captureScreenshot( String screenshotName) **throws** IOException {

String dateFormat= **new** SimpleDateFormat("yyyyMMddhhmmss").format(**new** Date());

TakesScreenshot ts =(TakesScreenshot)*driver*;

File srcFile=ts.getScreenshotAs(OutputType.***FILE***);

String screenshotPath="./Screenshots/"+screenshotName+dateFormat+".png";

File destFile=**new** File(screenshotPath);

FileHandler.*copy*(srcFile, destFile);

**return** screenshotPath;

}

}