06-MAR-2020

-----------

CROSS BROWSER TESTING USING TESTNG XML FILE

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

Whenever we execute script on different browsers to check the compatibility of the application is called Cross Browser Testing.

In general following are the Cross Browser validation test factors

-- GUI of the application

-- Functionalities of the application

-- Performance test factors

\*\* Where as using TestNG.xml file we can pass the Browser name on which browser test suite need to execute

EX01:

Create TestNg class to validate login functionality in Orange HRM application

by reading the data from Testng xml file

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

TestNg XML file code

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

<suite name="Parameterization">

<test name="Login validation">

<parameter name="uid" value="Admin"/>

<parameter name="pwd" value="admin123"/>

<classes>

<class name="testng.pack.OHRMLogin"/>

</classes>

</test>

</suite>

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

TestNG class

package testng.pack;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.Assert;

import org.testng.Reporter;

import org.testng.annotations.AfterClass;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.Parameters;

import org.testng.annotations.Test;

public class OHRMLogin {

WebDriver driver;

@BeforeClass

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();

}

@Test

@Parameters ({"uid", "pwd"})

public void adminLogin(String userId, String passwd) throws InterruptedException {

WebElement objUserName = driver.findElement(By.id("txtUsername"));

if (objUserName.isDisplayed()) {

Reporter.log("Username element found in Application and Script executing ....",true);

//to perform login operation

driver.findElement(By.id("txtUsername")).sendKeys(userId);

driver.findElement(By.id("txtPassword")).sendKeys(passwd);

driver.findElement(By.id("btnLogin")).click();

Thread.sleep(5000);

String pgTitle=driver.getTitle();

Assert.assertEquals(pgTitle, "OrangeHRM");

Reporter.log("Successful login operation");

}

else {

Reporter.log("Username Element not found");

}

}

@AfterClass

public void tearDown() {

driver.close();

}

}

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

Ex:02 : Create script to perform login operation in OrangeHRM application on

different browsers based on keywords (i.e. Browsers) given from testNG xml

file

TestNG xml file

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

<suite name="Cross Browser testing">

<test name="Login validation in Chrome browser">

<parameter name="testBrowser" value="Chrome"/>

<classes>

<class name="testng.pack.OHRMLogin"/>

</classes>

</test>

<test name="Login validation in FF browser">

<parameter name="testBrowser" value="FF"/>

<classes>

<class name="testng.pack.OHRMLogin"/>

</classes>

</test>

<test name="Login validation in IE browser">

<parameter name="testBrowser" value="IE"/>

<classes>

<class name="testng.pack.OHRMLogin"/>

</classes>

</test>

</suite>

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

TestNg class

package testng.pack;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.ie.InternetExplorerDriver;

import org.testng.Assert;

import org.testng.Reporter;

import org.testng.annotations.AfterClass;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.Parameters;

import org.testng.annotations.Test;

public class OHRMLogin {

WebDriver driver;

@BeforeClass

@Parameters ({"testBrowser"})

public void setUp(String myBrowser) {

switch (myBrowser) {

case "Chrome":

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

driver=new ChromeDriver();

System.out.println("suite executing on Chrome Browser");

break;

case "FF":

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

driver=new FirefoxDriver();

System.out.println("suite executing on Firefox Browser");

break;

case "IE":

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

driver=new InternetExplorerDriver();

System.out.println("suite executing on IE Browser");

break;

}

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

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

}

@Test

public void adminLogin() throws InterruptedException {

WebElement objUserName = driver.findElement(By.id("txtUsername"));

if (objUserName.isDisplayed()) {

Reporter.log("Username element found in Application and Script executing ....",true);

//to perform login operation

driver.findElement(By.id("txtUsername")).sendKeys("Admin");

driver.findElement(By.id("txtPassword")).sendKeys("admin123");

driver.findElement(By.id("btnLogin")).click();

Thread.sleep(5000);

String pgTitle=driver.getTitle();

Assert.assertEquals(pgTitle, "OrangeHRM");

Reporter.log("Successful login operation");

}

else {

Reporter.log("Username Element not found");

}

}

@AfterClass

public void tearDown() {

driver.close();

}

}

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

TESTNG - DATA PROVIDER

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

Data provider is a TestNG Annotation

-- Which is used to pass set of test data to the test case

\*\*\*It always returns a double dimensional object array

-- Data Provider annotation will have only one attribute 'name' which is an optional

In one class we can have more than one Data Provider annotation methods

To refer Data Provider annotation method we use "data provider" option

Syntax:

------

data provider= data provider name

Ex: for Data Provider annotation method

@Data Provider (name="Test Data") // Here we can give any name in place of "Test Data"

public Object[][]getData(){

Object[][]data=new Object[3][2];

data[0][0]="mercury";

data[0][1]="mercury";

data[1][0]="Admin";

data[1][1]="admin123";

data[2][0]="mercury";

data[2][1]="mercury";

return (data);

}

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

data

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

(0, 0) (0, 1)

mercury mercury

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

(1, 0) (1, 1)

Admin admin123

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

(2, 0) (2,1)

mercury mercury

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

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

Ex: Create TestNG class with to concatenate strings by taking the data

from DataProvider annotation method

package testng.pack;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

public class ConcatenateEx {

@Test (dataProvider="SampleData")

public void concateStrings(String a, String b) {

System.out.println(a+b);

}

@DataProvider (name="SampleData")

public Object[][] getData(){

Object[][] data= new Object[3][2];

data[0][0]= "Live";

data[0][1]="Tech";

data[1][0]= "Madhukar";

data[1][1]="QAIT";

data[2][0]= "IT";

data[2][1]="Jobs";

return (data);

}

}

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

Creating Reports in Selenium

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

WD (Web Driver) does not provide any library to generate test execution reports where as we can use some 3rd party APIs to generate reports

1. Using TestNg framework

2. Using log4j properties

3. Extent Reports v4

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

1. Using TestNg Framework

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

By default TestNg will generate html reports for each suite execution

Disadvantage:

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

It is not possible to customize

It will override the report

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

2. Apache Log4j (log4j. properties)

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

Log4j is an open source framework provided by Apache for Java Project

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

================================================================END OF CLASS======================================================================================