Cucumber-BDD Framework

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

cucumber is a framework which is used for implementing BDD (Behavior-

driven development)

In BDD automation programs are created based on the behavior of the

application based on the testdata

cucumber was initially implemented with Ruby later it was extended

to work on Java and C#

in Cucumber the automation programs are created based on a file called

feature file

In feature file the task which we need to automate is written in plain

english.

ex:

Go to the website- Mercury Tours

Enter username and password

click on login

validate the login successful or not

These English statements are connected to the selenium program which

perform those activities

The advantage of this process is the flow of the Automation programs can

be understand by non-technical person (i.e. Manual tester and client)

cucumber used its own language called as Gherkin. this works on Certain

annotations

Feature:

This represents the module or functionality that is under test

Scenario:

this represents the test case that is been automated in a particular feature

one feature can have multiple scenarios

generally the title of the test case is given as Scenario

Given: this represents the pre-condition of the test case

When: This represent the exact action that is performed in the test case

And: This represents any additional actions that should be performed on the test case

Then: This represents the outcome of the test case

But: This represents any negative condition that should be tested

Ex:

Feature: Addition

Scenario: Add two numbers

Given I have entered 50 into the calculator

And I have entered 70 into the calculator

When I press add

Then the result should be 120 in the screen

For every feature file we need to implement the steps in this case we have 4 steps

step defination for all those steps defined in the feature file, where internally

we are using our Selenium + java programs only

when we execute the feature file, It will

move to the java code under the step definition and execute the program....

Importent Annotations in JUnit:

@Test:

This Represents a Test case

@Before:

This is executed prior to each test cases in the currect class

Note:

it similar to @BeforeMethod in TestNg

@After:

This is executed after each test case in the current class. similar to

@AfterMethod in TestNg

@BeforeClass:

This is executed before each class in the current test suite

@AfterClass:

This is execued after each class in the current test suite

@Runwith: This used to specify what kind of program under execution

Ex:

It is a Test suite if cucumber program

Cucumber Configuration

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

Step 1: Install the plugin to Eclipse

Navigation:

"Help" menu in Eclipse

select "Eclipse Market place"

enter "cucumber Eclipse plugin" in "Find" editbox then click on "Go"

click on "Install"

accept the license agreement

click on "Finish"

"Install anyway" on warning window

Step 2: Create a project

Step 3: Configure Cucumber jar files to projects & sss.jar

a. Library folder (WD jars)

b. driver files(browser .exe files)

c. Cucumber jars

cobertura-2.1.1

cucumber-core-1.1.2

cucumber-java-1.1.2

cucumber-junit-1.1.2

cucumber-jvm-deps-1.0.3

cucumber-reporting-0.1.0

gherkin-2.12.1

Junit-4.11

mockito-all-2.0.2-beta

Step 4: Create a folder in the project (OHRM)

Step 5: Create a file. in the folder (given a name with extension as .features)

Note:

A deature file will be created in the folder with default Gherkins Annotations

Note:

In feature file we update the task to be automated written in simple English

using Gherkins annotations

Note:

if cucumber plugins are not installaed then we will get plain file

Write following keywords in Feature file

Feature: OrangeHRM Login functionality testing

Scenario: validating login functionality of Orange HRM application

Given: Open Firefox browser and navigate to OrangeHRM application

When: user enter valid username and password and click on Login button

Then: user should be able to successfully login and close the application

Step 6: Create a package in the project (com.stepDefinition)

Step 7: create a Java class in a package (TestRunner)

Step 8: Run/Execute the current class (i.e. TestRunner) as JUnit Test

Ex:

package com.stepDefinitation;

import org.junit.runner.RunWith;

import cucumber.api.junit.Cucumber;

@RunWith (Cucumber.class)

//to the features give the folder name with features file is created

@Cucumber.Options (features="Feature")

public class TestRunner {

}

Step 9: Create a package in the project (com.OrangeHRM)

Step 10: create a Java class (LoginTest)

Step 11: Copy the user defined methods created out of execution of the

test runner class from the eclipse output console and paste it is the current class

Step 12: develop the selenium automation test scripts for the methods

Ref:https://www.youtube.com/watch?v=Vv7hHrVX\_Zg

ref2: https://www.youtube.com/watch?v=vHzMJuc9Zuk

ref3:https://www.softwaretestinghelp.com/cucumber-bdd-tool-selenium-tutorial-30/