Cucumber framework

**What is Cucumber?**

Cucumber is a testing approach which supports Behavior Driven Development (BDD). It explains the behavior of the application in a simple English text using Gherkin language. Cucumber makes **it easy to read and to understand the application flow.**Cucumber BDD framework also **benefits the client to understand the application code** as it uses Gherkin language which is in Plain Text. Initially, Cucumber was implemented in Ruby and then extended to Java framework. it support **native JUnit.**

Behavior Driven Development is an extension of Test Driven Development and it is used to test the **system** rather than testing the particular piece of code.

**Jar files For Cucumber :**

* Cucumber-core
* Cucumber-html
* cobertura code coverage
* Cucumber-java
* Cucumber-junit
* Cucumber-jvm-deps
* Cucumber-reporting
* Hamcrest-core
* Gherkin
* Junit
* Selenium java
* Junit

### **Cucumber Basics**

**1) Feature Files:**

Feature files are the essential part of cucumber which is used to write test automation steps or acceptance tests. This can be used as the live document. The steps are the application specification. All the feature files end with **.feature extension.**

**EX: Sample feature file:**

**Feature**: Login Functionality Feature In order to ensure Login Functionality works, I want to run the cucumber test to verify it is working

**Scenario**: Login Functionality

**Given** user navigates to SOFTWARETETINGHELP.COM  
**When** user logs in using Username as “USER” and Password “PASSWORD”  
**Then** login should be successful

Each scenario should follow **given, when and then** format. This language is called as “**gherkin**”.

1. **Given:** given specifies the pre-conditions. It is basically a known state.
2. **When**: This is used when some action is to be performed.
3. **Then:**The expected outcome or result should be placed here.
4. **Background:**Whenever any step is required to perform in each scenario then those steps need to be placed in Background. For Instance: If a user needs to clear database before each scenario then those steps can be put in a background.
5. **And**: And is used to combine two or more same type of action.

**2) StepDefination** : step defination ek java class hai jo ki feature file ke sath connected hota hai jisme hum selenium, java, tags ka use karte hai

**3) JUnit Runner:**

To run the specific feature file cucumber uses standard JUnit Runner and specify tags in **@Cucumber. Options**. Multiple tags can be given by using comma separate. Here you can specify the path of the report and type of report you want to generate. Test runner ek java class hota hai jise run karne ke liye hum junit ka use karte hai and hume output ka report genarate karke deta hai

**Example of Junit Runner:**

|  |
| --- |
| **import** cucumber.api.CucumberOptions;  **import** cucumber.api.junit.Cucumber;  **import** org.junit.runner.RunWith;  @RunWith(Cucumber.**class**)  @Cucumber.Options(format={"SimpleHtmlReport:report/smokeTest.html"},tags={"@smokeTest"})  Public **class** JUnitRunner {  } |

1. HTML form me report chahiye to hum **format ={“pretty”, “html:test-outputreport”}**
2. package cucumberTest;



1. import org.junit.runner.RunWith;
2. import cucumber.api.CucumberOptions;
3. import cucumber.api.junit.Cucumber;

@RunWith(Cucumber.class)

@CucumberOptions(

features = "Feature"

,glue={"stepDefinition"}

)

public class TestRunner {

}

Jisse humare project me “test-outputreport” name ka folder ban jayega and voh hume **html** report dega

**4) Scenario Outline:**

Scenario outlines are used when the same test has to be performed with different data set. Let’s take the same example. We have to test login functionality with multiple different sets of username and password.

**Scenario Outline**: Login Functionality

**Given** user navigates to SOFTWARETESTINGHELP.COM  
**When** user logs in using Username as <**username**> and Password <**password**>  
**Then** login should be successful

**Examples:**  
|username |password          |  
|Tom                     |password1        |  
|Harry                   |password2        |  
|Jerry                    |password3        |

Vertical pipes are used to separate two different columns. An example can contain many different columns.

**5) Tags:**

Cucumber by default runs all scenarios in all the feature files. In real time projects, there could be hundreds of feature file which are not required to run at all times.

### **Cucumber Options available**

**Following Options are use in the step Runnuer class**

**1)dryRun :** iski value true ya false hota hai and yeh check karta hai ki feature file ke har ek line ka step defination likha hai ya nahi one to one mapping check karta hai and script run hone se pahle hume agar koi missing step defination hai to console pe bata deta hai

And script ko run karne ke liye uski value humsa false hona chahiye

So pehle hum iski value true likh kar check karege ki koi step defination miss to nahi hai n

Fir uski value false likhte hai

**2)Features :**yaha hum feature file ka path dete hai

**3)Glue():** yaha hum step defination ke package ka name dete hai

**4)Tags:** {“@smoke”, “@Regression”}

Tags ki help se hume jo test cases ko run karvana hota hai sirf vahi testcases ko run karva skte hai

Tags ka uses humesa feature file me kiya jata hai

Agar muje ek sath do se jada tags ko run karvana ho to hum uss “,” ki helps se sabhi tags ko likhege example {“@smoke, @Regression”}

Agar mujhe dono common vale tags run karvana ho to hum {“@smoke”, “@Regression”} likhege

Agar mujhe kisi tags ko ignore karna hai to **“~”** iska use karte hai {“~@smoke”, “@Regression”}

**5)Monochrome:** iski value true dege to jo conscole par output aata hai voh kafi achhe readable formate me ho aayega | agar false de duga to readable formate me console par output nahi aayega

**6)Format:** agar output json ,html and “junit xml” form me chahiye to hum is ka use karte hai

**7)Strict:**iski bhi value true ya false hota hai agar true likhege to jab run hoga script to yeh hume bata dega ki koun sa step stepdefination file me nahi likha hai

### **DATA DRIVEN INCUCUMBER**

1)Simple Data Driven(without Examples keyword)

2) Simple Data Driven(with Examples keyword + Scanario outlines)

3) Simple Data Driven(By using tables)

**1)Simple Data Driven(without Examples keyword)**

**Ex:** driver.findElement(by.id(“name”)).sendKeys(“Name”);-> simple selenium code

**In the feature file :** When user enters “Naveen” and “Admin@123”

**In the StepDefination file** : public void userName\_and\_Password(String username, String password)

{

driver.findElement(by.id(“name”)).sendKeys(username);

driver.findElement(by.id(“password”)).sendKeys(password);

}

**2) Simple Data Driven(with Examples keyword + Scanario outlines)**

**Scenario Outline: login to the application**

**In the feature file :** When user enters “<username>” and “<password>”

Examples : |username|password|

|Rishikesh|Admin@123|

|Shivani|Mishra@123|

|Babuni|Neeta@123|

**NOTE: Yaha three window open hoga because yaha humne three userid and password liya hai**

**In the StepDefination file** : public void userName\_and\_Password(String username, String password)

{

driver.findElement(by.id(“name”)).sendKeys(username);

driver.findElement(by.id(“password”)).sendKeys(password);

}

**3) Simple Data Driven(By using tables)**

Scanario outlines keyword ka use hum yaha nahi karege voh jum hum examples keyword ka use karte hai tab likhte hai

**In the feature file :** When user enters username and password

|Rishikesh|Admin@123|

**In the StepDefination file** : public void user\_enters\_uName\_and\_psw(DataTable credentials)

{

List<List<String>> data= credentials.raw();

driver.findElement(by.id(“name”)).sendKeys(data.get(0).get(0));

driver.findElement(by.id(“password”)).sendKeys(data.get(0).get(1));

}

**Note: yaha feature file me jis line me data chahiye uske turnt baad vale line me hum data likhte hai Stepdefination vale class me “DataTable” predefined keyword hai uska object bana kar hum use karte hai. Yaha par hum row and column ka concept use karke data fatch karte hai**

**Lecture no. 5 ko sumjane ke liye Map ka concept aana chahiye**

**List of depandancys**

<dependency>

<groupId>info.cukes</groupId>

<artifactId>cucumber-java</artifactId>

<version>1.2.5</version>

</dependency>

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

<dependency>

<groupId>info.cukes</groupId>

<artifactId>cucumber-junit</artifactId>

<version>1.2.5</version>

</dependency>

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

<dependency>

<groupId>info.cukes</groupId>

<artifactId>cucumber-jvm-deps</artifactId>

<version>1.0.5</version>

<scope>provided</scope>

</dependency>

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

<dependency>

<groupId>net.masterthought</groupId>

<artifactId>cucumber-reporting</artifactId>

<version>5.0.2</version>

</dependency>

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

<dependency>

<groupId>info.cukes</groupId>

<artifactId>gherkin</artifactId>

<version>2.12.2</version

<scope>provided</scope>

</dependency>

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

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13</version>

</dependency>

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

<dependency>

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

<artifactId>selenium-java</artifactId>

<version>3.141.59</version>

</dependency>

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