What is Cucumber?

Cucumber is software tool that supports Behaviour-Driven-Development.

It runs automated acceptance tests written in a behavior-driven development (BDD) style.

Cucumber reads **executable specifications** written in **plain text** and **validate** that the **software does what those specifications say**.

The **specifications** **consist** of **multiple example or scenarios**.

**Each scenario** is a **list of steps** for **cucumber** to **work through**. Cucumber **verifies** that the **software conforms** with the **specification** and **generate** a **report indicating ✅ success or ❌ failure** **for each scenario**.

In order for cucumber to understand the scenarios, they must follow some basic syntax rules called Gherkin.

Gherkin documents are stored in .feature text files.

*3 popular BDD testing tools*

* Cucumber - Cucumber is a Ruby based framework
* JBehave - JBehave is a JAVA based framework
* Specflow ­- Specflow is a .NET based framework

*Advantages of Cucumber*

* Cucumber acceptance tests are written from user’s perspective
* Team Collaboration (Product owners, business analysts, developers, testers are all involved in writing scenarios)
* Code reusability
* Tests are written in plain-text English so people with less technical skills can also write scenarios
* Cucumber can be integrated with Selenium and other testing frameworks like JUnit & TestNG

*The two files required to run a cucumber test.*

* Feature file
* Step Definition file

*What is a feature file?*

Feature file is an entry point of Cucumber execution. Cucumber scenarios are written inside the feature file using a plain-text language called Gherkin. It can contain a scenario or multiple scenarios. All the acceptance tests are written in feature file using Given-When-Then statements.

*What is the two main purpose of using Gherkin?*

* Documentation
* Automated tests

*. What are the keywords used in Feature file?*

* Feature
* Background
* Scenario
* Scenario Outline
* Given
* When
* Then
* And
* But

*Is it mandatory to use Given, When, Then keywords while writing scenario?*

No.  \*  can also be used to write steps in the feature file.

*What is the difference between Given, When, Then steps in feature file?*

* Given defines the context of the scenario
* When defines the actions of the scenario
* Then defines the outcome of the scenario

*Explain Scenario Outline in feature file*

Scenario Outline keyword in feature file is used to execute scenarios multiple times using a different set of test data. Multiple sets of test data are provided by using ‘Examples’ in a tabular structure separated by pipes (| |)

*Explain background in feature file*

Steps written under the Background keyword are executed before every scenario.

*How to comment a line in Feature file?*

# is used to comment a line in feature file

*What is a Step definition file?*

Step definitions file has the actual code implementation of the scenario step.

*Explain Cucumber Tags*

Cucumber tags are used to organize scenarios in your feature file. You can have as many tags as you like before a scenario or feature. @ is used to represent tags. Example: @regression, @sprint5, @EndtoEnd

Tags are used to

* Group scenarios
* Ignore scenarios from execution
* Logically group (OR & AND)

***Cucumber Tags are case sensitive. True or False?***True

*Name any two testing framework that can be integrated with Cucumber?*

* JUnit
* TestNG

*Name any two build management tools that can be integrated with Cucumber?*

* Gradle
* Maven

*What software do you need to run cucumber in JAVA?*

* Eclipse or IntelliJ IDE
* Gradle or Maven build tool
* Junit or TestNG testing framework
* Cucumber
* Selenium (To automate browser)

***How does a JUnit Test Runner class look like?***

@RunWith (Cucumber.class)  
@CucumberOptions(features = "src/test", *//// set: the path of the feature files -- { }* tags={"@sort"}, */// to instruct what tags in the feature files should be executed -- { }* dryRun = false, */// true: checks if all the steps have step definition -- default false* strict = true, */// true : will fail execution if there are undefined or pending steps -- default false* plugin = "json: cath.json",  
 monochrome = true,format = {"html:src/test/java/cucReports" ,"json:src/test/java/cucReports/cath1.json","pretty"})  
 *//format = {"pretty"})  
 /// glue -- set: the path of the step definition files --- {}  
 /// monochrome -- true : display the console output in much readable way -- default false  
 /// format -- set: what all report formaters to use -- default false*

public class CucumberTestRunner {  
  
}

*What is @CucumberOptions in test runner? List the properties of @CucumberOptions*

@CucumberOptions are used to set specific properties for your cucumber test.

Properties are,

* Feature – path to feature file
* Glue – path to step definition
* dryRun – boolean value – check for missing step definition
* tags – used to group cucumber scenarios in the feature file
* strict – boolean value – fail the execution if there is a missing step
* monochrome – boolean value – display console output in a readable way

*Name any advanced framework design that can be used with Cucumber?*

* Page Object Model
* Log4j
* Extent Reporting
* Dependency Injection (Example: Pico Container)
* Object Repository

*Selenium can be integrated with Cucumber. True or False?  True*

*Explain Cucumber Hooks*

Cucumber Hooks are blocks of code that can be used to run before and after the scenarios using @before and @after methods. It helps us eliminates the redundant code steps that we write for every scenario and also manages our code workflow.

***Note: @After will be executed even though the scenario is failed. @Before will be executed before ‘Background’ steps in the feature file.***

@smoke @regression

Feature: Login functionality

Background:

Scenario Outline: Valid username and password

Given As a end user I am on login page

When user enter <username> and “<password>”

Then user should land on homepage

Examples

| username | password !

! [ballachakri@yahoo.com](mailto:ballachakri@yahoo.com) ! Khalifa12 |

Step Definitions

In addition to feature files, Cucumber needs a set of Step definitions. Step definitions map each Gherkins step into runnable programming code to carry out what action should be performed by the step.

Step definitions can be written in many programming languages.

[ Cucumber was originally written in the Ruby programming languagemnhhyrf6u95310 ].

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

To create Page Object Model Framework: using Cucumber BDD format

* New Maven project with QuickStart.
* Add below three main dependencies.

*<!-- info.cukes cucumber -java dependency -->* <dependency>  
 <groupId>info.cukes</groupId>  
 <artifactId>cucumber-java</artifactId>  
 <version>1.2.5</version>  
 </dependency>  
*<!-- info.cukes cucumber -junit dependency -->* <dependency>  
 <groupId>info.cukes</groupId>  
 <artifactId>cucumber-junit</artifactId>  
 <version>1.2.5</version>  
 </dependency>  
*<!-- seleniumhq selenium java -->* <dependency>  
 <groupId>org.seleniumhq.selenium</groupId>  
 <artifactId>selenium-java</artifactId>  
 <version>3.14.0</version>  
 </dependency>

* Create a resources Package in src/test/java folder for feature files
* Create another Package src/test/java folder for stepdefinitions

To run test we need to create CucumberTestRunner class

@RunWith (Cucumber.class)  
@CucumberOptions(features = "src/test", *//// set: the path of the feature files -- { }* tags={"@sort"}, */// to instruct what tags in the feature files should be executed -- { }* dryRun = false, */// true: checks if all the steps have step definition -- default false* strict = true, */// true : will fail execution if there are undefined or pending steps -- default false* plugin = "json: cath.json",  
 monochrome = true,format = {"html:src/test/java/cucReports" ,"json:src/test/java/cucReports/cath1.json","pretty"})  
 *//format = {"pretty"})  
 /// glue -- set: the path of the step definition files --- {}  
 /// monochrome -- true : display the console output in much readable way -- default false  
 /// format -- set: what all report formaters to use -- default false*

public class CucumberTestRunner {  
  
}