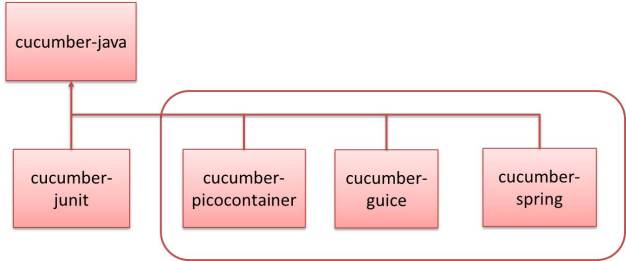
**BDD with Cucumber-JVM**

**Express the behavior of system**

Cucumber-jvm is an open source BDD tool that lets user express the behavior of system under test in plain language like English, French, German etc. It is a great way to have an executable documentation of your system.

**Required Maven Dependencies**

[](https://testingneeds.files.wordpress.com/2015/10/picture1.jpg)

*info.cukes*

*cucumber-picocontainer*

*${cucumber.version}*

*info.cukes*

*cucumber-junit*

*${cucumber.version}*

**Gherkin Keywords**

Gherkin is the language that Cucumber understands.G*herkin* is primarily used to write ***structured***tests which can later be used as project documentation. The property of being *structured* gives us the ability to automate them.

**Scenarios – Given When Then**

We can define scenarios in given, when and then format.

**Example-**

Given a customer added one Item in shopping cart.  
When the customer adds same Item in shopping cart.  
Then the customer should be warned that same item is already in yours shopping cart.

**Writing Scenario’s in Cucumber**

• Scenarios are organized together into features.  
• Each feature is represented as a plain text file.  
• Feature files must have the “.feature” extension.  
• Each feature can contain many scenarios.

**Step Definitions**

• A single Method representing a Step from a Feature file.  
• The Method is annotated with information about the step it represents.  
• Cucumber generates “Code Snippet’s” for Step Definition it can’t find.

**Step Annotations**

*import cucumber.annotation.en.\*;*

*Each Gherkin step keyword has a Annotation.*

*@When("^we want to stop traffic$")*

*public void methodName() {*

*..........*

*}*

**Passing and Failing**

Each step has it own result state.  
– Passed  
– Pending  
– Skipped  
– Failed  
Individual step result state combined determine the Scenarios success or failure.  
Steps can be failed on exceptions  or JUnit Assertion failures.

**Regex Lesson**

Cucumber uses Regular Expressions to match steps to definitions.

*@When("^we want to stop traffic$")*

*^   Matches the starting position*

$   Matches the ending position

**Capturing Arguments**

The pair of brackets is a capture group  
Everything captured is passed to method parameter  
Cucumber will automatically convert to the method parameter type  
Equal number of method parameters and capture groups

@Given("^an owner with a pet called \"Browny\"$")

public void an\_owner\_with\_a\_pet\_called() {

......

}

@Given("^an owner with a pet called .\*$")

public void an\_owner\_with\_a\_pet\_called(String name) {

......

}

**Type Conversion**

If the captured string is not in the correct format for type conversion an error will be thrown.  
Use regular expressions to capture the correct format string for the type.  
**Note:** Escape regular expression characters with backslash.

**Typical Capture Groups**

* (.+) Good for capturing strings and dates
* \”([^\”]\*)\” Captures a string within Double quotes
* (\\d) Capture decimal numbers only
* (\\d+\\.\\d+) Capture floating point numbers
* (true|false) Captures a Boolean value

**Date Formats**

@Given ("^today is (.+)$")

public void today\_is(Date date) {

......

}

Dates are recognized if it’s a format from the DateFormat class.

* SHORT is completely numeric, such as 12.13.52 or 3:30pm.
* MEDIUM is longer, such as Jan 12, 1952.
* LONG is longer, such as January 12, 1952 or 3:30:32pm.
* FULL is pretty completely specified, such as Tuesday, April 12, 1952 AD or 3:30:42pm PST.

**Example-**

**import** cucumber.DateFormat**;**

@Given ("^today is (.+)$")

public void today\_is(@DateFormat ("yyyy-MM-dd") Date date) {

......

}

**Tags**

The tags can be used when specifying what tests to run through any of the running mechanism.  
So tag is uses for grouping Scenarios together

* Identify slow running test that shouldn’t be run frequently.
* Identify tests that require a particular library or server.

So we can add tags in our feature file-

@feature

Feature: TestSample

@one @three

Scenario: Example

Given .....

When ......

Then ......

@one

Scenario: Example Two

Given .....

When ......

Then ......

@three

Scenario: Example Three

Given .....

When ......

Then ......

@ignore

Scenario: Example Four

Given .....

When ......

Then ......

For the better understanding of tags and there execution please view [@CucumberOptions.](https://testingneeds.wordpress.com/2015/09/15/junit-runner-with-cucumberoptions/)