Selenium JBehave Framework

In this POC, I have tried to explore the Selenium-JBehave frame work.

As per the introduction by JBehave team, *JBehave is a framework for Behaviour-Driven Development (BDD). BDD is an evolution of test-driven development (TDD) and acceptance-test driven design, and is intended to make these practices more accessible and intuitive to newcomers and experts alike. It shifts the vocabulary from being test-based to behaviour-based, and positions itself as a design philosophy.*

So trying to put this in simple terms, from a TDD’s approach we as testers and other SME’s try to develop stories (correlate with a feature to test) by writing the scenarios (Correlate with test case) which contains the building blocks called steps (Correlate with test steps available in the test case) . When the stories are written or the behaviour of the application is depicted, other teams like Development team will start implementing those behaviours. In this example I am just concentrating on the Test automation part of the TDD.

The stories we have written are mapped to the java action classes, which will carry out the user actions specified in the test case. To start with, below are the two scenarios I have worked in this example

1. Scenario 1: User lands into the Flipkart site. And goes to the customer care page and selects and chooses a question from the available list under the “Offer Redemption” section. Have taken two examples in this.
   1. One a positive scenario, selecting available from the list and when user lands on the next page, I am doing a comparison that the appeared question text is expected or not.
   2. Negative scenario, trying to select a question which is not present on the page. Expecting the test case to fail and report the error
2. Scenario 2: User logs into Flipkart site. Searches for “Timex watches” .Picks a result from the search results and views that item.

We will see them in detail .

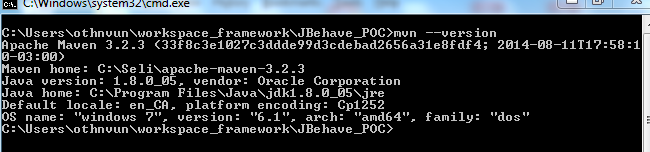
# Prerequisites ::

# Before we start running the stories ,

1. Make sure java is installed in your machine. my jdk version is jdk1.8.0\_05
2. System should be configured with Maven. In order to configure Maven please go through the below link

<http://maven.apache.org/guides/getting-started/maven-in-five-minutes.html>

check the maven version :: mvn –version( my current version)



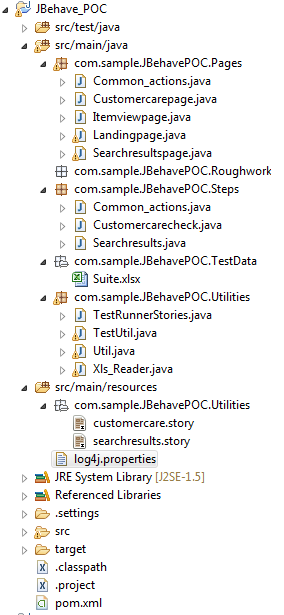
1. Under the C drive create a log file under C:\Jbehavelogs\application.log
2. Should have knowledge on Selenium WebDriver
3. Should have basic knowledge on Hybrid framework and Page object models

# Framework Components::

These are the Five steps involved

1. Write the Test Stories
2. Map the test steps to Java
3. Configure the Stories
4. Run stories
5. View Reports

Below is the high level overview of project structure



We have all the artifacts of the project under two folders one src/main/java and other one src/main/resources

## Src/main/Java

We will just see what are under src/main/java::

## com.sample.JBehavePOC.Pages::

All the pages we encounter during the test flows are maintained under this package .In the two scenarios I have, test flow goes through the flipkart’s

* Landing page
* Customercare page
* Searchresults page
* Itemview page

Sceanario1 Flow : Landing page->Customer care page

Sceanrio2 Flow: Landing page->Searchresults page->Item View page

Each page will have user specific functions or action actions that are needed to be performed. All these actions are captured as methods in these pages. Go through the Page java files available.

## com.sample.JBehavePOC.Steps::

The Steps written in the Stories are mapped in the files under this section. Its easy to maintain a separate Steps file for each story you have written. In my case I have two stories customercare and searchresults. I made two step files for them. And for all the common actions each story can contain. I have created a Common\_actions steps file. It contains actions like, user opening specific browser or closing the browser after the test etc. It also contains the actions user needs to perform before starting the story/scenario. And action that are needed to be performed after the story/scenarios are finished.

Stories written will be mapped to the corresponding java , using the JBehave annotations. Some of the notations used in these steps files are

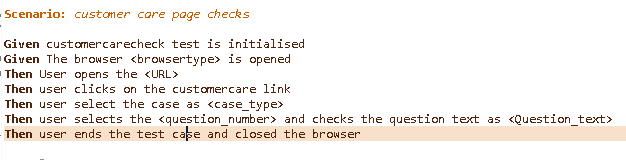
**@When**

**@Given**

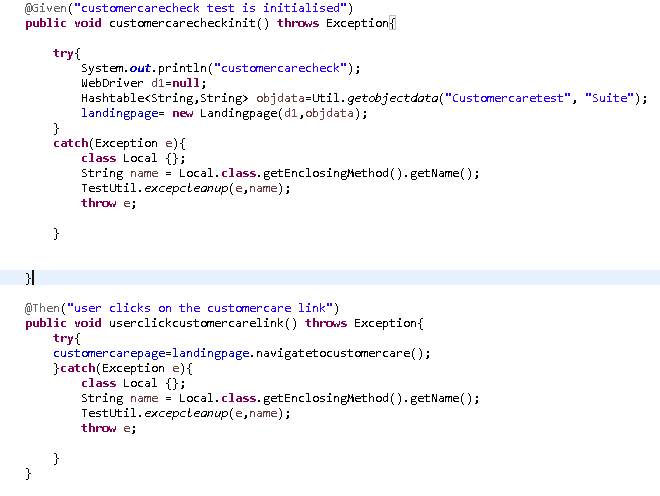
**@Then**

Please look at the JBehave annotations for more info

**Note**::Provided a sample Story and the corresponding step mapping.



Step to Code mapping ::



## com.sample.JBehavePOC.Utilities

Under this I have placed all the commonly used code.

### TestUtil ::

File what I have here, contains all the selenium layer of code. I have created the commonly used functions click, type, select, openbrowser, navigate , function to take the screenshot etc.. all the selenium web Driver code is available in this

### Util::

This file is maintained , to fetch the object identifiers of the elements( Used Xpaths to identify the element). When we use the getobjectdata function part of this file, we will get a Hashtable<String,String> with key as object name and value as object xpath, for the current running test story.

### Xls\_Reader:

File with functions to read the data from excel . ( reading the data from excel using the poi api’s. Configurations related to these are available in the pom.xml of this project )

### TestRunnerStories\*\*\*\*\*\*\*::

This is the main file, where JBehave is going to start the execution. We can have the name of this file as you want, but make sure file name should end with “Stories”. In this file we make the configurations required.

There are different ways JBehave runs the test flows, the way we are using here is running the stories as “embeddables”.

On a high we needs to do the following changes in this file

1. **Configuration**:: We are just going to use the default configuration JBehave as given . We are going to use MostUsefulConfiguration setup provided by JBehave. Check the *configuration()* in the file.
2. **Steps**:: The steps files we have written in this project should be notified to the JBehave controller. Check *the stepsFactory()* in the file . In this example, the step files I have developed for the stories are Customercarecheck, Searchresults,Common\_actions. I have included these three in the list.
3. **Storypaths**:: We specify where are the stories located in the project . And what stories we are going to run. Please look at the *storyPaths().*

**Note** :: I have provided the story location in the relative path way like “\*\*/\*.story” in the included list . So in this case all the .story files available under the project are configure to run. It you want specific test story to run . Please modify according ( eg: “\*\*/customercare.story”)

## com.sample.JBehavePOC.TestData

An excel file with all the Object related identifiers is maintained here. Make sure the name of the excel is Suite. And under the sheet “ObjectData” we specify all the Object identifiers. Names should be proper given, as these drive the selenium actions. Object properties of all the stories can be given in this sheet. Make sure we have blank row , between each test related data. And followed by the story name for which the test data is starting .Example is provided below.

### Where exactly we are getting the test data:::

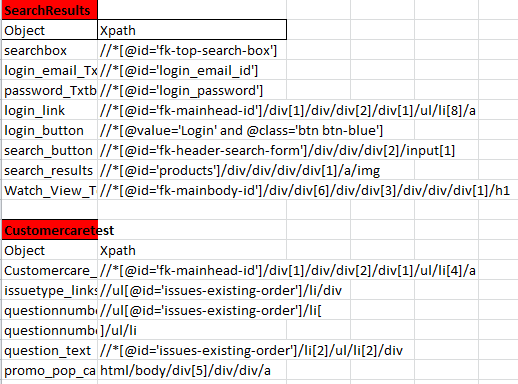
In all the stories I have written , the first step is something like “**Given** customercarecheck test is initialised” . This step I just have to initialise my runner with required test data. If you check the actual implementation of this step under the “Customercarecheck.java” under steps package we has the below actions.

WebDriver d1=**null**;

( Creating a webdriver instance for this story)

Hashtable<String,String> objdata=Util.*getobjectdata*("Customercaretest", "Suite");

(Getting the test data of “Customercaretest” from the excel “Suite”, we have named the test data excel as Suite. Make changes accordingly if you need)



## src/main/resources

We will just see what are under src/main/resources

### Story file::

Under the same package “TestRunnerStories.java” file exists, we are going to have all our story files. Check the project structure, we have the story files under com.sample.JBehavePOC.Utilities

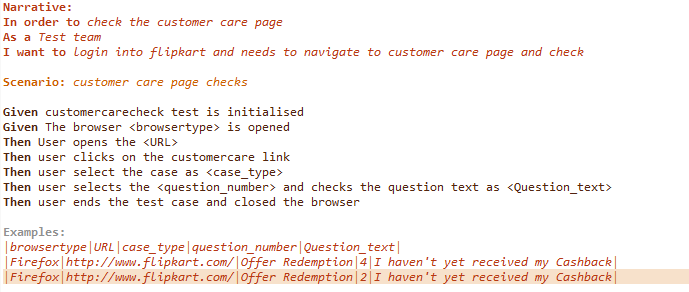
customercare.story::

We have two instances of the story , one the positive scenario and other one with a negative scenario , I am expecting the test case to fail and repot the error to you .

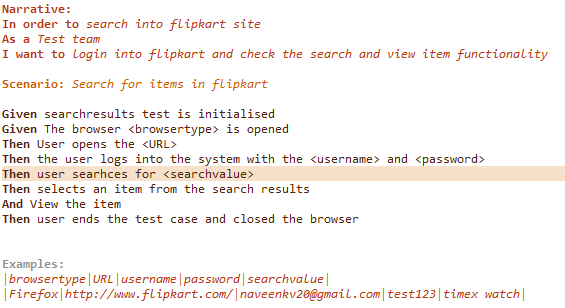
For Parameterising the test data, JBehave has the “Examples” module. You can see in the test steps, the parameter that is required to be parameterised is captured in “<>” and same is given as the column name in the Examples:: section . When we have these kind of Parameterised variables in test steps, the corresponding step mapping should also reflect the same. Given an example below

@Given("The browser <browsertype> is opened")

**public** **void** browseropen(@Named("browsertype") String browsertype) **throws**



Searchresults.story

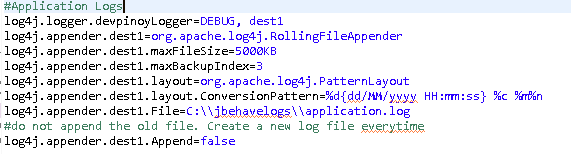


### Log4J Properties file ::

Under src/main/resources folder we should have the log4j.properties file , which we use for Logging purpose.

Before running the test case , make sure you create a log file C:\Jbehavelogs\application.log

Below are the values we should have



## Exception handling::

Throughout the this framework, my intention is when a story or a step in story is failed, execution should not stop at that story .It should continue with the next story. JBehave framework automatically does that handling the next story part. So I have used that and in addition to that I am trying to clean up the remaining work products of failed test case. For example we are opening a browser in each story, when story fails, execution is taking the controller to next story, but the browsers opened by previous case not closed by JBehave. So I am trying to handle them manually

Along with this, I am trying to take a snapshot of error page and capture the exception to the logger file we maintain.

All through this implementation, you can see them in try catch blocks available in the test step files.

## Maven Dependencies::

All the build and deployment in this frame work is through Maven . So the project I have shared contains the pom.xml , which contains the dependencies that are required

JBehave Dependency,

Selenium Dependency,

POI api Dependency,

Log4j Dependency.

It also have the jbehave-maven plugin through which we run the test cases.

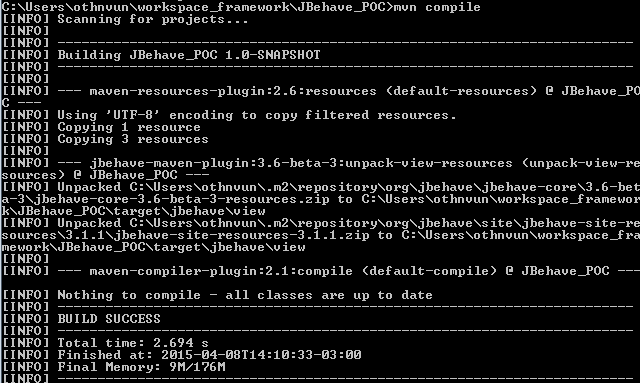
Get the pom.xml from the project code.

## How to Run the test cases ::

1. Download the project from

<https://github.com/naveenv20/JBehave_POC>

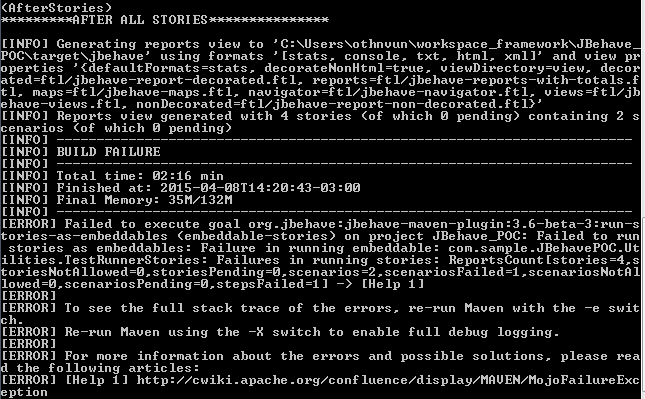
1. From command prompt, navigate to the project location ( till the location where the pom.xml is present) . In my case the project is under C:\Users\othnvun\workspace\_framework\JBehave\_POC
2. Make sure all the test data items are properly configured
3. In all the Steps files , make sure we pass the right test case name and excel sheet name for fetching the right set of object data
4. In the TestRunnerStories.java file, I have given the included test cases as a very generic one “\*\*/\*.story” , so all the files with .story will be executed.
5. Give the command “mvn compile”. Check for errors.



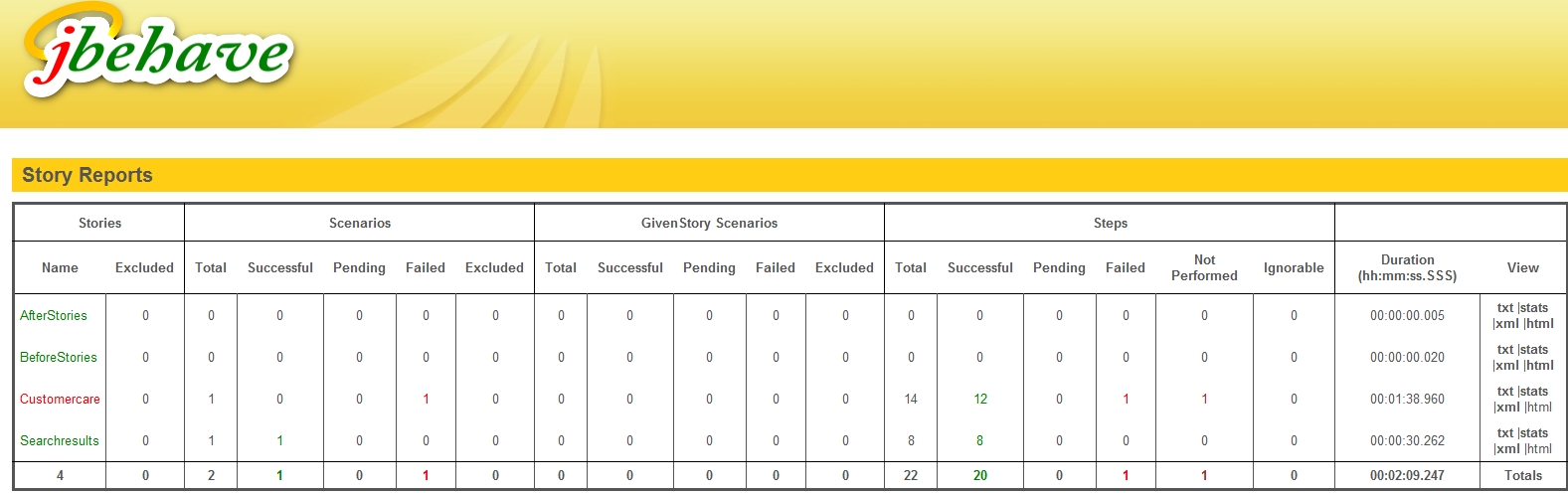
My build is successful. For the first time when we give this, all the dependency files might get downloaded from maven repositories. So it will take a bit of time

1. When you build is successful, give the command “mvn clean integration-test”

When all story are finished, you should see the execution status something similar to the below one



1. For checking the results , under the project folder , navigate to target\jbehave\view and check the reports.html



We can see our stories Customercare and Searchresults appearing over there .Pick the html version of each story to get the detailed report

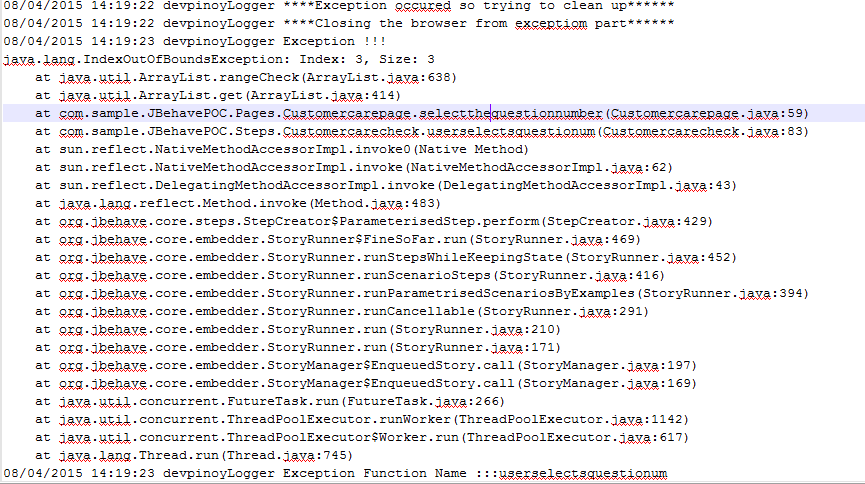
Customercare story::



Searchresults story::



1. You can see as expected under the customercare story one scenario has passed and other is failed .It has shown us in what story step it has failed and what’s the reason for this. We have 3 links available on the page and we are trying to access the fourth link . So the test case failed quoting the same .
2. Searchresults test case has passed.
3. For getting the error or exception what has failed the customercare test navigate to Logger file and check for the exceptions



1. As per the exception handling mechanism I have in this framework, we are going to obtain the screenshot the page , in which the exception or the error has occurred. Navigate to target\screenshots. And check for the page snapshots.
2. If you want to import the Project into eclipse for further modifications , then use the following commands
3. mvn eclipse:eclipse
4. when the build is successful
5. Then got to File ->import->General->Existing Projects into Workspace
6. Pick the project you have downloaded and finish .
7. Your project will be imported into eclipse where you can run
8. When we change any of your dependencies , make sure you use the above command mvn eclipse:eclipse and refresh the project once to observe the required changes

## Enhancements that can be done::

1. As per the example given in the JBehave site, they have used the Spring framework with this. In this example i haven’t used that, I kept it simple. So we can check the possibility of improving the framework with spring.
2. Used the basic configuration type in this example. Only enhancement I have in this is reporting types we can have. I have specified I need the report formats in *CONSOLE, TXT, HTML, XML.* Apart from getting the required reports. There are huge things we can do with these configurations. JBehave has number of other changes like , how we do we specify parameters name etc… check the JBehave site for this
3. Screenshot function I have developed can be enhanced. Currently to get unique name for file, I have used the time variable. So the enhancements could be appending the step number and method name in the file name
4. We can check the possibility of running the test stories in parallel.

## References::

<https://blog.codecentric.de/en/2011/03/automated-acceptance-testing-using-jbehave/>

<https://blog.codecentric.de/en/2012/06/jbehave-configuration-tutorial/>

<http://behaviourdriven.org/>

<http://en.wikipedia.org/wiki/Test-driven_development>

<http://fazlansabar.blogspot.ca/2013/06/jbehave-tutorial-bdd-framework-for.html>

<http://www.slideshare.net/shadrik/bdd-with-java-8323915>