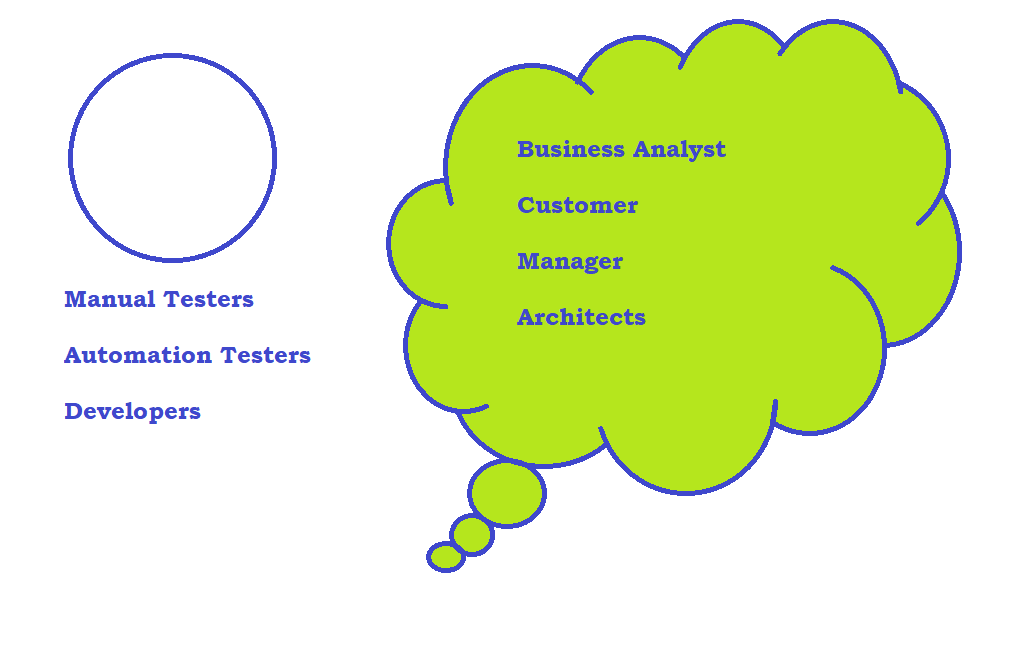
BDD – Cucumber

If we want to include Customers, BA, Managers, Architects in Testing (Automation testing) we can go with BDD Approach.

BDD is a tool independent Automation approach. irrespective of any tool we choose we can use BDD as a automation framework.



Usage :

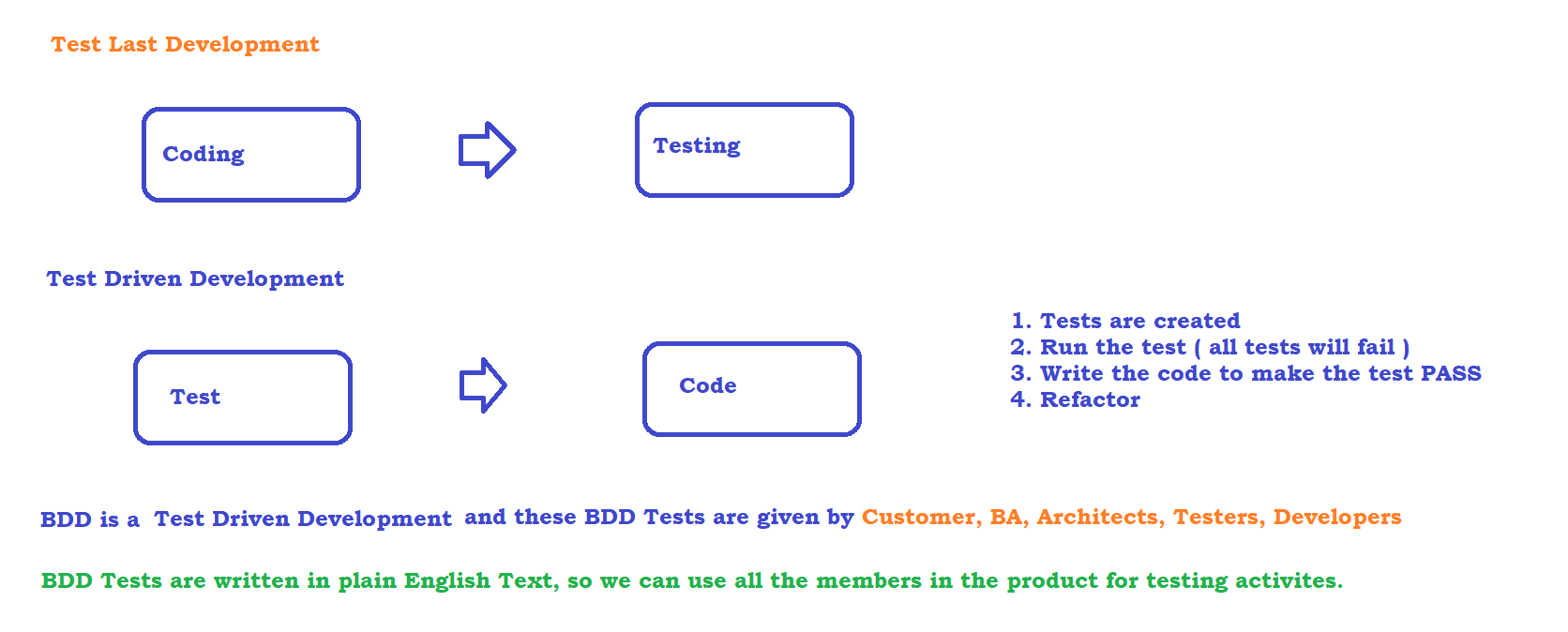
- Web Automation

- Mobile Automation

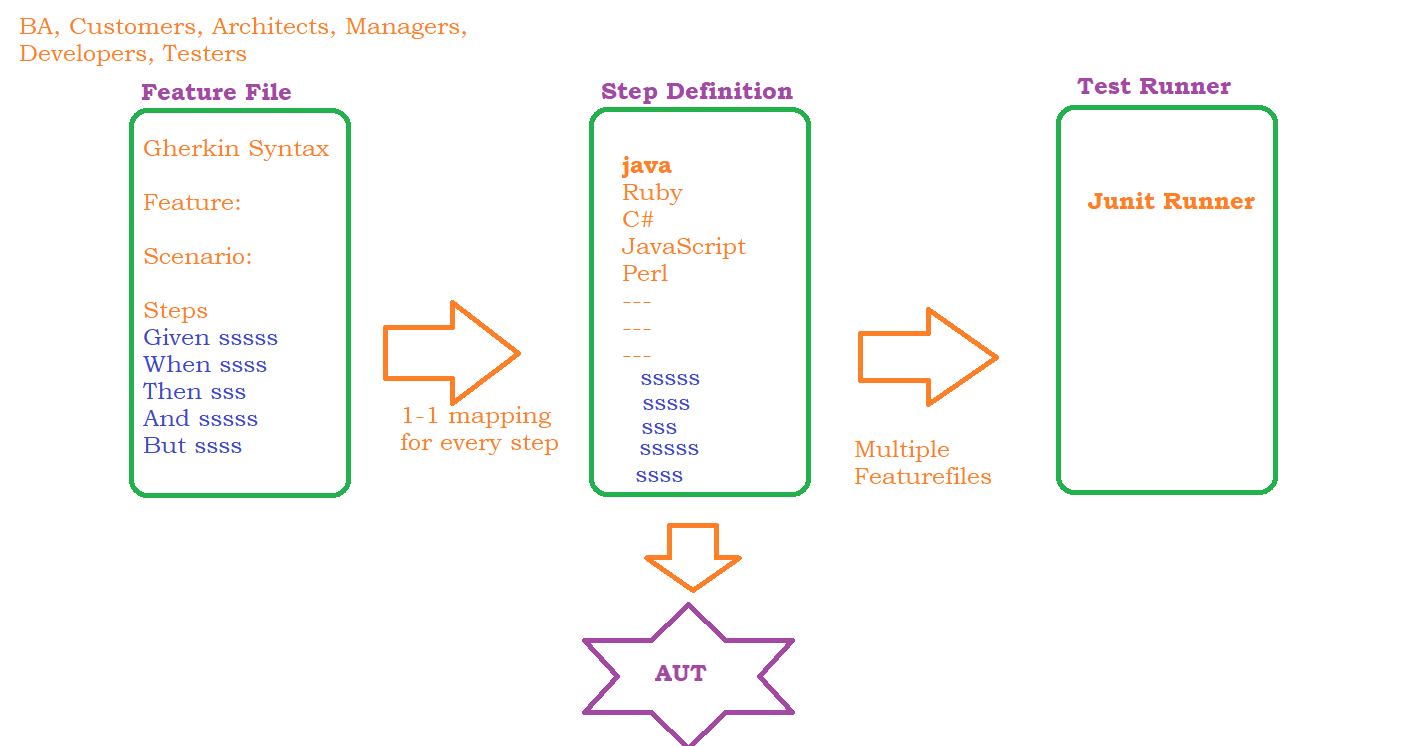
- API Automation

- Unit Tests

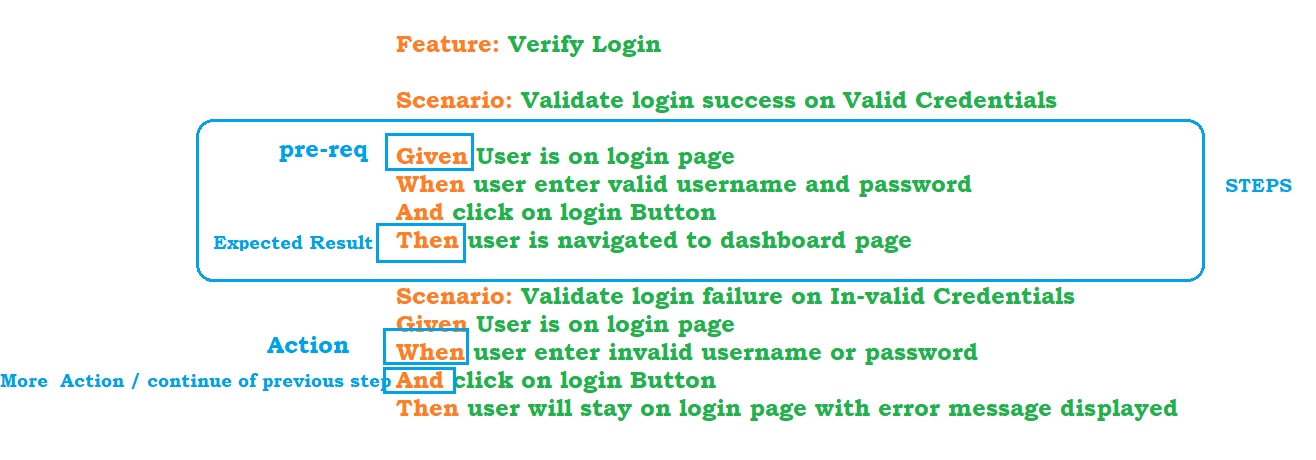
**Development Approach:**



Main Components of BDD

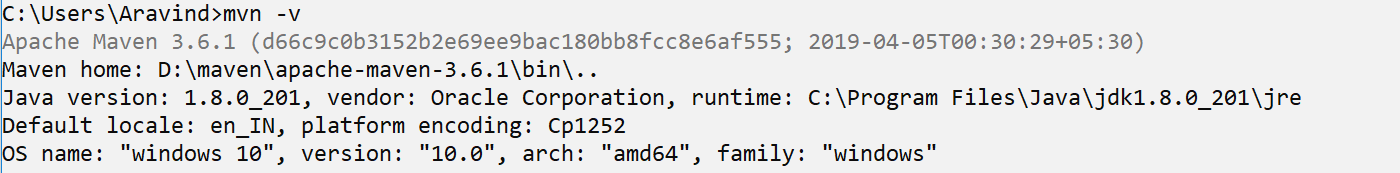


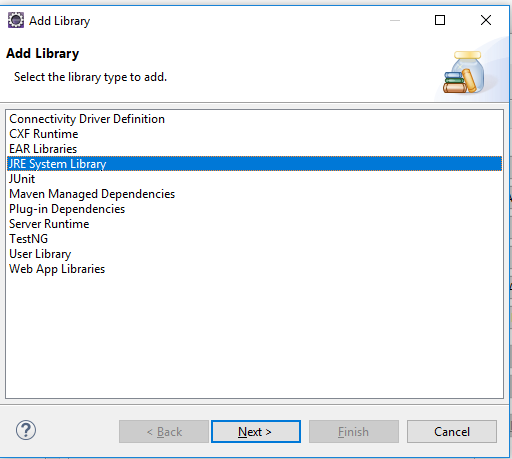
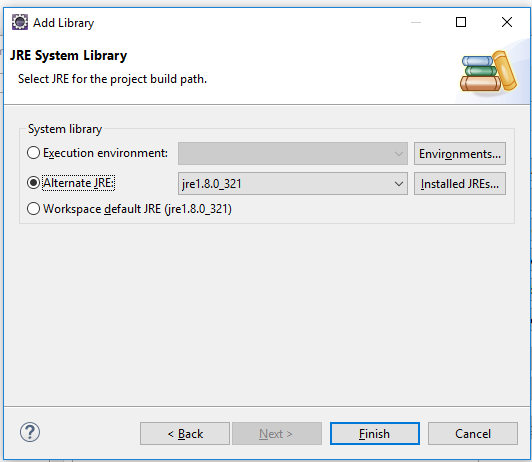
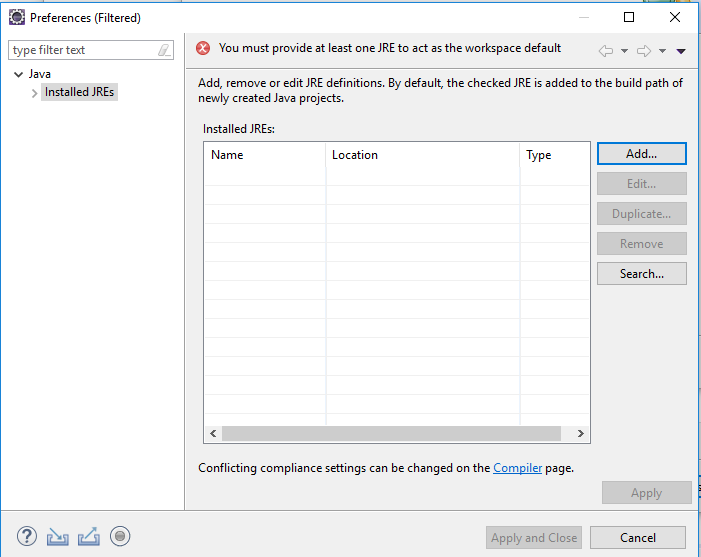
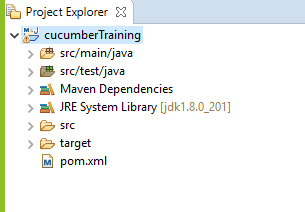
Sample Feature File



Configure Eclipse to use BDD

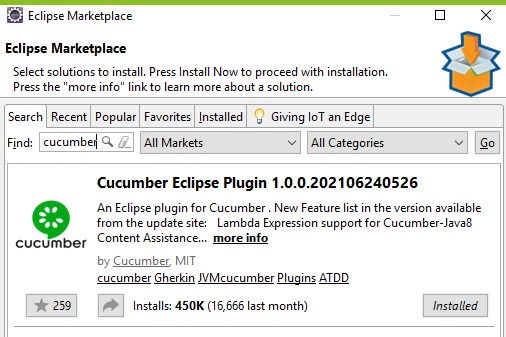
* Download and install maven : <https://maven.apache.org/download.cgi>
* Download the binary zip file and unzip to any location, EX: D:\maven
* Update the environment variables
  + MAVEN\_HOME : D:\maven\apache-maven-3.6.1
  + M2 : D:\maven\apache-maven-3.6.1\bin
  + PATH : D:\maven\apache-maven-3.6.1\bin
* command prompt mvn –v



* File ->New -> other ->Maven -> Maven project
* select maven-archetype-quickstart
* Provide groupid and artifactid
* **Update JRE to latest**
  + Rightclick on project
  + go to properties
  + click on java build path
  + click on Libraries
  + click on libraries
  + select the JRE and remove it
  + click on add library
  + 
  + 
  + 
  + under JRE Home browse and select the JRE Present in JDK location (C:\Program Files\Java\jdk1.8.0\_201\jre)
  + click on ok and apply and close
  + 
* Update Java compiler
* Update pom.xml with latest cucumber-java and cucumber-junit dependency

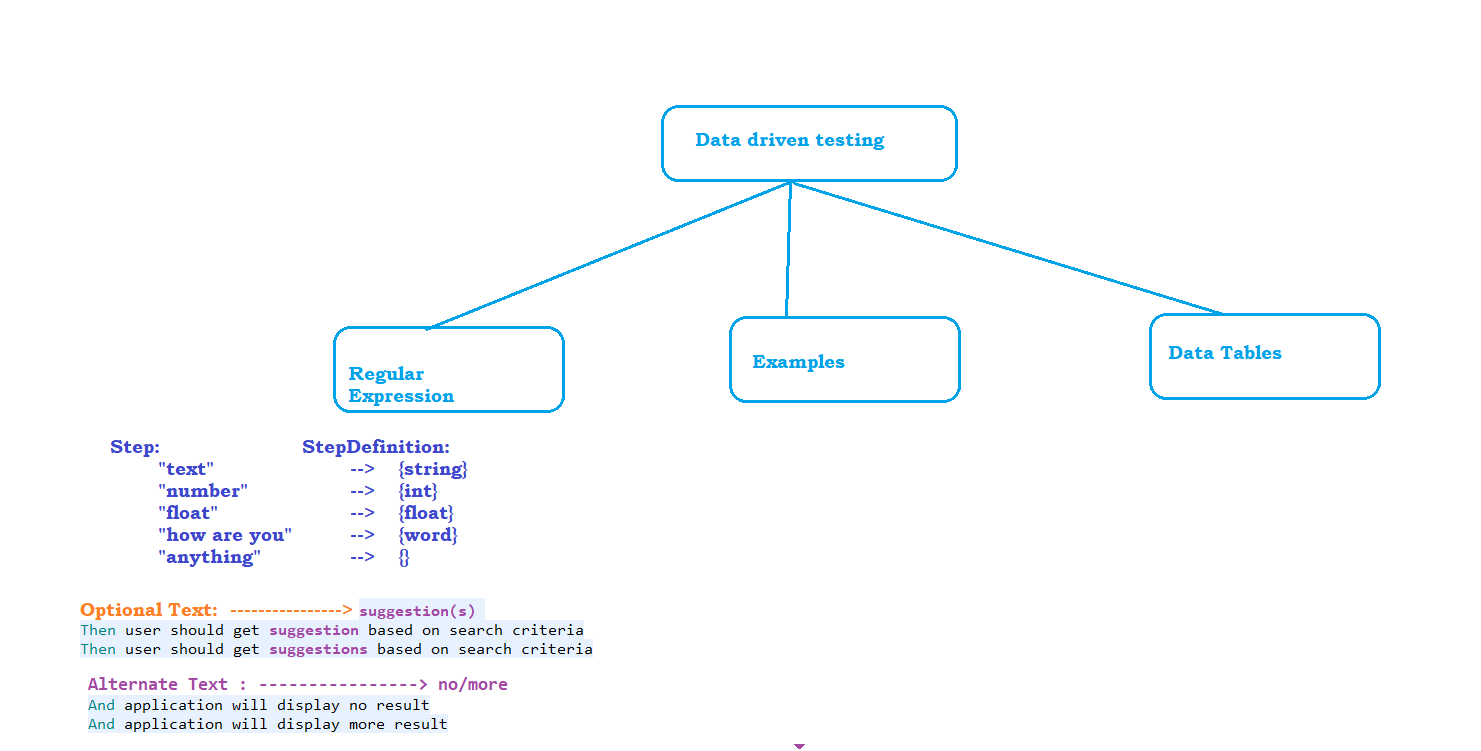
https://mvnrepository.com/artifact/io.cucumber/cucumber-java/7.2.3

<https://mvnrepository.com/artifact/io.cucumber/cucumber-junit/7.2.3>

* Install cucumber plugin
  + Help -> Eclipse Market place ->
  + 

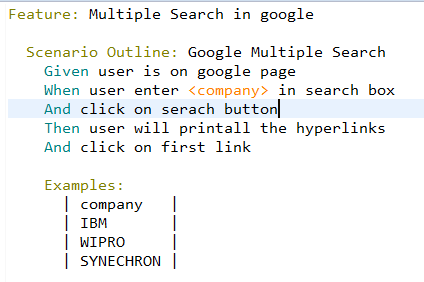
Data driven testing

<https://github.com/cucumber/cucumber-expressions#readme>



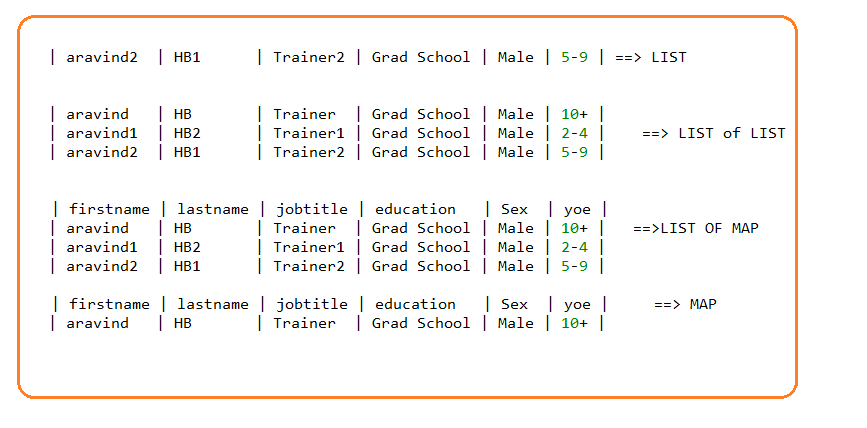
Examples :

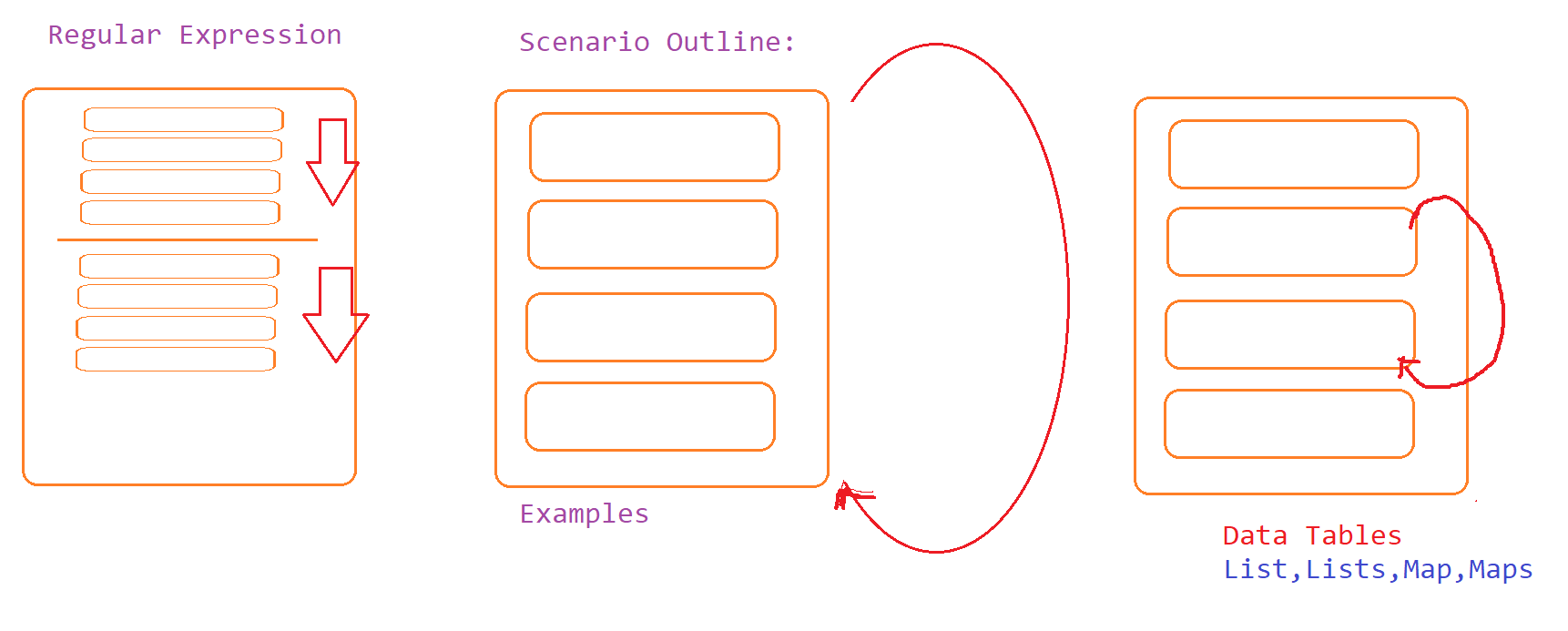
Main drawback of using RE is we are forced to keep multiple scenarios. to avoid that we can go with Examples Table.



Data Table :

Main drawback of Examples is, it will execute the complete test for every data. if we want only some steps to be executed then we can go with data table

******

******

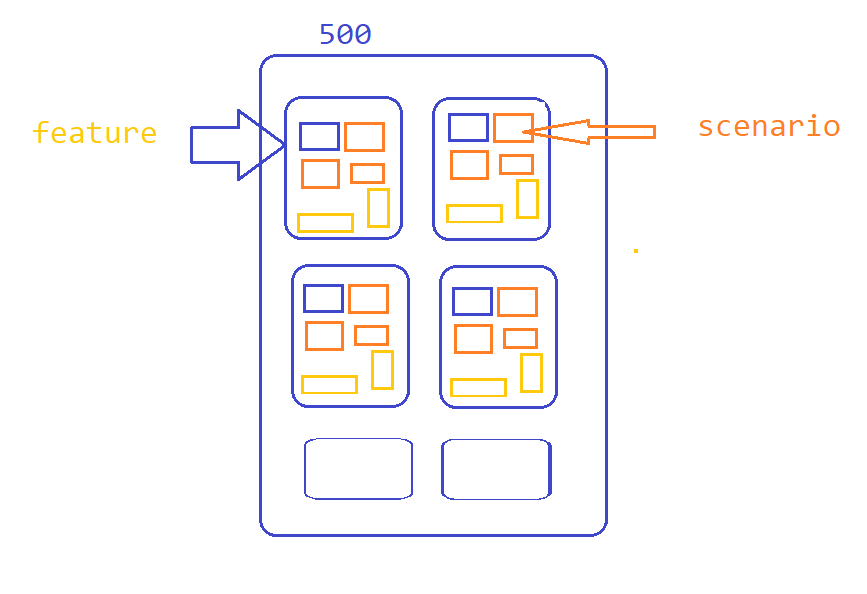
Tags : tags are mainly used to group the tests

@regression

@uat

@smoke

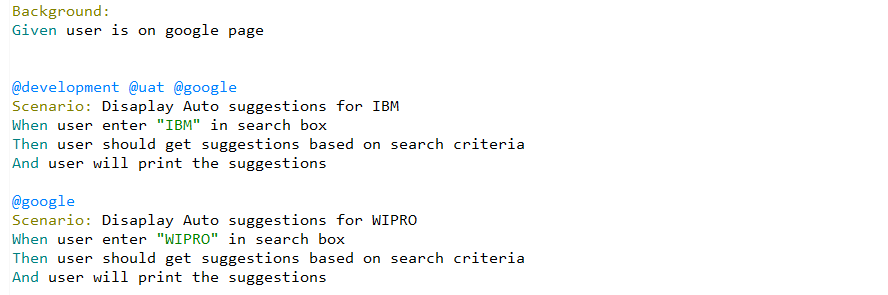
@sanity

******

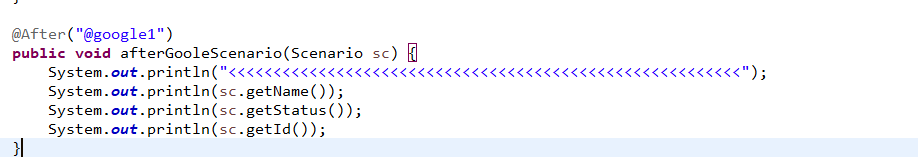


Background :

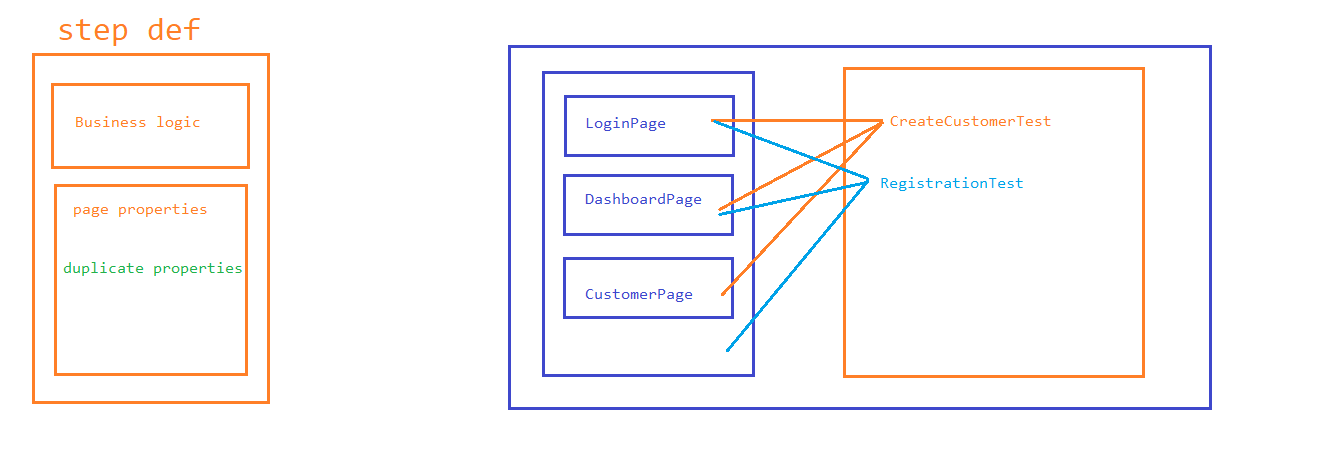
Background are the preconditions to the Scenario. instead of keeping the code in all the tests which are in common, we can put it in Background



Hooks :

* hooks are used for performing pre and post conditions for a tests
* @Before
  + create a driver
  + launch application
  + DB connections
  + Proxy settings
  + Enable cookies
* @After
  + close driver
  + logout
  + close db connection
  + clear proxy
  + delete all cookies
* Tagged hooks or conditional hooks
  + before(“@google”)
  + before(@google and @uat)
  + after(“@google”)
  + after(@google and @uat)
* we can also use scenario class to get some details of the scenario
* 
* Order of execution
* @Before(order = 1)
* @Before(order = 2)
* To execute any piece of code before and after executing step
  + @BeforeStep
  + @AfterStep

POM

******

CLI

