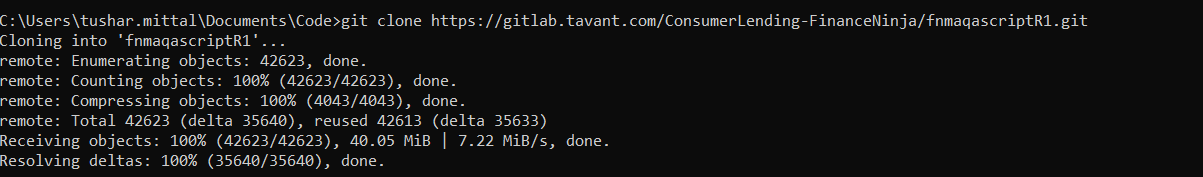
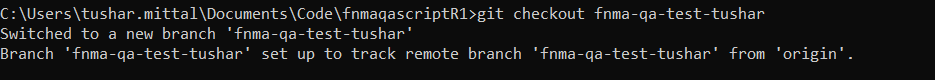
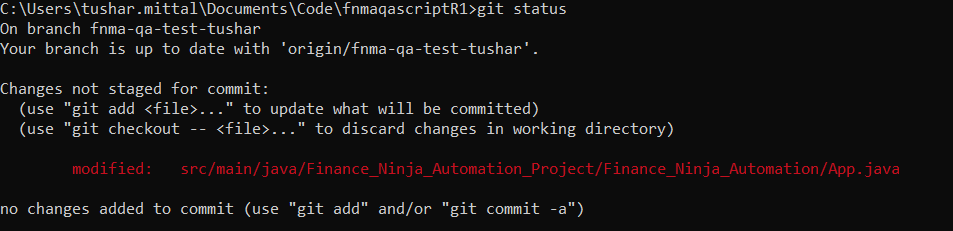
1. If we are unable to check in any type of file, jar file, application to GitHub. Then check in gitignore file. May be it has been stopped in gitignore file to go to Git. Or maybe like it won’t process that file further.
2. **Git Commands:**
3. **Cloning any new Project** - If we want to clone any project code, just go to the folder where we want to paste our code and enter cmd in the command line. Then enter “git clone <<< code url>>”. So basically this will download code from master branch.



1. Then Just enter cd and click on Tab bar in the keyboard. It will take to the project/ folder location
2. **How to checkout to our individual branch from master branch( or the branch from where the cloning happened) -** Then We have to check out our own branch, means we can’t work on Master branch directly, so we have to switch to our qa branches. We have to give comment like “git checkout <<branch name>>



1. **How to verify the current branch ie. Code is pointing to which branch currently -** Then just enter the command “git status”. We should get below thing. Here I just changed 1 small change to one of my code.



1. **How to check in the code** - Then we have to give command “git add .” --- It means that we want to add all the changes to check out. Generally if we want to do it for a specific file, we have to mention that particular file name only.

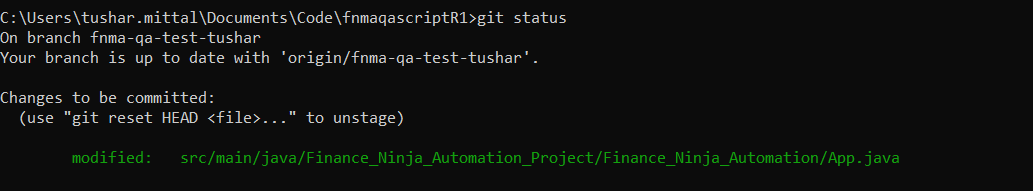
One more catch here is, if we use “.” In add statement, then its like adding all changes to commit. But if we want to add specific files to commit, then use “git add <file>”

Suppose we have 3 files to commit – File1 , File2 and File3.

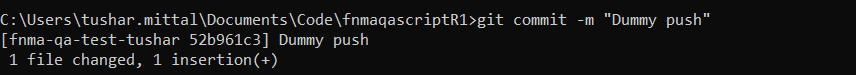
Then command will be :

“Git add File1,File2,File3”

1. Then enter comment “git status” to check whether our changes are ready to check in or not. Change file should turn into green.



1. **Committing the changes -** Then we have to commit our changes by this command “git commit -m “Dummy Push”. Here -m represents if we have to give some comment while pushing our changes.

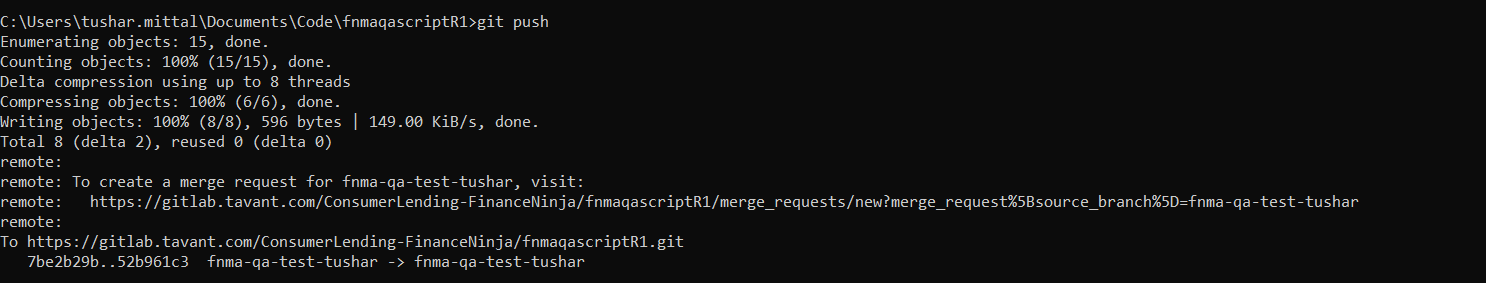


Note : In above way, we are adding the changes first then commit it, means using git add . then, using got commit .

We can do this activity in 1 statement as well,

Commit -a

1. **Push the changes to QA branch after commit –** Git push command is used to upload local repository content to a remote repository. Finally we have to push our changes to the checkout branch. Give command “git push”



While Pushing code to the remote branch or current branch, we have a command like ---

“git push origin Ninja-web-tusharNew”

But 1 thing to note here, if suppose we are already on the current branch in which we did some modifications. Then we are committing code and want to push the changes to the current branch only. Then command “git push” is fine.

But if suppose we want to push our changes to any other branch then, we have to use above command in which we’ll mention branch name after origin keyword.

**Diff b/w git pull and git fetch –**

When downloading content from a remote repo, git pull and git fetch commands are available to accomplish the task. You can consider git fetch the 'safe' version of the two commands. It will download the remote content but not update your local repo's working state, leaving your current work intact. git pull is the more aggressive alternative; it will download the remote content for the active local branch and immediately execute git merge to create a merge commit for the new remote content. If you have pending changes in progress this will cause conflicts and kick-off the merge conflict resolution flow.

Command “git branch” will give the list of local branch refs.

But if we run command “git branch -r” , then it will give all the branches of the Origin master of project

1. **Git Pull** – If we want to pull or take the code from a remote branch to the current branch( local repo branch). Then we have to write this command –

git pull origin <<branch name from where code has to pull>> <<current branch or that branch name to which code has to pull>>

See, we have seen that if we use git fetch command to fetch a code from 1 branch to another. Then we have to use git merge command as well. Since git fetch in itself is not sufficient to copy code from 1 branch to another. But if we use git pull, then git merge is using inside or we can say behind the scenes.

1. If we need to verify that any particular element shouldn’t displayed in the screen too, then use findlelements method so that that list will be empty but it’ll not fail the further execution

**if**(DriverFactory.*getWebDriver*().findElements(warningMessage).size() != 0)

{

System.***out***.println("Entered here");

SoftAssert softassertion = **new** SoftAssert();

softassertion.assertTrue(**false**);

}

1. **Cucumber feature file - Keywords & Other imp notes**
2. Framework – It’s an Execution System used to run and maintain Automated tests.
3. What is Cucumber: Cucumber is one of the framework used in running our Automated TCs. Generally we call Cucumber as BDD Framework. Cucumber is the BDD Framework for running Automated Tests. Since it’s an framework, it will not Automate your TCs. It just help in executing those Automated TCs and gives a path to do that. Its not any specific testing tool, just an framework.
4. Since its an framework, not a testing tool. So what basically it helps in doing : It helps in Data Driven (driving the data from external file/ feature files), Parameterization, Execution Controls and Hooks (Used to run any specific TCs, any specific modules TCs), Reports and Automation utilities. So basically we can implement these thing on our Automation TCs with the help of Cucumber.

Need to check the Relation & Difference between Cucumber & BDD :

* Behavioral Driven Development (BDD) is a software development approach that has evolved from [TDD](https://blog.testlodge.com/what-is-tdd/) (Test Driven Development). It differs by being written in a shared language, which improves communication between tech and non-tech teams and stakeholders. In both development approaches, tests are written ahead of the code, but in BDD, tests are more user-focused and based on the system’s behavior.
* BDD or Behavior-driven development is a process of developing software based on TDD (Test Driven Development) which focusses on the behavioral specification of software testing units.
* We write tests first and then the application code in BDD and Tests are explained as behavior of application and are more User focused. Tests are written in plain descriptive English which we call it as Gherkin language. It’s an easy to understand language. Cucumber is the name of BDD framework.
* Cucumber is a testing framework which supports Behavior Driven Development (BDD).
* Test cases can be written using real-time examples of the actual requirements, to explain the behavior of the system.
* *Cucumber* uses plain English in the Gherkin format to express user stories.
* A cucumber is a framework/ execution system used to run automated acceptance tests created in a BDD format
* 1 more BDD framework is JBehave.

1. IMP Question >> What Type of Automated Test cases does Cucumber support ? Answer – Cucumber supports any test (Web, Mobile, API, Unit Test) written in Java & Ruby language.
2. IMP Question >> How cucumber stands unique and best from other frameworks in market (Keyword, Data driven & Hybrid) – Manual Test case writing effort can be reduced. Since the TC/Req are defined on BDD methodology (Gherkin syntax) which is easy to understand for anyone in the Project.
3. What is Gherkin : It’s a business readable, domain specific language, the language in which TCs has to be written. In cucumber, Test cases are represented as scenarios.
4. Keywords use in Gherkin language are : Given, When, Then, And, But (Case Sensitive). Below mentioned about ‘And’ & ‘But’ keywords.
5. And : This is used for statements that are an addition to the previous steps & represent positive statements.
6. But : This is used for statements that are an addition to the previous steps & represent negative statements.
7. Feature file act as a Test Suite which consists of all the scenarios.
8. Maven is a Build management tool, Natural is the plugin need to add for Cucumber code. Its just a plugin to give colors to the feature file, not any mandatory plugin to run the code.
9. When we are writing our Feature files, we shouldn’t give gap in the first sentence :

Feature: Application Login ----- Correct

Feature : Application Login ----- Wrong

1. **Latest Version of Selenium is 4.0.0 alpha6. Other versions listed down as below :**

4.0.0.alpha6 - May 28, 2020 (154 KB)

4.0.0.alpha5 - March 18, 2020 (131 KB)

4.0.0.alpha4 - January 09, 2020 (131 KB)

4.0.0.alpha3 - July 08, 2019 (104 KB)

4.0.0.alpha2 - May 02, 2019 (102 KB)

4.0.0.alpha1 - May 01, 2019 (844 KB)

3.142.7 - December 27, 2019 (845 KB)

3.142.6 - October 04, 2019 (845 KB)

3.142.5 - October 01, 2019 (845 KB)

3.142.4 - September 02, 2019 (845 KB)

3.142.3 - May 21, 2019 (845 KB)

3.142.2 - May 11, 2019 (845 KB)

3.142.1 - May 07, 2019 (845 KB)

3.142.0 - April 24, 2019 (845 KB)

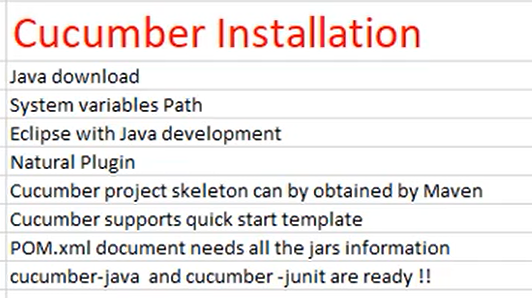
3.141.5926 - April 18, 2019 (845 KB)

3.141.592 - April 17, 2019 (845 KB)

3.141.59 - April 17, 2019 (844 KB) yanked

* 1. - November 01, 2018 (842 KB)

1. **Cucumber Installation :**
2. Install Java – Set Java\_Home path and Java bin path in System variables
3. Add 1 Plugin in Eclipse : Natural 0.7.6
4. Setup a Maven (Build Management Tool) Project in Eclipse
5. While creating a Maven Project, remember that select the Template “maven-archetype-quickstart”
6. We need to download jars available in Maven Repository to run our code. To achieve this, we mention those jars as a dependency in Maven Project.
7. We need 2 jars to run Cucumber supporting code – cucumber-java and cucumber-junit



1. **Some More Points on Cucumber framework :**
2. throw new pendingException() – it means that u have something pending to write inside that method.
3. Now we’ll create Runner class – we have to use Annotation (@CucumberOptions) . **@CucumberOptions** are like property files or settings for your test.
4. Glue : glue is a parameter that define steps definition file. We Just need to pass the package names. The path of the step definition files.
5. **VV Imp** – If we want to run any Java class using “Run as Java Application” then it will search for the main method in that Java class. If there is no main method, then it won’t run.
6. **V IMP** – If we want to run any Java class as testng, then it should have at-least one @Test annotation and then it will run all the methods under that class.
7. **V IMP -** [Eclipse Maven dependency jar grayed out, can't import classes from it](https://stackoverflow.com/questions/51372670/eclipse-maven-dependency-jar-grayed-out-cant-import-classes-from-it). -- Answer is , its because of using “test” in added dependency in pom file.

So change from test to compile :

<scope>test</scope>

To

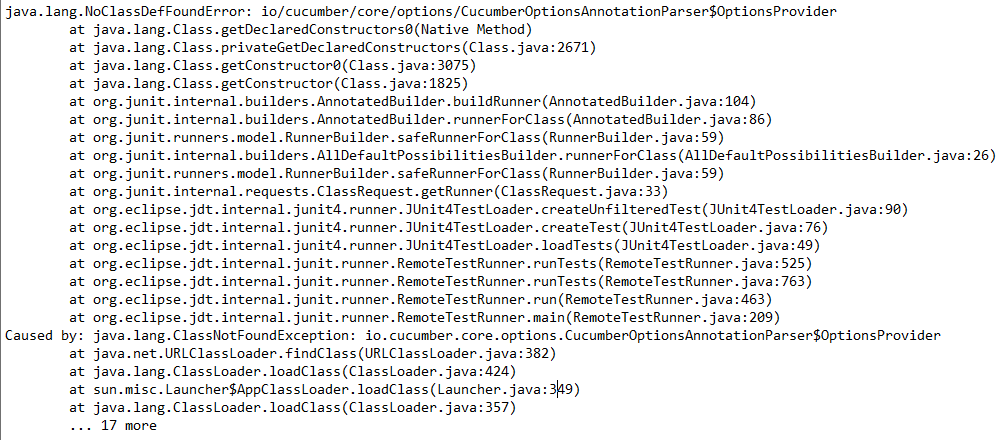
<scope>compile</scope>

It will look like :



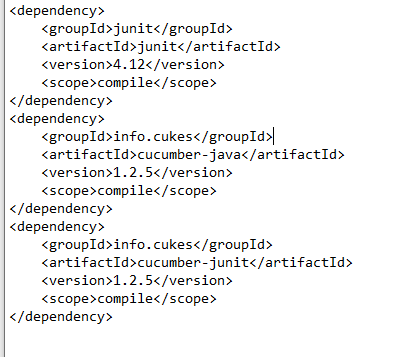
Post that, grey out will go off.

1. Now, if we want to run a basic Cucumber class – TestRunner class, we can’t run it ‘Run as Java Application’, ‘Run as TestNG’. We can run only as “Run as Junit Test”.
2. I am getting an error while running the TestRunner class in Cucumber framework with “Run as Junit Test” :



How to solve this :

This error is causing due to version changing only, nothing else. Use below Maven dependencies :



1. How to use Data Table :

No of ways are present , suppose we write below statement in Feature file

*And* its passing new values

|testA| testdata22|test charte12|

Now here, we are writing only 1 row of values, means values at index 0 only. So since its just 1 row, we can get these values in Java List<List<String>> format like below :

Case 1 : If we use List<List<String>>,and we use data.raw(), then outer layer of List<String> will get the whole 1st row at 0 index value

@When("^its passing new values$")

**public** **void** its\_passign\_new\_values(DataTable data) **throws** Throwable {

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

System.***out***.println(obj.get(0).get(0)); // Answer will be testA

System.***out***.println(obj.get(0).get(1)); // Answer will be testdata22

System.***out***.println(obj);

System.***out***.println(obj.get(0)); // this will display the whole first row of values

Case 2:

List<String> obj1 = data.asList(String.**class**);

System.***out***.println(obj1);

System.***out***.println(obj1.get(2)); }

Case 3 : Refer List<Map<String,String>>, since it will take the data like 1st row as headers, then other row as Values.

1. Different ways of passing values in Feature file :

1---- *And* User is passing two values "John" and "Peter"

*And* User is passing three values "John" and "Peter" and "Samuel"

Step definition will look like :

@When("^USer is passing two values \"([^\"]\*)\" and \"([^\"]\*)\"$")

**public** **void** user\_is\_passing\_two\_values\_and(String arg1, String arg2) **throws** Throwable {

System.***out***.println(arg1);

System.***out***.println(arg2);

}

@When("^USer is passing three values \"([^\"]\*)\" and \"([^\"]\*)\" and \"([^\"]\*)\"$")

**public** **void** user\_is\_passing\_three\_values\_and\_and(String arg1, String arg2, String arg3) **throws** Throwable {

System.***out***.println(arg3);

}

Explanation : In this case, Same string values mentioned in Feature file will directly pass to the method

2---- *And* When User login to application with mentioned *<Username>* and *<Password>*

**Examples:**

|Username|Password|

|User1|Pwd1|

|User2|Pwd2|

Or *And* When User login to applicatin with mentioned "*<Username>*" and "*<Password>*"

Difference between these 2 types of syntax is :

1. What is Doc String in Cucumber Feature file ?

Answer : Doc String is a way of giving test data in feature file when requirement of sending data is that, we need to send data in multiple lines, not in a continuous line. In that case, we’ll use Doc Strings.

Below snapshot from **Feature File** :

*And* When User login to application with mentioned "*<Username>*" and "*<Password>*"

*And* I fill in message with:

"""

Dear Patron,

I have recieved the letter

Your truly

"""

*Then* User pass the answer "*<Mad>*"

**Explanation** : We have to write “with:” to initiate the Doc String sentence, And 3 codes is must.

**Step Definition** file for above statement :

@When("^I fill in message with:$")

**public** **void** i\_fill\_in\_message\_with(String arg1) **throws** Throwable {

System.***out***.println(arg1);

}

**Output :** The below output is only for Doc String sentence.

Dear Patron,

I have recieved the letter

Your truly

1. Few Terminologies of Cucumber Framework / BDD –

* Doc Strings
* Step Tables/ Data Tables
* The Background Section
* Scenario Outline
* Tags
* Comments

1. **Complete Understanding of Test Runner Class & other concepts of Cucumber :**
2. Cucumber Options Annotation – Complete Details - -- Pending to read
3. Tags – How to give single-1 Tag in Runner class, how to give multiple Tags, how to ignore any particular tags.—Pending to read – toolsqa website
4. Background : Background need to write like a pre requisite for any TC , it need to be write only on the top before all other TCs in the feature file. Also, it works only for that particular feature file where it got mentioned. Background is only applicable for the condition where we know that Pre requisite is same for all the scenarios mentioned in that specific feature file.
5. Hooks : Hooks also work like a Pre- requisite but it will help if pre- requisite is different for multiple scenarios of the same feature file.

What is the Key difference between Background and Hooks – Both work as a pre-requisite but Background work only for that specific Feature file where its mentioned but Hooks works through out all the Feature files.

Now, suppose we create Feature file with 2 scenarios, and we create 1 Hooks Java class. Now, we created 2 methods with @BeforeClass & @AfterClass annotation.

@Before

**public** **void** beforeExecute()

{

System.***out***.println("Before execution");

}

@After

**public** **void** afterExecute()

{

System.***out***.println("After execution");

}

Here, we didn’t give any particular Tag name to run, it means these 2 methods will run for all the scenarios mentioned in all the Feature file.

Now, we’ll mention 1 Background in Feature File. Now, both Hooks and Background will be run. The priority will be :

* Hooks (@Before)
* Background
* All steps of the scenario
* Hooks (@After)

Now, other scenario would be, we make 1 more annotation (@BeforeClass) but this time for a specific tag scenarios. Now, when we run the Feature file for all the scenarios, preference would be,

* Hooks (@Before) – Without any parameter for any particular Scenario
* Hooks (@Before) – With the tagged scenario when that scenario turn came
* Background
* All steps of the scenario
* Hooks (@After) - With the tagged scenario when that scenario turn came
* Hooks (@After) – Without any parameter for any particular Scenario.

1. Glue - We have to give only package name inside glue, it could be step definition packages, or any other Java class Packages. Glue will have step-definition package details.
2. Plugin – this is used for Reports.

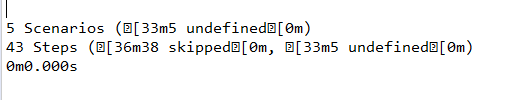
Syntax is like “

plugin = { "pretty", "json:target/cucumber.json", "html:target/cucumber-reports" })

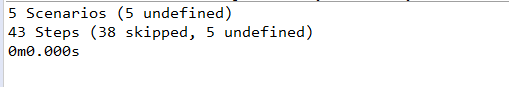
* “pretty” keyword basically applied color to the actual report.
* Now we have to tell what kind of output report we are expecting. Suppose if we are expecting an html report then we need to write like “html:”
* We can create html, json and xml report. Cucumber gives that Option . How to achieve this, by using below statements :
* json:target/cucumber.json – This is for json report
* html:target/cucumber-reports – This will create a folder “cucumber-reports” and html report will be created inside
* junit:target/cukes.xml – This one is for xml report

1. dryRun : We’ll use this attribute to find if any scenario or test step mapping with Step definition file is missing ? If its missing, it will clearly list down all the unimplemented scenarios. Also, its not specific to any particular Feature file. It works across all the Feature file. This values has to set as “dryRun=true”
2. monochrome – It will give the proper Output in console,

We can see below output, full of symbols



Now, when we give monochrome = true, then we’ll get a proper output like below :



1. strict – if we don’t want to pass any TC or step if there is any missing implementation, then we’ll use strict attribute with value as true. Generally we set this keyword value as false. But if set true, execution will happen, but it will fail. And if we didn’t set this attribute value as true, then Execution will happen and Pass as well.
2. Common Framework Issues/ Exceptions –
3. We should write a reusable code – Also, no duplicate step definition should present for any particular Scenario.
4. How to drive the data into TC – Use DataTable
5. How to parameterize TCs with multiple set of test data – Use Scenario Outline
6. How to run all Test in 1 click or how to run any particular set of TCs. – For particular use Tags, Hooks, any specific feature file or feature file folder.
7. How to generate HTML reports and Junit Reports ? By using plugin.
8. What is Parallel Runner class ? How to do Parallel execution ?
9. How to run any TC multiple times continously,
10. Cross browser testing in Cucumber framework.
11. Need to learn to execute code in cucumber framework with different-2 browsers and web, mobile also. I mean code should be flexible enough to do these changes.
12. UI Automation – with selenium
13. How to integrate 2 different Projects in Eclipse. If required how to fetch the data and all .
14. Why we use Maven to run our code or made the project. ?

Answer : Maven is a software project management and Build management tool for Java frameworks. By Java frameworks we means like Selenium framework if we write code in Java.

Maven can be used in maintaining projects. Few points to note here that :

1. Central repository to get dependencies. No need to download jar files from outside and paste it to the Build path at project level. So with having central repository and adding dependencies, its easy to select Maven project.
2. Maven gives an automatic created “project folder structure” that can be used like a standard structure.
3. Flexible in integrating CI tools like Jenkins to Maven easily – Some maven commands are available which can do these things.
4. Plugins – Plugins like TestNG, Junit supported by it.

Now we have to install Maven, Java is the pre-requisite for that. Download bin.zip file (Binary Zip Archive). Further set the Maven\_Home path in system variables.

Understanding of pom file Terminologies :

We see the dependency mentioned below like :

<dependency>

<groupId>io.cucumber</groupId>

<artifactId>cucumber-java</artifactId>

<version>4.0.0</version>

<scope>compile</scope>

</dependency>

Among this dependency :

groupID : groupID will identify your project uniquely across all the projects. Maven will uniquely identify cucumber project with the groupID given above

Artifact – An Artifact is a file, usually a jar, that’s get deployed to a Maven repository.

IMP – While creating a Maven Project, whatever artifactID we are passing that’s the Project name and groupID will refer to Package Name.

1. **VV IMP** – In order to run our TCs with the help of Maven, We have a plugin called “Maven surefire plugin, we need to place it in our pom.xml file. So, that the importance of this plugin, we have to install it if we want to run our TCs in Maven. Since its that that plugin with what we can execute our Maven TCS
2. **Maven Clean –** Command will look like “mvn clean”. SO it will delete all the temporary files or builds whatever Current project has. This is preferred before you execute any new Test. And, these commands we should run at the project level dos command prompt, not the whole system prompt level. The Main point to note here that – Basically it should point out the pom.xml file. See why we run always at Project level path only, because we always place our pom.xml file directly in Project level, not inside any folder.

So, the command from where we are running the “mvn clean” will point to pom file only.

**VV IMP** Point here is that, I didn’t put Maven surefire plugin, and I am trying to run these commands at Project level “mvn clean” & “mvn compile”. I got the error message like “mvn is not a recognized. After putting this plugin, everything starts working.

I am able to run these commands perfectly.

**Maven Compile -** Mvn compile will check all the compilation errors in the Project.

1. **Next Step to run TC with Maven command line :** V IMP – If we want to run Maven command with the help of TestNG file, then we have to add the below lines in pom.xml file :

<configuration>

<suiteXmlFiles>

<suiteXmlFile>${basedir}\testng.xml</suiteXmlFile>

</suiteXmlFiles>

</configuration>

Here in the above code, we are giving “testng.xml” directly, so we need to run below maven command to run the TC :

* Mvn test or mvn clean install

Now the other way of giving suite xml file is :

<suiteXmlFile>${basedir}\${xml}.xml</suiteXmlFile>

After that, we need to run the test case using below commands :

mvn clean install -Dxml=testng

or whatever testng xml file name we gave like in my project I gave name “cucumber.xml”. so command will be

mvn clean test -Dxml=cucumber

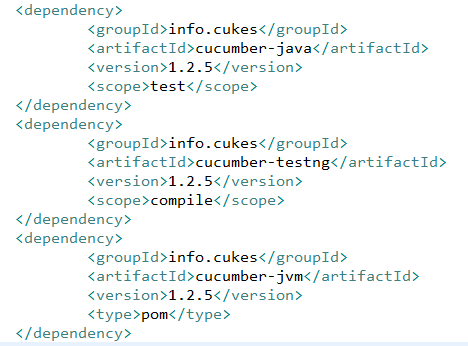
1. Now we need to run Test Runner file through Maven command line – How to achieve this..

First, we should validate that all the Cucumber dependencies should be of same version.

Main dependencies that should be present to run this thing will be below mentioned and use below maven command to run TC :

*mvn clean test -Dxml=testng*

*After that, Execution will run based on Tags used in Test Runner. So we can run our TCs based on given tags in Runner class or based on given tags in maven command line itself.*



1. Parallel Execution in Maven for Cucumber

**TESTNG TUTORIAL :**

1. Some Imp notes for TestNG Framework :
2. If we want to install TestNG in Eclipse, go to Add new Software link and add below details :

Name: TestNG

url: [**https://dl.bintray.com/testng-team/testng-eclipse-release/6.14.3/**](https://dl.bintray.com/testng-team/testng-eclipse-release/6.14.3/)

**Imp Note:** There is a deprecation of ***http://beust.com/eclipse/***. For Eclipse06-18, you can install TestNG from the [***Eclipse Marketplace***](https://marketplace.eclipse.org/content/testng-eclipse), and for 09-19, you need to follow the new link as given in this tutorial.

**VV IMP Info** starts now :

Now we want to work with Chrome driver with Maven Project. We have 2 options to do the same, either we install Chrome drive exe file, then place that file in our Project , then give the command like System.setProperty(chromedriver.exe, << path of the file>>)

And the other way is with Maven Dependencies. 2nd way is the preferable one to do.

We have to add webdrivermanager dependency. It’s an compulsory dependency for browser driver. No need to add Chrome or Mozilla driver dependency separately. Webdrivermanager dependency is sufficient to add.

So, first we need to add below dependency –

<dependency>

<groupId>io.github.bonigarcia</groupId>

<artifactId>webdrivermanager</artifactId>

<version>3.3.0</version>

</dependency>

After that, we need to add below code in our Java class, which should be like the first thing to do.

WebDriver driver = **null**;

WebDriverManager.*chromedriver*().version("83.0.4103.39").setup();

// or WebDriverManager.chromedriver().setup();

ChromeOptions options = **new** ChromeOptions();

driver = **new** ChromeDriver(options);

In above code, we have to give the correct version of ChromeDriver which suits chrome browser version. Like I am using 83.0 version of Chrome browser, then I have to use “83.0.4103.39” version of Chrome driver only.

Please find below list which will give other chrome driver versions compatible with Chrome browser :-

Current Releases

* If you are using Chrome version 84, please download [ChromeDriver 84.0.4147.30](https://chromedriver.storage.googleapis.com/index.html?path=84.0.4147.30/)
* If you are using Chrome version 83, please download [ChromeDriver 83.0.4103.39](https://chromedriver.storage.googleapis.com/index.html?path=83.0.4103.39/)
* If you are using Chrome version 81, please download [ChromeDriver 81.0.4044.69](https://chromedriver.storage.googleapis.com/index.html?path=81.0.4044.69/)
* If you are using Chrome version 80, please download [ChromeDriver 80.0.3987.106](https://chromedriver.storage.googleapis.com/index.html?path=80.0.3987.106/)
* If you are using Chrome version 79, please download [ChromeDriver 79.0.3945.36](https://chromedriver.storage.googleapis.com/index.html?path=79.0.3945.36/)
* If you are using Chrome version 78, please download [ChromeDriver 78.0.3904.105](https://chromedriver.storage.googleapis.com/index.html?path=78.0.3904.105/)
* If you are using Chrome version 77, please download [ChromeDriver 77.0.3865.40](https://chromedriver.storage.googleapis.com/index.html?path=77.0.3865.40/)
* If you are using Chrome version 76, please download [ChromeDriver 76.0.3809.126](https://chromedriver.storage.googleapis.com/index.html?path=76.0.3809.126/)
* If you are using Chrome version 75, please download [ChromeDriver 75.0.3770.140](https://chromedriver.storage.googleapis.com/index.html?path=75.0.3770.140/)
* If you are using Chrome version 74, please download [ChromeDriver 74.0.3729.6](https://chromedriver.storage.googleapis.com/index.html?path=74.0.3729.6/)
* If you are using Chrome version 73, please download [ChromeDriver 73.0.3683.68](https://chromedriver.storage.googleapis.com/index.html?path=73.0.3683.68/)

So, we have to keep the matching version of Chrome Driver with Chrome Browser.

We can use below statement as well : WebDriverManager.chromedriver().setup();

**The primary method is not necessary for a TestNG file. Moreover, the methods in the TestNG file need not be static in their behavior.**

1. ***What is TestNG Suite ?***

**Answer : T**he collection of [TestNG Tests](https://www.toolsqa.com/testng/testng-test/) together is called a **Test Suite**. A test suite can run multiple tests at once by executing the test suite. Additionally, these test cases can be dependent on each other or may have to be executed in a specific order independently.

We can implement multiple Test classes as well in testng.xml file.

1. ***Test NG Annotations :***

TestNG annotations are the code that is written inside your source test code logic to control the flow of the execution of tests. It is essential to annotate your methods in TestNG to run the tests. [*TestNG*](https://testng.org/doc/)will ignore the method which does not contain an annotation since it won’t know when to execute this method

***@BeforeMethod (alwaysRun = true) . So this one is the pending one.***

***Types Of TestNG Annotations :***

In TestNG, there are ten types of annotations:

* ***@BeforeSuite*** – The @BeforeSuite method in TestNG runs before the execution of all other test methods.
* ***@AfterSuite*** – The @AfterSuite method in TestNG runs after the execution of all other test methods.
* ***@BeforeTest*** – The @BeforeTest method in TestNG runs before the execution of all the test methods that are inside that folder.
* ***@AfterTest*** – The @AfterTest method in TestNG executes after the execution of all the test methods that are inside that folder.
* ***@BeforeClass*** – The @BeforeClass method in TestNG will run before the first method invokes of the current class.
* ***@AfterClass*** – The @AfterClass method in TestNG will execute after all the test methods of the current class execute.
* ***@BeforeMethod*** – The @BeforeMethod method in TestNG will execute before each test method.
* ***@AfterMethod*** – The @AfterMethod method in TestNG will run after each test method is executed.
* ***@BeforeGroups*** – The @BeforeGroups method in TestNG run before the test cases of that group execute. It executes just once.
* ***@AfterGroups*** – The @AfterGroups method in TestNG run after the test cases of that group execute. It executes only once.

These annotations have self-explanatory meanings. It is one of the primary reasons to prefer TestNG as it is simple and easy to learn. If TestNG draws so much from its annotations, there must be a few benefits associated with it.

When we are using Annotations. We don’t need to extend any Test class like JUnit.

Hierarchy of above mentioned TestNG Annotations :

* *@BeforeSuite*
* *@BeforeTest*
* *@BeforeClass*
* *@BeforeMethod*
* *@Test*
* *@AfterMethod*
* *@AfterClass*
* *@AfterTest*
* *@AfterSuite*

What is the priority of running the Test methods in a whole class :

* Ctrl + Shift + O is the shortcut to import all the packages in any class in Eclipse.
* Now if suppose there are 3 Test methods in any class, then BeforeMethod and AfterMethod annotations will run 3 times.
* If there are multiple @Test cases (means Test methods of a particular class), TestNG runs the test cases in the alphabetical order.
* We can set Priority to the Test Annotations. We have to use syntax like

@Test(priority = 1)

**public** **void** logout()

{

System.***out***.println("Test Data part");

}

So, we have to write priority like above and it starts from 0

* 1 Imp point, suppose we have 4 test methods in the class, and we have set Priority only for 2 annotations, and for other 2 we didn’t set. Now it will see the Priority of the given annotation, if Priority starts with 0, then that test method which has priority set as 0 will run first. Then other remaining annotation which don’t have priority. Then in the last that annotation which has priority set as some other no. other than 0.
* 1 more point – if suppose we created 3-4 Test methods, and some method start with lower case and some with upper case. Then Upper case methods have the first priority over lower case method names.
* If in the above point, both having the same priority as 0. And other 2 don’t have any priority then the annotations with priority as 0 will run first in alphabetical order, then rem 2 methods.

***Are multiple parameters allowed in annotations?***

*Yes, you can use multiple parameters in the annotations.*

1. **TestNG Groups :**

* TestNG groups combine the tests into groups and let the tester choose which one to run and which one to ignore.
* Groups in TestNG denotes the process of grouping different tests together into a straightforward group and running these tests together by just running the group in a single command.
* IMP Point to note: It does not even matter if they belong to different classes.
* It is important to note that Groups are declared in the testng.xml file in the TestNG and can be found inside <test> tag or <suite> tag.
* One imp thing to observe that if we going to use groups, then we have to explicit define group to all such methods including the common annotations of TestNG like “@BeforeMethod”, “@AfterMethod” etc etc..
* We can define 1 or more than 1 group to any particular annotation, ex :

@Test(groups={“demo”}

@Test(groups={“demo”, “automation”}

* If suppose we gave 2 groups for any particular Test annotation, and we include 1 group in testng.xml file but exclude the other one. Then priority will be exclude over include.
* Now if we use only Exclude statement for the group in testng file, in this case all the TestNG methods will execute except that particular method in which the Excluded group name got mentioned.
* **We can use Regular Expressions as well with TestNG group** : Suppose we have 2 groups like demo1 and demo2. Now we can give group name in Testng file like “demo.\*”. In this case, it will catch both groups.

1. **Dependent Test in TestNG :**

The dependent tests in TestNG determine the dependency of a test on a single or group of tests. In this case, we say that a test is dependent on another test. We can do it in 2 ways

* Using attributes dependsOnMethods in @Test annotation. The dependsOnMethods lets us make a test depend on a particular method. See the Eclipse code,
* Using attributes dependsOnGroups in @Test annotations. It means, it can depend on a whole group rather than a single test.
* Single Dependent Test Methods : We can have like, methodA which is dependent on methodB and also a methodC which is further dependent on methodA itself.
* Multiple Dependent Test in TestNG : When a single test depends on 2 Test methods.
* Inherited Dependent Test Methods in TestNG : A method of a class can be extend a another method of a diff class.

1. **Parameters in TestNG :**

TestNG Parameters : A simple reason to use parameters is that they let us run a function many times with different values or to run different functions with the same values. Parameters pass the values in runtime.

Now, we can define Parameters to any method like below :

@Test

@Parameters({"value1","value2"})

public void loginSetup(String a, String b)

{

System.out.println(a);

System.out.println(b);

}

Now, we can’t run this method directly with TestNG. We have to define Parameters in testng.xml file and gives from there. Then we can use it.

**Note:***TestNG Parameters are run through the TestNG XML file and not from the test case files directly.*

Now, how we need to define in testng.xml file

</test>

<test thread-count=*"5"* name=*"Automation Sample Test2"*>

<parameter name=*"value1"* value=*"TestPart"* />

<parameter name=*"value2"* value=*"DataPart"* />

<classes>

<class name=*"com.automation.stepDefs.TestNGAuto"* />

</classes>

</test>

Synopsis :

* Above, We have defined parameters inside test case level, means for a particular Java class
* Now we can define it at suite level also :
* Optional Parameters : Need to write notes here
* **IMP :** Position of Parameters and Test annotation tag can be interchanged as well. It means, @Parameters can also come below and above of @Test or @BeforeClass etc. etc Annotations.

1. **Data Providers in TestNG :**

The *DataProviders* in TestNG are another way to pass the parameters in the test function, it’s a method that return an array of array of Object.

DataProviders pass different values to the [TestNG Test Case](https://www.toolsqa.com/testng/testng-test/) in a single execution and in the form of [TestNG Annotations](https://www.toolsqa.com/testng/testng-annotations/). DataProviders help in passing the parameters in different ways.

IMP : Difference between Data Provider and Parameters in TestNG ?

Imp Point to note : There are 2 ways we can give Data Provider :

@DataProvider (name = "data-provider")

**public** Object[][] dpMethod(){

**return** **new** Object[][] {{"First-Value"}, {"Second-Value"}, {"Third-Value"}};

}

@Test (dataProvider = "data-provider")

**public** **void** myTest (String val) {

}

In above case, we gave a name to a data provider annotation, then we gave the Same annotation name to the Test Annotation also,

Now other way is :

@DataProvider

**public** Object[][] dpMethod(){

**return** **new** Object[][] {{"First-Value"}, {"Second-Value"}, {"Third-Value"}};

}

@Test (dataProvider = "dpMethod")

**public** **void** myTest (String val) {

}

Now, in the above 2nd case, we gave the Data Provider method name itself in the Test Annotation.

IMP : Values given in Data Provider and when we use that Data provider to any Test class, both should be in sync. We’ll prove this with below example :

Scenario 1 :

@DataProvider(name = "Selenium Data")

**public** Object[][] seleMethod(){

**return** **new** Object[][] {{"Tushar","Mittal"}};

}

And method :

@Test (dataProvider = "Selenium Data")

**public** **void** myTest (String val) {

System.***out***.println("Passed Parameter Is : " + val);

}

Now, here Execution will fail because we are passing 2 values in Data Provider, So it also need a method which needs 2 parameters. In above method, it just need only 1 parameter.

Now, how to rectify this issue. Don’t change anything in above Data provider method. Just change the Test method :

Scenario 2:

@Test (dataProvider = "Selenium Data")

**public** **void** myTest (String val, String data) {

System.***out***.println("Passed Parameter Is : " + val);

System.***out***.println(data);

}

This will give Output :

Passed Parameter Is : Tushar

Mittal

Scenario 3 :

Now don’t change the Test method, just change data provider method little bit :

@DataProvider(name = "Selenium Data")

**public** Object[][] seleMethod(){

**return** **new** Object[][] {{"Tushar","Mittal"},{"ABhi","Tanu"}};

//return new Object[][] {{"Tushar","Mittal"}};

}

Output will be :

Passed first Parameter Is : Tushar

Passed second Parameter Is : Mittal

Passed first Parameter Is : ABhi

Passed second Parameter Is : Tanu

1. **Cross Browser Testing using TestNG :**

[Cross-browser testing](https://www.toolsqa.com/cross-browser-testing/what-is-cross-browser-testing/) is the process of testing our website on different browsers and operating systems. With cross-browser testing, we make sure that the site is rendered the same in every browser. Since all browsers are not built up on the same code base, and shows same behavior for any certain functionality.

* Cross-browser testing using TestNG ensure a better performance on different browsers and OS.
* Image orientations mess up a lot of the time. We can take care of it.
* We have to use TestNG Parameters to achieve Cross Browser Testing.

We have to write Below code :

@Parameters("Browser")

@BeforeClass

**public** **void** methodToLogin(String Browser) {

**if**(Browser.equalsIgnoreCase("firefox")){

WebDriverManager.*firefoxdriver*().setup();

FirefoxOptions options = **new** FirefoxOptions();

driver= **new** FirefoxDriver(options);

driver.get("https://www.toolsqa.com/testng/cross-browser-testing-using-testng/");

}

**else** **if**(Browser.equalsIgnoreCase("chrome")){

WebDriverManager.*chromedriver*().setup();

ChromeOptions options = **new** ChromeOptions();

driver = **new** ChromeDriver(options);

driver.get("https://www.toolsqa.com/testng/cross-browser-testing-using-testng/");

}

}

Then we just need to define the Parameter in testng file :

<parameter name=*"Browser"* value=*"firefox"* />

1. **Parallel TestNG Execution :**

Parallel testing or parallel execution, as the name suggests, is a process of running the test case parallelly rather than one after the other. Parallel execution would give us the correct idea of the stability and performance of the software much faster than running serially.

Parallel testing is used heavily with [Selenium](https://www.toolsqa.com/selenium-tutorial/) because of the importance of [cross-browser testing](https://www.toolsqa.com/blogs/importance-of-cross-browser-testing-and-how-it-is-performed/) in the market today

Advantages of Parallel Testing :-

* Reduces Time: Running the tests in parallel reduces the overall execution time.
* Allow Multi-Threaded Tests: Using the parallel execution in TestNG, we can allow multiple threads to run simultaneously on the test case providing independence in the execution of different components of the software.

*Where can we apply parallel testing in TestNG?*“

Answer : There are 4 reasons/ ways where we can apply this.

* *Methods: This will run the parallel tests on all @Test methods in TestNG.*
* *Tests: All the test cases present inside the <test> tag will run with this value.*
* *Classes: All the test cases present inside the classes that exist in the XML will run in parallel.*
* *Instances: This value will run all the test cases parallelly inside the same instance.*

*Parallel test execution in TestNG triggers with the help of keyword “***parallel**.”

*We have to add parallel tag in TestNG.xml file. Check the code in Eclipse, we have added a statement for the parallel test like :*

<test name=*"Parallel Test"* parallel=*"methods"*>

*We have added in test tag, but given methods in parallel.*

1. Java script code to enter any text in text field :

WebElement textarea = driver.findElement(By.xpath());

JavascriptExecutor js = (JavascriptExecutor) driver;

Js.executeScript(“arguments[0].innerHTML = ‘<<<Type data to enter>>’ “, textarea)

1. I was not getting name() method in below line, I have added dependency of apache poi 3.10 .. I have to add 4.1.1 dependency to get this method.

System.***out***.println(sheet.getRow(j).getCell(k).getCellType().name());

Below are the dependencies need to be added :

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-ooxml</artifactId>

<version>4.1.1</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi</artifactId>

<version>4.1.1</version>

</dependency>

1. **TestNG Listeners :**
2. **Java Script method to highlight any element :**
3. **Appium – Mobile Automation –**

**Ans :**

* We have to download Appium V 1.15.1. We can just download it from Google. Not a prob
* We have to download Node JS V 13.8.0
* We have to download Android Studio, since we have to work with Android phones more.
* Now, just launch Android studio, then open AVD Manager, Select the type of device we want to use, select pixels and all and finally Install it. Run the virtual Android device.
* Mobile (Emulator) will launch on the screen. It will be in same shape and length as the real device.
* Run Appium Server as well.
* We have to set the path as well of Android studio. If studio got successfully installed in machine, just go to C Drive, Users, search for “App Data” folder. Mostly it will be hidden. Then, go inside bin folder. We have to set path for ANDROID\_HOME AND bin path
* We have to use “UI Automation Viewer”. This is to give the replica of the mobile handset. It helps us to find the xpath and web elements. Its like replica of any application visible in Mobile to web.

1. **Verify Element present in Selenium IDE –**

**Ans**:

We can use following two commands to verify the presence of an element:

* verifyElementPresent - returns TRUE if the specified element was FOUND in the page; FALSE if otherwise
* verifyElementNotPresent - returns TRUE if the specified element was NOT FOUND anywhere in the page; FALSE if it is present.

We can use below codes to verify presence of a certain text :

* verifyTextPresent - returns TRUE if the specified text string was FOUND somewhere in the page; FALSE if otherwise
* verifyTextNotPresent - returns TRUE if the specified text string was NOT FOUND anywhere in the page; FALSE if it was found

We can use below code to verify specific position of an element.

* verifyElementPositionLeft - verifies if the specified number of pixels match the distance of the element from the left edge of the page. This will return FALSE if the value specified does not match the distance from the left edge.
* verifyElementPositionTop - verifies if the specified number of pixels match the distance of the element from the top edge of the page. This will return FALSE if the value specified does not match the distance from the top edge.

1. **getOptions() method in Selenium**

**Ans**:

Select dropdown = new Select(driver.findElement(By.id(dropDownId)));

List<WebElement> options = dropdown.getOptions();

1. **Working with iframes in Selenium –**

Ans : if we have to switch to frame, then use below command :

Driver.switchto().frame(“<<frame id or name or index>>”);

Driver.findelement(….. )

Now if we want to switch back to the content page :

Driver.switchto().defaultContent();

Then again we can write

driver.switchTo().frame("frame2");

Now, if suppose there are multiple frames, then we can’t move from 1 frame to another. We have to come back to the main page. Use above commands only.

1. **How To Handle SSL Certificate Error In FF For Selenium WebDriver**

Ans : Sometimes, When you access site manually In browser It will works fine but shows "This Connection is Untrusted" error when you access same URL through webdriver. It Is working fine In default system browser because you already Imported required certificate In that browser. That error will not appear once you Import required certificate In browser. But as you know, Selenium webdriver opens fresh profile every time you run test so that browser do not have required SSL certificate.

Steps to resolve it :

* 1. Create new firefox profile with name = "certificateIssue
  2. Start browser using that "certificateIssue" profile manually.
  3. Access the URL In which you are getting SSL certificate error. It will show you "This Connection is Untrusted" error.
  4. Import certificate by clicking on I Understnad the Risks -> Add Exception button. So error will disappear and URL will be opened In new profile browser.
  5. ProfilesIni is a class

ProfilesIni firProfiles = new ProfilesIni();

FirefoxProfile wbdrverprofile = firProfiles.getProfile("certificateIssue");

wbdrverprofile.setAcceptUntrustedCertificates(true);

wbdrverprofile.setAssumeUntrustedCertificateIssuer(false);

WebDriver Driver = new FirefoxDriver(wbdrverprofile);

Driver.get("site URL where certificate error");

1. **GetwindowHandle() and getwindowHandles() method in Selenium –**

**Ans :** Concept is :

1. getWindowHandle() returns the window handle of currently focused window/tab. getWindowHandles() returns all windows/tabs handles launched/opened by the same driver instance including all parent and child window.
2. Return type of getWindowHandle() is String while return type of getWindowHandles() is Set<String>. The return type is Set as window handle is always unique.
3. getWindowHandles() internally uses LinkedHashSet. So whatever Set it returns, it will give window handles in the order it is opened.

Set<String> AllWindowHandles = driver.getWindowHandles();

String window1 = (String) AllWindowHandles.toArray()[0];

String windowHandle = driver.getWindowHandle();

driver.switchTo().window(windowHandle);

1. **How to find an element that displays and hides within few seconds in Selenium Webdriver ?**

**Ans** : We have to follow certain steps –

1. Do that n chrome, open Development tools (ctr+shift+I)
2. Do that action, which will display that error message popup for few secs
3. As soon as element displays, press F8 button (this will pause the DOM, it’s a debugger).
4. Then we can easily inspect the element.
5. **IMP – If we get this error java.lang.NoClassDefFoundError: org/apache/http/ssl/TrustStrategy**

**Ans :**

We have to add below dependency in our pom file to resolve this issue :

<dependency>

<groupId>org.apache.httpcomponents</groupId>

<artifactId>httpclient</artifactId>

<version>4.5.3</version>

</dependency>

1. **Switching between Tabs in Selenium / How to open a new tab in Selenium**

Ans : we have to set the Selenium 4 maven dependency in our pom file

Post that, we just need to write below code -

driver.switchTo().newWindow(org.openqa.selenium.WindowType.***TAB***);

System.***out***.println(driver.getTitle()); // This will give title of new Tab

driver.get("https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java/4.0.0-alpha-3");

// Now we have 2 tabs open and want to switch between 2 tabs. So will use the technique of windows handler. We’ll use the parent and child IDs to do that.

Set<String> handles = driver.getWindowHandles();

List<String> ls = new ArrayList<String>(handles);

driver.switchTo().window(ls.get(0)); // We are in Parent Tab

driver.get("https://www.youtube.com/watch?v=7SpCMkUKq-Y&t=241s");

driver.switchTo().window(ls.get(1)); // We are in child Tab

driver.get("http://www.software-testing-tutorials-automation.com/2015/02/how-to-open-tab-and-switching-between.html");

Other way to open a new Tab is using (Keys.CONTROL + “t”) technique

driver.findElement(By.cssSelector("body")).sendKeys(Keys.CONTROL +"t");

driver.get("http://only-testing-blog.blogspot.com/2014/05/form.html");

1. **contains() and Starts-with function of Xpath – Need to do some practical examples.**
2. **How to do Parallel Execution in Cucumber –**

**Ans** : There are various ways to do that –

1. Using Testng xml file –

Need to add below code in Test Runner class first –

@Override

@DataProvider(parallel = **true**)

**public** Object[][] scenarios() {

**return** **super**.scenarios();

}

This will come inside TestRunner class like below -

**public** **class** TestRunner **extends** AbstractTestNGCucumberTests {

@BeforeClass

**public** **static** **void** setUpSuite() {

System.*setProperty*("jagacy.properties.dir", "FeatureFiles");

System.*setProperty*("test.window", "true");

}

@Override

@DataProvider(parallel = **true**)

**public** Object[][] scenarios() {

**return** **super**.scenarios();

}

}

Then, we have to add below statement in testng.xml file,

<suite name=*"TestingSuite"* data-provider-thread-count=*"2"*>

Here, we added on Suite level, its running 2-2 Test case scenarios of Single feature file

Now, I am running only 1 feature file, it has suppose 2 scenarios, then it ran both Scenario in parallel.

Next Scenario -

1. **Diff Type of Selenium Wait commands & Wait to check the visibility of any Web element –**

**Ans :** For this we have a method *visibilityOf* in Class ExpectedConditions)

Code will be :

WebDriverWait wait = **new** WebDriverWait(driver,100);

wait.until(ExpectedConditions.*visibilityOf*(weblement)); // This is the WebELement obj

In automation testing, wait commands direct the test execution to pause for a certain length of time before moving onto the next step.

We have 3 types of Wait commands available in Selenium –

* Implicit Wait
* Explicit Wait
* Fluent Wait

Implicit Wait - Implicit Wait directs the Selenium WebDriver to wait for a certain measure of time before throwing an exception. Once the command is in place, Implicit Wait stays in place for the entire duration for which the browser is open ie,. It has to wait for the given time if element is found even. It’s drawback is it increases execution time unnecessarily.

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

Explicit Wait - By using Explicit Wait command, the WebDriver is directed to wait until a certain condition passes before proceeding with executing the code. ie. It only wait till the driver searched the web element, then it proceed to further steps of execution. 1 condition is, it can be applied only for specified elements.

In order to declare explicit wait, we have to use “ExpectedConditions” class of Selenium.

Code will be :

WebElement element = driver.findElement(By.*id*("test"));

WebDriverWait wait = **new** ~~WebDriverWait~~(driver, 20);

// Now we have 2 methods to define this wait, either we can use visibilityOf(WebElement element) like below :

wait.until(ExpectedConditions.*visibilityOf*(element));

// We have 1 more method to use, visibilityOfElementLocated(By locator), like below

wait.until(ExpectedConditions.*visibilityOfElementLocated*(By.*xpath*("test")));

// So, here above 2 statements will execute only till the visibility of the Web Element given in it within 20 secs.

// but if the element still doesn't get fine out after 20 secs, then TimeoutException will come out.

element.click();

// Once element got find out, it will execute the next steps.

Fluent Wait - Fluentwait in selenium webdriver in one of the examples of the dynamic wait which will solve many sync issues in Selenium Webdriver. It basically that wait which we can handle as per our conditional req.

One unique diff b/w Fluent wait and other wait methods are in Fluent wait, we can change the default polling period based on our req.

1. **IMP Topic – Keyboard operation from Selenium Code :**
2. Enter Operation – We can use **Keys Enum** for this operation. There are 2 methods, Keys.RETURN and Keys.ENTER. We can use any of these 2 operations to do that. Code will be :

driver.get("https://accounts.google.com/login/signinchooser?flowName=GlifWebSignIn&flowEntry=ServiceLogin");

driver.findElement(By.*id*("identifierId")).sendKeys("mittal.tusar@gmail.com");

// Both the below methods will be use to do the Enter operation of keyboard from Java code

//driver.findElement(By.id("identifierId")).sendKeys(Keys.RETURN);

driver.findElement(By.*id*("identifierId")).sendKeys(Keys.***ENTER***);

1. If we want to delete or remove all the characters of a text field then we have 2 below options –

Keys.DELETE .. We have to use below code

driver.get("https://accounts.google.com/login/signinchooser?flowName=GlifWebSignIn&flowEntry=ServiceLogin");

WebElement emailID = driver.findElement(By.*id*("identifierId"));

emailID.sendKeys("mittal.tusar@gmail.com");

emailID.sendKeys(Keys.***CONTROL***,"a");

emailID.sendKeys(Keys.***DELETE***);

The above code will delete all the text from that particular text field.

Now, other option is to use Keys.BACK\_SPACE

emailID.sendKeys(Keys.BACK\_SPACE);

But thing is this method will just remove only 1 character at a time not the whole text.

1. Tab Key – Keys.TAB
2. Other Keys functions :

PgUp Key Keys.PAGE\_UP

PgDn Key Keys.PAGE\_DOWN

Spacebar Keys.SPACE

Arrow Key – Down Keys.ARROW\_DOWN

Arrow Key – Up Keys.ARROW\_LEFT

Arrow Key – Left Keys.ARROW\_RIGHT

Arrow Key – Right Keys.ARROW\_UP

1. **Exceptions in Selenium --**

**Ans** : Below are the exceptions :

1. **ElementNotVisibleException -** In spite of the element being present in the DOM, it is not visible (can not be interactive). For example, elements defined in HTML with type =”hidden”
2. **NoSuchElementException**: Webdriver is not able to determine the elements during runtime, i.e., the FindBy method cannot find a particular component
3. **NoSuchFrameException**: Webdriver attempts to switch to an invalid frame, which is unavailable
4. **NoSuchWindowException**: Webdriver is trying to switch to an invalid window, which is unavailable
5. **StaleElementReferenceException**: The referenced element is no longer present on the DOM page (a reference to a component is now Stale). For example, the item belongs to a different frame than the current one or the user has navigated away to another page.
6. **SessionNotFoundException**: Webdriver is acting immediately after ‘quitting’ the browser
7. **TimeoutException**: The command did not complete in the specified time. For example, the element didn’t display at the specified time. This is especially encountered when working with waits
8. **WebDriverException**: Webdriver is acting immediately after ‘closing’ the browser
9. **Invalid Argument Exception** – If the url
10. **Alert Class in Selenium / How to deal with Pop-up :**

**Ans** : Will use **Alert Interface** to deal with pop-up like below :

driver.switchTo().alert().getText(); // to get text of the message

driver.switchTo().alert().accept(); // to click on OK button

driver.switchTo().alert().dismiss(); // to click on Cancel button

driver.switchTo().alert().sendKeys("Test data"); // to send any data in it

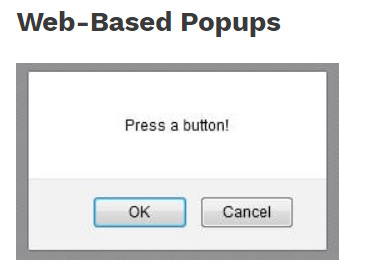
Also, we should know the types of Pop-up

1. Simple Alert- pop-up which has OK button to click.
2. Prompt alert – pop-up which has OK and Cancel button, also have text field to enter as well.
3. Confirmation alert – same like Prompt alert only, not much diff.

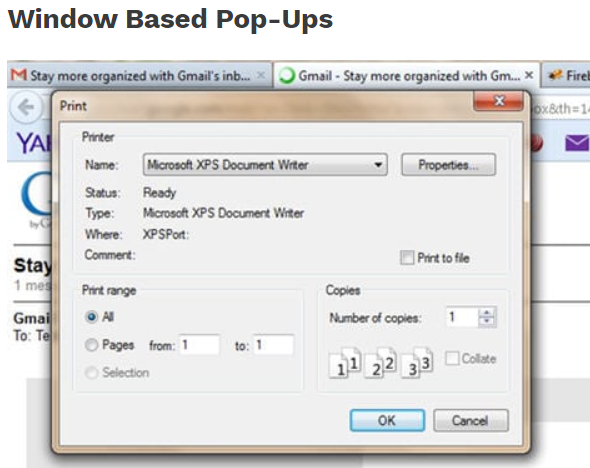
There are mainly 2 types of alerts available :

* Window based alert pop-ups
* Web based alert pop-ups

**IMP Point to note** : We can’t handle window based pop-up through Web Driver. What we have studied above, those are web based pop-up ie. Simple Alert…..



How to handle Window Based pop-up :



Window based pop-up like Print pop-up or a browsing window while uploading a file.

We have to use **Robot class** to handle window based pop-ups.

**What is Robot class ?**

**Ans : Robot Class** is used in Selenium because, in certain Selenium automation tests, users need control over keyboard or mouse to interact with OS windows like download pop-ups, print pop-ups, etc. and native applications like notepad, calculator, etc. Selenium Webdriver cannot handle these pop-ups/applications, so in[Java](https://www.guru99.com/java-tutorial.html)version 1.3, robot class was introduced which can handle OS pop-ups/applications.

Robot class is not from Selenium package (org.openqa.selenium). Its from JAVA API awt package.

We can use Robot class for below mentioned things :

* Keyboard and Mouse events
* Upload and download of files

**We’ll see Window based Upload and Download pop-up code using Robot class –**

driver.get("https://tus.io/demo.html");

Thread.*sleep*(3000);

driver.findElement(By.*xpath*("//input[@id='js-file-input']")).click();

Robot rb = **new** Robot();

ClipboardOwner owner = **null**;

String fileLocation = "D:\\Document Copy\\practicePerfect.docx";

StringSelection sb = **new** StringSelection(fileLocation);

Toolkit.*getDefaultToolkit*().getSystemClipboard().setContents(sb,owner);

rb.setAutoDelay(4000);

rb.keyPress(KeyEvent.***VK\_CONTROL***);

rb.keyPress(KeyEvent.***VK\_V***);

rb.keyRelease(KeyEvent.***VK\_CONTROL***);

rb.keyRelease(KeyEvent.***VK\_K***);

rb.setAutoDelay(5000);

rb.keyPress(KeyEvent.***VK\_ENTER***);

rb.keyRelease(KeyEvent.***VK\_ENTER***);

rb.setAutoDelay(5000);

**2 main methods of Robot class –**

* **keypress – use to press an key of keyboard**
* **keyRelease – use to release press of any keyboard key. Suppose like Shift key, Basically when we hold shift key and click on some word, then only it works. Now, to proceed further we have to release that press as well. Press of that Shift key. In that case, we have this method available.**

1. **Select Class and Actions Class for drop-downs :**

WebElement ele = driver.findElement(By.*xpath*("test"));

Select select = **new** Select(ele);

List<WebElement> allValues = select.getOptions();

select.selectByIndex(6);

**for**(**int** i=0;i<allValues.size();i++) {

allValues.get(i).getText();

}

System.***out***.println(select.isMultiple()); // to check whether the drop-down is multi select

1. **Action Class in Selenium –**

Ans : Complex functions like Drag and drop or Double click will be done by Actions class. Also, this class can be used to do Keyboard and Mouse operations.

Actions class is the collection of individual Action that we need to perform.

Actions class provides methods for keyboard movements like:

* Pressing Shift Key : Actions Class Method => keyDown
* Sending desired text : Actions Class Method => sendKeys
* Releasing Shift key : Actions Class Method => keyUp

The keyDown method performs a modifier key press after focusing on an element, whereas keyUp method releases a modifier key pressed.

After the above actions, we need to call .build() and .perform() method as well.

.perform() method have to take place after all the action sequences. Once this is done, then only we can see action is performed.

**Note** : .build() method is internally called in perform() method itself. Use it or don’t. not a prob

driver.get("https://demoqa.com/auto-complete/");

WebElement element = driver.findElement(By.*id*("autoCompleteMultipleInput"));

Actions action = **new** Actions(driver);

Thread.*sleep*(3000);

// Suppose req is to enter caps letter in a text field.

action.keyDown(element,Keys.***SHIFT***);

action.sendKeys("test");

action.keyUp(Keys.***SHIFT***);

Thread.*sleep*(1000);

// We can use any of the below steps, since build() is performing inside perform() method only

//action.build().perform();

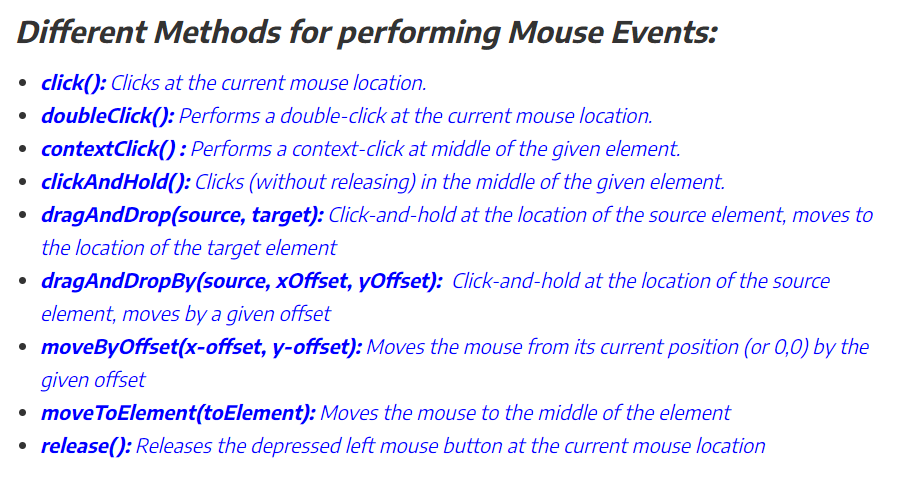
action.perform();

Thread.*sleep*(3000);

So, 2 main categories of methods of Action class is :

* ***Keyboard Events***
* ***Mouse Events***

List of all Mouse Actions which can be done by Action class -



**Right click of Mouse in Selenium –**

We’ll use Action class to do the same. Method available is contextClick(WebElement)

So logic is, find the element on which we need to do right click, that element we need to pass on to contextClick() method. After that just invoke perform() method like below :

actions.contextClick(webElement).perform();

WebElement element = driver.findElement(By.*id*("autoCompleteMultipleInput"));

Actions action = **new** Actions(driver);

// Right click on Mouse

action.contextClick(element).perform();

**Double Click of Mouse in Selenium**

**Action class method doubleClick(WebElement)**is required to be used to perform this user action.

driver.get("https://demoqa.com/buttons");

action.doubleClick(element).perform();

**Drag and Drop function –**

https://demoqa.com/droppable/

actions.dragAndDrop(source,target).perform();

**Mouse Hover :**

driver.get("https://www.toolsqa.com/selenium-webdriver/mouse-hover-action/");

WebElement element = driver.findElement(By.*xpath*("//nav[@class='navigation']//span[text()='Tutorial']"));

Actions action = **new** Actions(driver);

action.moveToElement(element).perform();

**Tool Tip –**

It can be of 2 cases, Case 1 – When tool tip is available in title attribute

Case 2 – When tool tip is available in div attribute

Tool tip can be searched even directly as well. Or we can use Action class as well

Like,

Action.movetoElement(WebElement element).perform();

Driver.findelement…. – that element xpath

Then just print getText().

1. **Program for Scroll bar in Selenium –**

((JavascriptExecutor) DriverFactory.*getWebDriver*()).executeScript("arguments[0].scrollIntoView(true);",element);

We have to pass Webelement element object.

1. **How to take Screen shots in Selenium ?**

**Ans** :

TakesScreenshot schShot = (TakesScreenshot) driver;

File file = schShot.getScreenshotAs(OutputType.***FILE***);

File Permfile = **new** File("File path");

FileUtils.*copyFile*(file, Permfile);

1. **Headless Browser Testing using Selenium WebDriver**

**Ans :**

A headless browser is a browser which doesn’t have a GUI. It runs programs even if we don’t have that particular browser installed in machine. When the programs, it will run in the background, we won’t be able to see any GUI running the Automation script.

There are 2 headless drivers available :

HtmlUnitDriver

PhantomJSDriver

Its adv : is a) We can save time, execution is fast with above headless browsers.

**Main adv** is imagine we want to run the Test on certain Chrome version which is not available as of now, in that case we can use Headless browsers.

Html Unit Driver is currently the fastest and most lightweight implementation of WebDriver. As the name suggests, this is based on HtmlUnit. HtmlUnit Driver is similar to the other drivers such as Mozilla Firefox, Google Chrome, Internet Explorer but you couldn’t see the GUI of Html UnitDriver.

WebDriverManager.*chromedriver*().version("83.0.4103.39").setup();

ChromeOptions options = **new** ChromeOptions();

WebDriver driver1 = **new** HtmlUnitDriver(options);

System.***out***.println("Entry1");

1. **Difference between ChromeOptions and Desired Capabilities ?**

Ans:

**Answer** 1 : ChromeOptions class has introduced in the latest/updated version of Selenium. It is helpful to make changes in the Chrome browser whereas, DesiredCapabilities is an old concept (its usage in Java is deprecated.) to configure or make changes in the browser.

But we can use both DesiredCapabilities and Chromeoptions class via merge method. Pls find below code:

DesiredCapabilities cap = new DesiredCapabilities(); // instantiate DC Class

ChromeOptions options = new ChromeOptions(); // instantiate CO Class

options.merge(cap); // passing DC class object to CO Class

driver = new ChromeDriver(options);

**Answer 2** - chromeOptions - is used in Selenium to customize the Chrome browser (majorly in Java)

desiredCapablities - was used earlier in Selenium using Java (now its deprecated and not being used. (now DesiredCapablities are used in Appium driver for Mobile Automation)

**Answer 3 – Imp Answer** - Chrome options class is used to manipulate various properties of Chrome driver. Chrome options class is generally used in conjunction with Desired Capabilities.

For ex : TO maximize the browser

Use,this code

ChromeOptions options = new ChromeOptions()

options.addArgument("start-maximized");

ChromeDriver driver = new ChromeDriver(options);

Below are the list of available and most commonly used arguments for ChromeOptions class

* **start-maximized**: Opens Chrome in maximize mode
* **incognito:**Opens Chrome in incognito mode
* **headless:** Opens Chrome in headless mode
* **disable-extensions**: Disables existing extensions on Chrome browser
* **disable-popup-blocking**: Disables pop-ups displayed on Chrome browser
* **make-default-browser:** Makes Chrome default browser
* **version**: Prints chrome browser version
* **disable-infobars:** Prevents Chrome from displaying the notification 'Chrome is being controlled by automated software