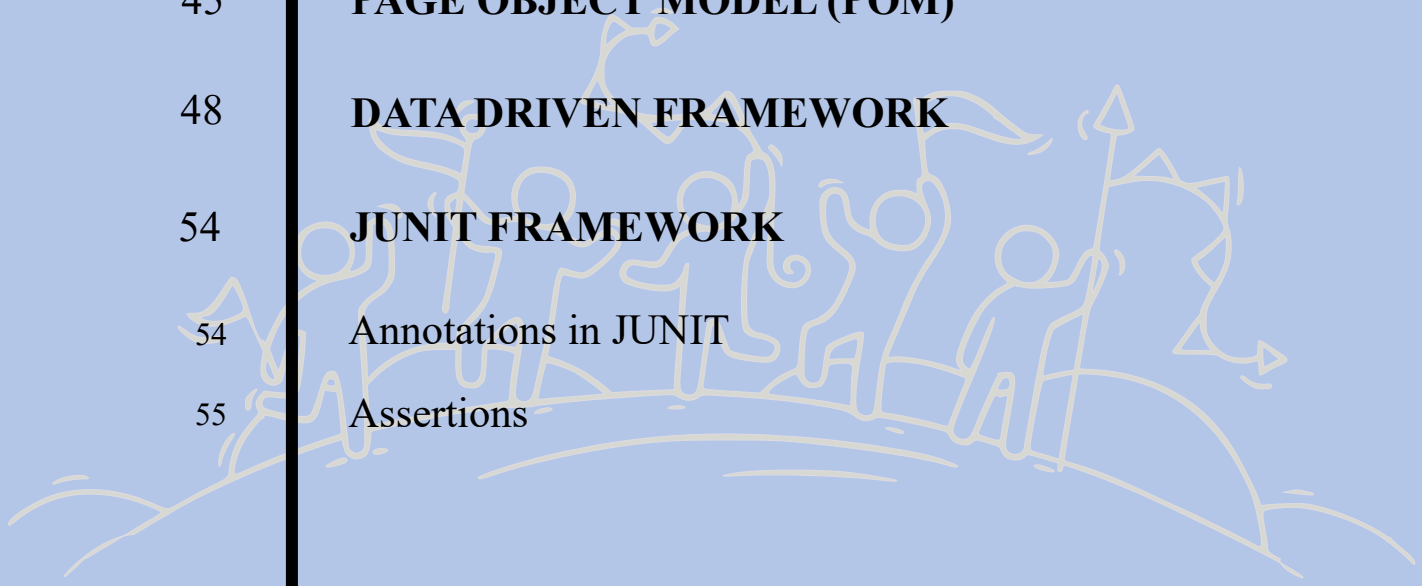
# JAVA SELENIUM

Over All Topics





# JAVA SELENIUM

Over All Topics

# J A V A  T O P I C S

**OOPS**:

---

OOPS is Object Oriented Programming Structure, it has

- class, method, object
- Encapsulation
- Inheritance
- polymorphism
- Abstraction

In our project we have implemented OOPS concepts lots of places

**Class**:

=======

1.Class is a collections of methods and objects

Object is an instance of a class

2.In our project We are using lots of predefined classes like

1.ChromeDriver

2.FireFoxDriver

3.InternetExplorerDriver

4.Actions

5.Robot

6.Select

7.RemoteWebDriver

**Methods**:

======

1.Method is a set of action to be performed

2.We used to write the business logics in methods

3.In our project We are using lots of predefined Methods like

get()

getTitle()

getCurrentUrl()

getText()

getAttribute()

**Encapsulation:(Data hiding)**

==========

Wrapping up of data and code acting on data in single unit called encapsulation

(or)

Also we can say the Structure of creating folders are called Encapsulation

Encapsulation concept used while implementing POM framework in my project.

1.In our project we used to Maintain the all locators in the POJO class as object repository

In POJO class i ll declare all my variables as private and using

getters we can access the locators in the differnt class using page object Model(POM) with pageFactory concept

**Inheritance**:

========

1.We can access one class property(methods) into another class without multiple object creation using extends keyword.

2.In our project,We used to maintain the all resuable methods(sendKeys,click) in base class and using extends keyword

we can access all the methods wherever its required

**Polymorphism**:

==========

**Method Overloading**:

==========

Multiple methods with same method name and the methods will differ only based on its argumemts datatype,datatype count and datatype order.

(or)

The Same method is going to act as different behaviour that is called method overloading

(or)

Same method is going to overload again and again with the different arguments.

2.In our project we are using method overloading concepts like
println(),sendKeys(),frame(),Waits(SECONDS,MINUTES,HOURS)

Eg:

===

sendkeys("abc123")

Actions ----a.sendkeys(WebElement,String)

frame(int index)

frame(String name)

frame(String id)

frame(WebElement ref)

**Method Overriding**:

==========

When we are not satisfied with the business logic of our superclass method we can override the same method in my subclass.

When we create object for child class and if we try to access the method only child class method will get preference.

(or)

Changing the business logic from super class methods into subclass using same method name.

1.Method overriding is used to provide the specific implementation of a method which is

already provided by its superclass.

2.In our project we implemented method overriding concepts like Userdefine

1.User define Exception(getMessage()---Exception class)

2.TakesScreenShot--getScreenShotAs()

3.IRetryAnalyzer---retry()

4.IAnnotationTransformer---transform()

5.List---(get(),indexOf())

6.Set----(add(),remove())

7.Map----(put,containsKey())

**ABSTRACTION**:

---

Hiding the implementation(business logic)

Abstract class:(we can acheive partial abstraction using Abstract class)

==========

1.It supports both abstract methods(there will be no business logic) and non abstracts method(with business logic)

2.Using extends keyword we write our business logics for abstract methods into any other where we are implementing this Abstract class.

3.In our project we are using By as abstract class

**Interface**:(we can achieve full abstraction using interface)

==========

1.Its supports only abstract methods

2.Using implements keyword we can write the implementations for abstract methods in whichever class we are implemeneting the

interface

3.In our project we are using lots of interfaces

1.SearchContext

2.WebDriver

3.JavaScriptExecutor

4.TakesScreenShot

5.IRetryAnalyzer

6.IAnnotationTransformer

7.Alert

8.WebElement

9.Wait

10.List

11.Set

12.Map

13.OutputType

# Inheritance

---

we can access one class property(method) from another class using extends keyword.

Why we go for inheritance

---

reusable code purpose.

memory wastage is low.

It use a keywords extends

public class ChildClass extends ParentClass

Types of inheritance:

==========

- **Single inheritance**: Combinations of one parent class and one child class.
- **Multiple inheritance**: More than one parent class access the child class paralelly at a time.

- **Multilevel inheritance**: More than one parent class accesing the child class in a tree level structure
- **Hierarchical inheritance:** Combination of one parent and more than one child.
- **Hybrid inheritance**: Combination of Single and Multiple inheritance

# Polymorphism

---

Executing methods in more than one form

poly--- many

morphism ---forms

1) **Method overloading** (compile time polymorphism)/ static binding/static polymorphism

---

same class

same method name

diff arguments

==========

arguments depend on datatype

arguments depend on datatype count

arguments depend on datatype order

2) **method overriding** (runtime polymorphism) / dynamic binding/dynamic polymorphism

---

Same method

Same argument

diff class

# Abstraction

---

hiding the implementation details or business logic details.

Types of abstraction:

---

- **partial abstraction**
- **Fully abstraction**

**partial abstraction** (abstract class):

---

contain both abstract and non abstract methods

contain keywords extends

use a keyword abstract in both class and abstract method

we can't create object

Don't have any default return type

**fully abstraction** (Interface)

---

contain only the abstract methods

contain keywords implements

use a keyword interface instead of class

we can't create object

Default return type is public abstract

# Access specifier

It will specify the extent of access

==========

private--class level access specifier (It will support only within the same class)

default--same package as well as different package level access specifier

(only implements keywords. default methods will be supported only in interface.

Protected—samepackage (extends, object) + different package (extends)

public--global level access specifier------samepackage (extends, object) + different package (extends, object)

# Access Modifiers

It will modify the extent of access

---

abstract

static

final

Abstract

---

Class

---

If we declare class as abstract we cannot create object for the class

method

---

If we declare method as abstract we canot write any bussiness logic for that method

variable

---

we canot declare variable as abstract

Static

---

Class

---

we can't declare class as static

method

---

If we declare method as static no need to create object we can call directly by a method name

In different class using extends we can directly call by a method name

without using extends "Classname.methodname();"

variable

---

If we declare variable as static no need to create object we can call directly by a variable name.

We can use the variable in throughout class.

In different class using extends we can directly by a variable name

without using extends "Classname.variablename".

Final

---

Class

---

we declare class as final we can't inherited

method

---

we declrae method as final we can't overrided

variable

---

we declare variable as final we can't modified.

# Array

We can store multiple values of similardatatypes in a single variable

Similar datatypes

Index based

Index starts from 0 to n-1.

It will support duplicates

Array syntax

==========

DataType variable [ ] = new Datatype [size];

Enhanced for loop/ for each

==========

(DataType Variable name: Stored Variable)

{

}

==========

Disadvantages of Array

---

- We can store only the similar datatypes
- once we fixed the size we can't modified
- memory wastage is high
- Compile time memory allocation

Different Between for loop and for each loop

| FOR LOOP | FOR EACH LOOP |
|---|---|
| index based | Value based |
| Control the iteration | Can't control iteration |
| initialization, condition, inc/dec | no initialization, condition, inc/dec |
| IndexOutOfBoundException | No Exception |

# Constructor

---

Class name and constructor name must be same.

It doesn’t have any return type.

We don’t want to call constructor which is creating object itself.

It will automatically invoke the default constructor.

It will support in method overloading but won’t support in method overriding

Syntax:

=======

public constructor_name()

{

}

Types:

======

1. Non-Parameterized constructor/default constructor

2. Parameterized constructor/argument based constructor.

**Constructor chaining**

---

To call one constructor to another costructor is called constructor chaining

Also to reduce the no of object creation

keywords

---

this()--> to call current class constructor

super()--> to call parent class constructor

# Data types

It specifies the type and size of the variable.

Variable is used to store the values.

---

| **Datatype** | **size(Byte)** | **Wrapper class** | **Default values** |
|---|---|---|---|
| byte | 1B (n=8) | Byte | 0 |
| short | 2B (n=16) | Short | 0 |
| int | 4B (n=32) | Integer | 0 |
| long | 8B (n=64) | Long | 0 |

used for whole numbers

| | | | |
|---|---|---|---|
| float | 4B | Float | 0.0 |
| double | 8B | Double | 0.0 |

used for decimal numbers/fractional numbers

| | |
|---|---|
| char ('A') | Character |
| String ("hello") | String (alphabets, numerals, special characters) null |
| Boolean | Boolean (True/false) false |

Range calculating formula

---

1 byte = 8 bits n= no of bits

**range= -(2^(n-1)) to ((2^(n-1))-1) byte = -(2^7) to (2^7)-1**

= -128 to 127

=-32768 to 32767

**Wrapper class**

---

class of data type is called wrapper class

long ph=123l;

float fo=23.56f;

Variable Declaration: SYNTAX

---

Datatype variableName = value;

Datatype a=10

=--->assignment operator

# COLLECTIONS (Interface)

==========

Why we go for collections:

It will support dissimilar data types.

It is dynamic memory allocation or Run time memory allocation

No memory wastage like array

It has 3 types,

1. List----------Interface

2. Set----------Interface

3. Map----------Interface

Class a = new Class();

Interface a = new Class();

# List :( Interface)

==========

It is used to store multiple values of dissimilar data types in a single variable.

List is index based(0 to n-1)

List will allow the duplicates

List accepts null value

1. **ArrayList(class)**
2. **LinkedList(class)**
3. **vector(class)**

Syntax

List ref = new ArrayList();

List ref = new LinkedList();

List ref = new Vector();

**ArrayList**:

==========

Syntax:

========

List ex=new ArrayList(); Here,

List-interface

ex-object name

ArrayList()-class

**Generics**:

==========

It will support particular datatypes or object only

It is a features of jdk 1.5

In the generics, we can mention only wrapper class

< >- This is generic symbol, is used to define the particular datatype

If we need integer datatype,

syntax:

List<Integer> ex=new ArrayList();

Methods in list

==========

(i) add();

add() is a method, it is used to insert a value.

Arraylist will display the output based on the insertion order

(ii)size();---- to find the length of the list

(iii) get();-----to get a particular value for a given index

(iv) Remove();-----to remove a particular value from the list

(v) add();----->indexed based add();

(vi) set();

(vii) contains();

(viii) clear();

(ix) indexOf(); lastIndexOf();

(x) addAll(); removeAll(); retainAll();

forloop

foreach

ArrayList: Worst case

In ArrayList deletion and insertion is a worst one because if we delete/insert one index value after all the

index move to forward/backward.

It makes performance issue.

ArrayList: Best case

In Arraylist retrieve/searching is a best one For ex we have 100 index is there, if we going to print 60th value,

we can easily search

LinkedList: Best case

Insertion and deletion is a best one because Here all values based on the seperate nodes. so,

here we can easily delete/insert one value(i.e) if we delete one value, the next node will join to the previous one

LinkedList: Worst case

Searching/retrieving is a worst For ex, if we have 100 nodes, we have to print 90th node value,

it will pass to all the previous nodes and comes to first and then it will print. It’s makes performance issue

**Difference between ArrayList and Vector:**

ArrayList: Asynchronize It is not a thread safe, parallel execution, fast

Vector: Synchronize Thread safe, sequential execution

# Set:(Interface)

==========

Set is used to store multiple values of dissimilar data type in a single reference

Work based on - value

Duplicates - It Don't allow duplicate

Printing order - based on the classes

Classes of Set

==========

Hashset (class) ---- random order----------one null

Linked hashset (class)------insertion order------one null

Treeset (class)----------ascending order-----doesn't allow null value

Set:

=====

It ignore the duplicate value It is value based

Methods In Set

==========

(i) size();

(ii) remove();

(iii) Only Enhanced for loop

All wrapper class default value is Null as well as all class default value is Null

Null:

=======

Null is a undefined/unknown/unassigned value

Null is won't create any memory

So Treeset will give exception in compile time if we use Null

Copying Values from List to set and Set to List

==========

(i) Finding no of duplicates.

Method Supported By list but not in the set

==========

1, ref.add(index, value);

2, ref.indexOf(value);

3, ref.lastIndexOf(value);

4, ref.set(index, value);

5, ref.get(index);

6, forloop

Methods Present in Both List and Set

==========

1, ref.add();

2, ref.size();

3, ref.addAll();

4, ref.remove();

5, ref.removeAll();

6, foreach

TreeSet ---> Generics------->ClassCastException

**ASCII value**

==========

A to Z - 65 to 90

a to z - 97 to 122

0 to 9 - 48 to 57

# Map (Interface)

---

key, value pair combination

key Don't allow duplicates

values allow duplicates

Types of map or Classes of Map

---

HashMap----Random ----> key=1 null values= n null

LinkedHashMap----Insertion---->key=1 null values= n null

TreeMap----Ascending order---->key=ignore null values= n null

Hashtable----------Random---->key=ignore null values= ignore

Methods of map

---

put()-- to insert the values

getKey()----displaying the corresponding keys values.

getValues()----displaying the corresponding values.

values()----displaying the values only and its return type is collection.

keySet()----display the keys only and its return type is set.

entrySet()--for iterating the map and its return type is set<Entry<>>

**Difference between hashtable and hashmap**

| Hashmap | HashTable |
|---|---|
| Asynchronized (Allows users parallelly) | Synchronize (Allows users one by one) |
| key allow 1 null in key | Ignore null values in key and values. |
| Non thread safe | Thread safe |

null----It is undefined unknown value don't occupy any memory

# Control Statements

==========

These statements are used to validate a condition and if the condition is satidfied then the programme will execute and if the condition is failed then the programme will not execute.

There are four control Statements

(i) if

(ii) if else

(iii) nested if

(iv) switch case----Jumping statement

if

==

Syntax:

```
if(condition){
business logic;
}
```

Condition :  passed --- business logic will get executed

failed ----business logic will not get executed

if else

========

Syntax:

```
if(condition){
business logic;
}
else{
business logic;
}
```

Condition : passed --- business logic will get executed

failed --business logic inside else block will get executed

**Operators**

=========

In order to give more than one condition we go for operators

(i) Logical Operators

a) Logical AND operator(&&)

AND (Multiplication) T-1, F-0

| Condition 1 | Condition 2 | Result |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

b) Logical OR operator(||)

OR (Addition)

| Condition 1 | Condition 2 | Result |
|---|---|---|
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

(ii) Bitwise Operators

a) Bitwise AND operator(&)

| Condition 1 | Condition 2 | Result |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

b) Bitwise OR operator

| Condition 1 | Condition 2 | Result |
|---|---|---|
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

Summary Bitwise and logical operators will give same result Logical AND operator will check the first condition if the first condition is false then it will directly give the result as false, if the first condition is true then it will check the second condition

Logical OR operator will check the first condition if the first condition is truethen it will directly give the result as true, if the first condition is false then it will check the second condition

Bitwise operators will check both the condition whether the first condition is true or false

(iii) nested if

Syntax:

```
if(condition){


business logic;

}

else if(condition){

business logic;


}

else{

business logic;

}
```

## Exception: ----> Superclass

---

It is like a error, Whenever it occur the program will terminate itself.

Types of exception:

---

1. **Unchecked Exception**
2. **Checked Exception**

**unchecked exception [run time]**

---

Whenever the exception will occur in runtime it is called run time exception

*Arithmetic Exception

*null pointer exception

*Input mismatch exception

*ArrayIndexOutofbound exception

*StringIndexOutofbound exception

*IndexOutofbound exception

*NumberFormat Exception

**checked exception [compile time]**

---

Whenever the exception will occur in complietime it is called complie time exception

*File not found exception----------> Whenever we create a new file.

*IO exception----------->super class of file not found Exception

*sql exception----------> When unable to fetch data from database.

*Class not found exception-----> When we load a class into another class, if that particular class is not found

Exception/Throwable is the super class of all exception

exception handling

---

(i)try---Whenever the exception will occur, we will use try block, it is used throw an error

(ii)catch--Catch is the block, it is going to catch the exception, it used to handle the error or exception

(iii)finally---It will execute the finally block,it doesnot consider whether the exceptions is handled or not

(iv)Throw

(v)Throws

try,finally

try,catch,finally

try,multiple catch

super class of Exception

sub class----------syntax error

sub class of Exception

super class----------programme will get executed

| Throw | Throws |
|---|---|
| Declared inside a method Block | Declared in the method Signature |
| throw only one Exception | handles Multiple Exception |

# File Operations

==========

To interact with the files, present in local disk or local folder by using java programme

1. Create a new Folder

    Step 1: Create obejct for predefined class called as File(java.io)

    Step 2: Include the path in the obejct to describe the location for folder creation

    Step 3: Use a predefined method called as mkdir();

        mk --make

        dir---directory(folder)

2. Create a new File

        Use a predefined method called as createNewFile();

        This method -- IO Exception

3. Check the given location is file or not

    ref.isFile()

4. Check the file created is readable or not

    ref.canRead()----Boolean

5. Check the file created is editable or not

ref.canWrite()----Boolean

6. Check the files exists in the given location or not

ref.exists()----Boolean

7. Print all the file name and folder name present in a given drive location

String[] s = ref.list();

File[] f  = ref.listFiles();

8. To read the content present in the file

Add commons io jar file -- version 2.7 (Add to Build path)

List<String> ref =FileUtile.readLines();

9. To modify the content present in File

FileUtils. write (fileref, data, boolean);

# STRING

What is mean by String?

==========

*Collections of letters, special charactor or word enclosed with double quotes is called as String.

String is a predefined class .Which is presented in Java.lang package

It is based on index.

*Example : "greenstechnology".

What are the method available in string?  String s ="java";  / String s1="Java ";

==========

*length();-------> size of given string (int)

*isEmpty();---------> empty --true, or else false (true/false)

*charAt();----------->  to find a character at a particular index (Character)

*indexOf();----------> to  find index position of a particular character (int)

*lastIndexOf();----------->to find the last index position of aparticular character(int)

*toUpperCase();----------->to change the given string into upper case letters (String)

*toLowerCase();----------> to change the given string fully into lower case (String)

*startsWith();---------->to check starts with a particular character(true/false)

*endsWith();---------->to chack the ends with a particular character(true/false)

*contains();----------> to check a particular character is present in the string or not(true/false, boolean)

*equals();---------->to compare two strings and both are equal in case.(boolean)

*equalsIgnoreCase();---------->to compare two strings, not case sensitive.(boolean)

*concat();---------->to join two strings (String)

*replace();---------->to replace a particular character(String)

*replaceAll();---------->to replace a complete string(String)

*trim();---------->to delete all characters present in string ("")

*split();----------> to split the given string(String)

*subString();

*compareTo();

# Encapsulation

---

>>>The structure of creating folder

>>It wraps the data & code acting on a data together in to single unit.

Encapsulation is used to hide the values or state of structured data object inside a class,

preventing unauthorised parties direct access to them .

POJO: plain Old Java Object

The class which contain only private variable,constructor, getters & setters

1.Create one class(Employee)

2.Create priavte variables

int id,String name,float salary;

3.Generate getters and setters

getId,SetId,getName,setName,getSalary,setSalary

right click...>>> click--->> getters & setters --->>> Select All -->> ok

Setters --->> Used to set the Value

Getters--->> Used to get the Value

4.Create another class and use collections concept

List<String> s=new ArrayList<>();

Set<String> s1=new LinkedHashSet<>();

Map<interger,String> s2=new LinkedHashMap<>();

List<Employee> s=new ArrayList<>();

Set<Employee> s1=new LinkedHashSet<>();

Map<Integer,Employee> s2=new LinkedHashMap<>();

5.Employee e=new Employee();

e.setId("1223");

e.setName("ABC");

e.setSalary("97789.09");

6.s.add(e);

s1.add(e);

s2.put(1,e)

7.sop(s2.get());

## Singleton

---

Single Object Creation

how to Create Single Object and Overwrite the object again and again

---

Step 1: Declare a Static Variable for the Class

Step 2: Declare a static method

Step 3: Inside the method compare the value of static variable

If the value of static variable is equal to null, then assign an object to the variable

If the value of static variable is not equal to null, then the original variable remains the same.

Step 4: In the return type of the method mention the Class name as the return type

Step 5: In the return value give object as the value.

To prove Singleton

---

Null - It is undefined, unassigned unknown value doesnt occupy any memory

Step 1: Find the memory location of the null value

Step 2: Find the memory locatio of the static variable after creation of object

Step 3: Call the static methods and find the memory location

Whenever we create  a Private constructor we cannot create object for the class

## User Defined Exception

---

Exception- It is a predefined class

Superclass of all exception is Throwable

User defined exception is a user defined class that inherits the super class of Exception

Step 1: Create a Class

Step 2: Extend the Super Class of Exception

Step 3: incur some message for the User defined Exception class

a) Create a constructor for User defined Exception class

b) Inside the constructor use this keyword

c) Create a method to pass the message for the User defined exception class

Step 4: Create one more class

Step 5: Create object for the user defined exception classs inside the try block

Step 6: handle the exception by using the User defined Exception class

null

====

It is undefined, unassigned and unknown value dont occupy any memory in space.

## Scanner Methods

Scanner Notes: (08.09.21)

↳ it is predefined class.

↳ get the input from the user at the Run-time.

↳ it is present in package "java.util".

↳ Default java package is "java.lang".

↳ To use the Scanner class, create an object of the class & use the any of the available methods in Scanner class documentation. Ex: we will use the nextLine(), which is used to read String.

Syntax:-

```
ClassName refName = new ClassName();
```

```
Scanner refName = new Scanner(System.in);
refName.methodName();
```

Scanner Methods:-

1). nextInt(); 2). nextByte(); 3). nextShort(); 4). nextFloat();
5). nextLong(); 6). next(); 7). nextLine(); 8). nextBoolean();
9). nextDouble();

nextLine() → it accepts the Space. Ex:- input: Arun Kumar
output: Arun Kumar.

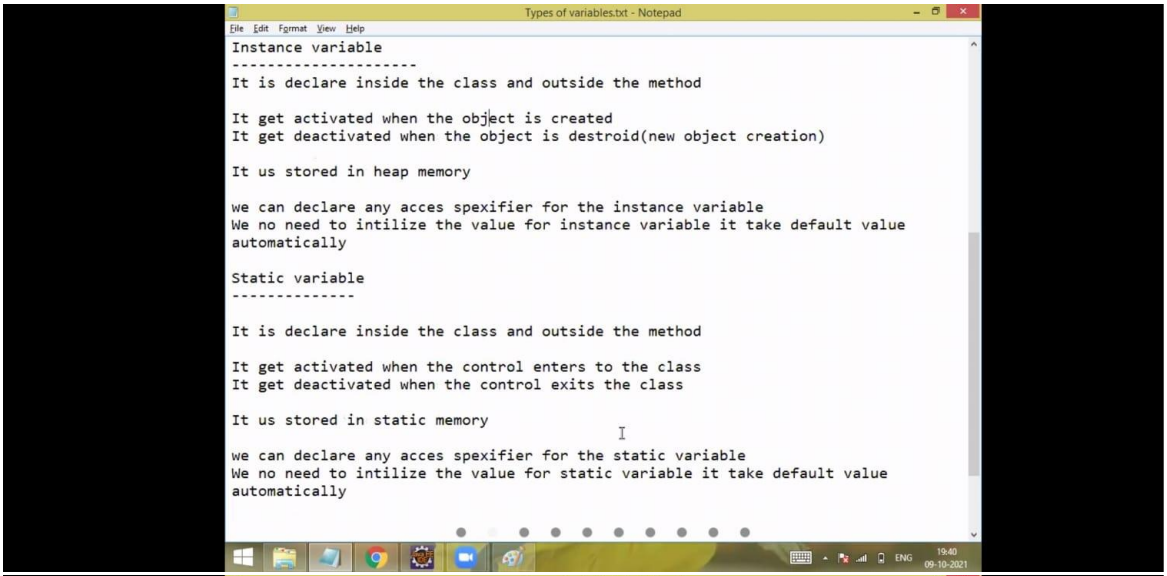
next(); → it ignore the Space Ex:- input: Arun Kumar
Output: Arun.

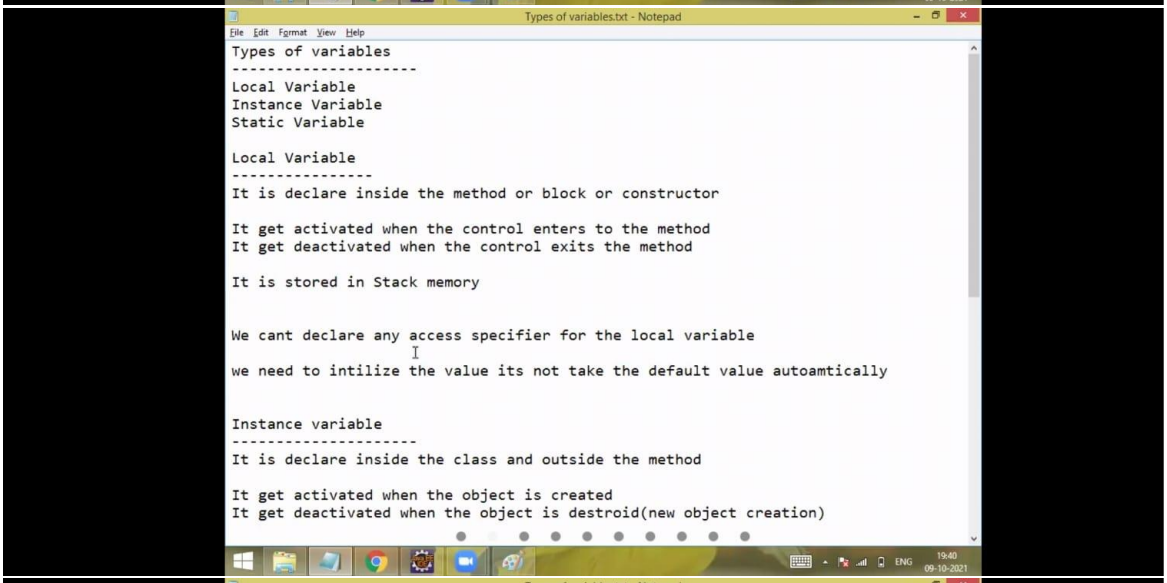
Pgm:-

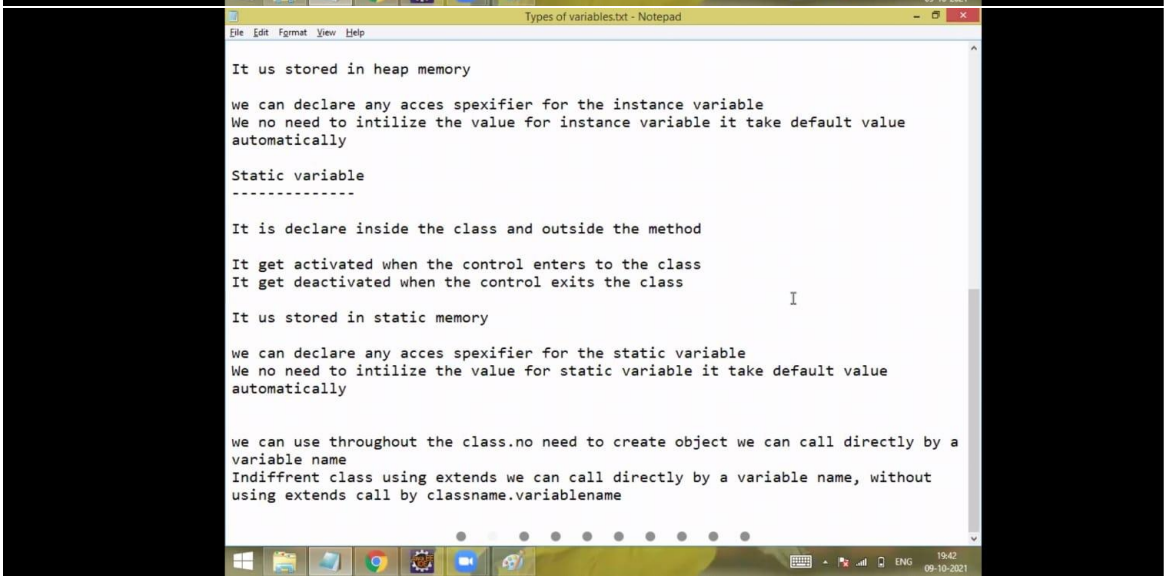
```
package
class
{
main method
{
    Scanner S= new Scanner(System.in);
    byte age = S.nextByte();
    System.out.println("Age is: "+age);
}}
```

output:- 10
Age is: 10

Note:- No need give 'l' for long & 'f' for float & 'd' for double.


Types of variables.txt - Notepad
File Edit Format View Help
Instance variable
---
It is declare inside the class and outside the method
It get activated when the object is created
It get deactivated when the object is destroid(new object creation)
It us stored in heap memory
we can declare any acces spexifier for the instance variable
We no need to intilize the value for instance variable it take default value automatically
Static variable
---
It is declare inside the class and outside the method
It get activated when the control enters to the class
It get deactivated when the control exits the class
It us stored in static memory
we can declare any acces spexifier for the static variable
We no need to intilize the value for static variable it take default value automatically


Types of variables.txt - Notepad
File Edit Format View Help
Types of variables
---
Local Variable
Instance Variable
Static Variable
Local Variable
---
It is declare inside the method or block or constructor
It get activated when the control enters to the method
It get deactivated when the control exits the method
It is stored in Stack memory
We cant declare any access specifier for the local variable
we need to intilize the value its not take the default value autoamtically
Instance variable
---
It is declare inside the class and outside the method
It get activated when the object is created
It get deactivated when the object is destroid(new object creation)


Types of variables.txt - Notepad
File Edit Format View Help
It us stored in heap memory
we can declare any acces spexifier for the instance variable
We no need to intilize the value for instance variable it take default value automatically
Static variable
---
It is declare inside the class and outside the method
It get activated when the control enters to the class
It get deactivated when the control exits the class
It us stored in static memory
we can declare any acces spexifier for the static variable
We no need to intilize the value for static variable it take default value automatically
we can use throughout the class.no need to create object we can call directly by a variable name
Indiffrent class using extends we can call directly by a variable name, without using extends call by classname.variablename

## Variables

See the above image for your references

# S E L E N I U M  T O P I C S

## ALERTS: ------Interface

==========

There are 3 types in java script pop-up or Alerts.

They are,

1. SimpleAlert (Only Ok button)

2. ConfirmationAlert (Both OK and Cancel button)

3. PromptAlert (Ok and Cancel button along with a text box)

Characteristics of javascript pop-up or Alerts

==========

We can't move the pop up.

We can not inspect the pop up.

It is black and white in color

If it contains only ok button then it is simple alert pop-up.

If it contains only ok and cancel button then it is confirmation pop-up.

If it contains only Text box, ok and cancel button then it is prompt pop-up.

we can handle any javascript pop up by using the statement

**Alert a = driver.switchTo().alert();**

In Alert interface we have different methods.

They are

1. accept()------- click on Ok

2. dismiss()----------click on cancel

3. getText()---------- get the text

4. sendKeys()----------enter the text

If the javascript pop-up is not present and still if we try to switch into it, then it will throw NoAlertPresentException

"http://demo.automationtesting.in/Alerts.html"

"http://www.greenstechnologys.com"

"http://toolsqa.com/handling-alerts-using-selenium-webdriver"

"http://ironspider.ca/forms/dropdowns.htm"

"http://demo.guru99.com/test/drag_drop.html

## SELECT----------Class

==========

DROP DOWN:

==========

1.Single value or Single Select

2.Multiple value or Multiple Select

Select ref = new Select (WebElemenetreference);

If the list box is developed by using select tag then we can handle it by using Select class.

Select class should be imported from the package org.openqa.selenium.support.ui

Select class contains one constructor which takes an argument of type WebElement wherein we have to pass address of the list box.

Select class contains some methods.

They are,

1 selectByIndex(int)---------- Select the options

2 selectByValue(String) ----------Select the options

3 selectByVisibleText(String) ----------Select the options

4 deselectByIndex(int) ---------- Deselect the options

5 deselectByValue(String)

6 deselectByVisibleText(String)

7 deselectAll()

8 getAllSelectedOptions() ----------To get all the selected options

9 getFirstSelectedOption()---------- To get first selected options

10 getOptions()---------- To get all the options

11 isMultiple()---------- To check whether list box is single or multi select

Note:

=====

Value and visible text are case sensitive.

If we pass any invalid arguments then it will throw NoSuchElementException

If the specified option is duplicate, then it will select 1st matching option.

We can handle duplicates using index.

In single select list box, we can not deselect the option.

If we try to deselect the option, it will throw UnsupportedOperationException.

Write Syntax

---

To print odd options

To print Even options

To Select all odd options

To Deselect odd options

To print last five options

To print first option

to print middle options

To print alternate options

Difference between FindElement method and FindElements Methods

| FindElement | FindElements |
|---|---|
| To identify the single WebElement. | To identify multiple webElements |
| Return Type WebElement | Return type List<WebElement> |
| WebElement txtUser= driver.findElement(By.id("username")); | List<WebElement> txtUsers=driver.findElements(By.id("username")); |

| | |
|---|---|
| If the locator is not found, then We get No Such Element Exception | If the locator is not found then we get empty array |

# Web Tables

==========

The tables present in webpage are called as Webtables

Two types

(i) Static WebTable(the position of the data do not change with time)

(ii) Dynamic WebTable(position of data changes with time)

To print all data present in Webtable to console

---

Step 1: Locate the table

Step 2: Locate the heading

Step 3: Locate the row inside the heading

Step 4: Locate the multiple heading elements

Step 5: Iterate to get individual heading element

Step 6: Use getText to get the text

Step 7: Locate the content of the table by using tagname called as tbody

Step 8: Locate multiple rows present in the table body

Step 9: Iterate each row using for loop

Step 10: Locate the multiple data using td tagname

Step 11: Iterate each data using for loop

Step 12: Get the data using getText method

# Window Handling

==========

It is used to move control from one window to another window

To Handle 2 windows:

==========

String parentWindowId = driver.getWindowHandle();

System.out.println("Parent Window ID:" + parentWindowId);

To get id of all the Windows

==========

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

To handle Multiple Windows

==========

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

List<String> l=new ArrayList<String>(allWindowId);

## JavaScript Executor:

---

-->If locator is found in DOM Structure but still it shows noSuchElement Exception due to hidden elements in the webpage,

if sendkeys not working, if click not working and for Scrollup and Scrolldown, we go for JavaScript Executor. It is an Interface.

There are few methods as

-->executeScript()

-->("arguements[0].setAttribute('value')",WebelementRef) --> sendkeys

-->("arguements[0].getAttribute()") --> get the particular value

-->("arguements[0].click()",WebelementRef) --> click

-->("arguements[0].scrollIntoView(true)",WebelementRef) --> scroll down

-->("arguements[0].scrollIntoView(false)",WebelementRef) --> scroll up

## TakesScreenShot

==========

It is a predefined interface

We can take the Screen shot by using this interface

Step 1: Create object for TakesSCreenshot

TakesScreenShot ts = (TakesScreenShot) driver;

Step 2: There is a predefined method called as getScreenShotAs();

getScreenShotAs()----This method will determine the screenshot to be stored in the form of a file or Byte or base64

In this method we pass the argument which determines output type

Step 3: Always the screenshot will be stored in a default location

If we want change the default to our desired location

Then

a, We have to add a jar file called as commons-io jar version is 2.7

b, In this jar file, we have a predefined class called as FileUtil

c, In this class we have a method called as CopyFile

d, This method enables us to copy the screenshot from default location to

our desired location

## Xpath:

---

-->Xpath is a XML path which is used to find locators on the webpage using DOM Structure.

-->There are two types of Xpath: Absolute which is denoted by single slash and Relative xpath which is denoted by double slash.

-->In that in our project we are using Relative xpath because we can directly find the element

anywhere at the webpage but in Absolute xpath we have to find from the head of the DOM structure.

## Actions:

---

-->Actions is a predefined class which is used for mouse over actions in a webpage. There are few methods as

-->moveToElement() --> used to do mouse over actions

-->doubleClick() --> used for double click

-->contextClick() --> used for right click

-->dragAndDrop() --> to move a webelement from one place to another

-->keyUp() --> for key release

-->keyDown() --> for key press

## Robot:

---

-->Robot is a predefined class present in java.awt package which is used for performing keyboard actions in a webpage. There are few methods as

-->keyPress() --> used to press a key

-->keyRelease() --> used to release a key

## Frames:

---

-->It is webpage embedded inside a webpage.

-->If any locators are placed inside a particular frame then we need to switch in to that particular frame and need to access the locators.

-->It is mainly for security purpose. We can switch to particular frame using ID, Name, Index and WebElement Reference.

--> Methods Used: driver.switchToFrame();

-->parentFrame(); --> switch to previous frame

-->defaultContent(); --> switch to parent window

## WAITS:

Synchronization:

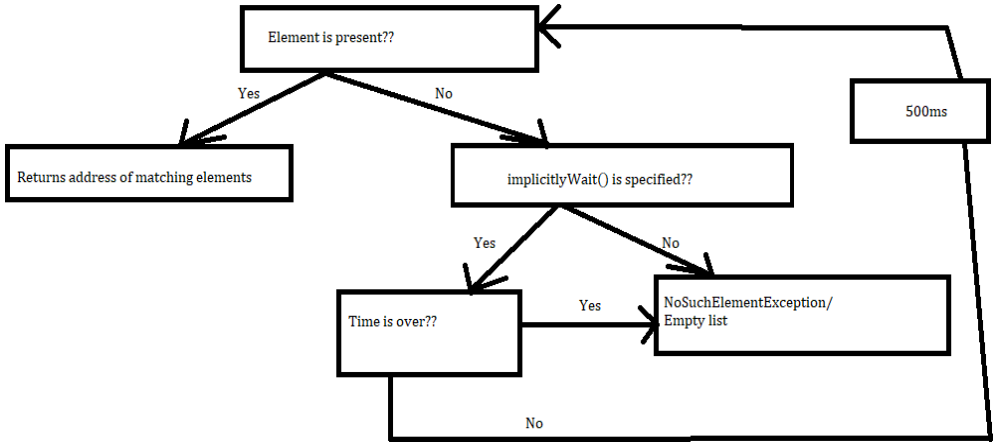
- Matching speed of selenium with the speed of application is called as synchronization.
- If we trying to find the elements due to application late response, it throws exception.
- Using waits we can resolve this exception
- we can handle synchronization by using,
    - implicit wait
    - explicit wait
    - Thread.sleep()
    - Fluent wait

## Implicit wait

- It will handle the synchronization of findElement() and findElements().

- implicitlyWait() takes 2 arguments of type long and TimeUnit.
- In the 1st argument we have to specify waiting time and the 2nd argument we have tospecify time unit
- The different time units are
  - ✓ DAYS
  - ✓ HOURS
  - ✓ MINUTES
  - ✓ SECONDS
  - ✓ MILISECONDS
  - ✓ MICROSECONDS
  - ✓ NANOSECONDS

## Work flow of Implicit wait:




- When the control comes to findElement() or findElements(), it will check whether theelement is present or not
- If the element is present, it will return address of the specified element.
- If the element is not present, it will check whether implicitlyWait() is specified or not.
- If implicitlyWait() is not specified then it will throw NoSuchElementException or Emptylist.
- If implicitlyWait() is specified then it will check whether the specified time is over or not.

- If the specified time is over then it will throw NoSuchElementException or Empty list.

- If the specified time is not over then for every 500ms it will check whether the element ispresent or not.

Note:

- 500ms is called as polling period which is present in a class called FluentWait.
- Using implicit wait we can handle synchronization of findElement() and findElements()only.

Syntax:

```
driver.manage().timeouts().implicitlyWait(time, TimeUnit.SECONDS);
```

Ex:

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

**Example:**

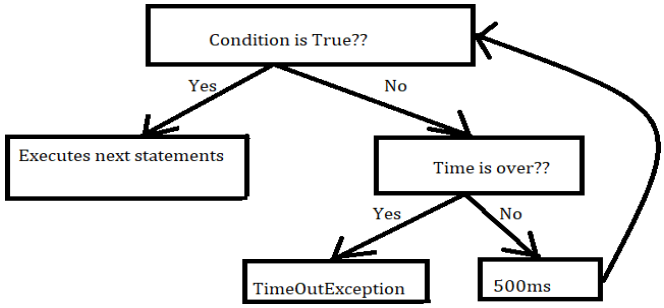
```
public class Login {

    public static void main(String[] args) throws

            InterruptedException {

            System.setProperty("webdriver.chrome.drive

            r",

                    "C:\\Users\\10657527\\Downloads\\chromedriver_win32
(1)\\chromedriver.exe");
            WebDriver driver = new ChromeDriver();
            driver.manage().timeouts().implicitlyWait(10,
            TimeUnit.SECONDS);
            driver.manage().window().maximize();
            driver.get("https://adactin.com/HotelApp/index.php");

            driver.findElement(By.id("username")).sendKeys("Venkat");
            driver.findElement(By.id("password")).sendKeys("Venkat@123");
            driver.findElement(By.id("login")).click();



    }
```

```
}
```

## Explicit wait:

- It is used to handle synchronization of any methods including findElement() andfindElements().
- WebDriverWait class is called as explicit wait which takes 2 arguments of type WebDriverand long
- Here the default time unit is seconds.

### Work flow of explicit wait:




- During the Run Time, when the control comes to wait statement, it will check whether thespecified condition is true of not.
- If the condition is true it will execute next statements otherwise it will check whether thetime is over or not.
- If the specified time is over then it will throw TimeoutException.
- If time is not over then for every 500ms it will check whether the condition is true or not.

### Note:

- 500ms is called as polling period which is present in a class called FluentWait.
- All the methods present in ExpectedConditions class are called as Predicates.

### Syntax:

```
WebDriverWait wait = new WebDriverWait(driver,time);
```

Ex:

```
wait = new WebDriverWait(driver,

time);

wait.until(ExpectedConditions.visibility

Of(element));
```

- The following are the Expected Conditions that can be used in Explicit Wait

1. alertIsPresent()
2. elementSelectionStateToBe()
3. elementToBeClickable()
4. elementToBeSelected()
5. frameToBeAvaliableAndSwitchToIt()
6. invisibilityOfTheElementLocated()
7. invisibilityOfElementWithText()
8. presenceOfAllElementsLocatedBy()
9. presenceOfElementLocated()
10. textToBePresentInElement()
11. textToBePresentInElementLocated()
12. textToBePresentInElementValue()
13. titleIs()
14. titleContains()
15. visibilityOf()
16. visibilityOfAllElements()
17. visibilityOfAllElementsLocatedBy()
18. visibilityOfElementLocated()

**Example:**

```
public class Login {
```

```
        public static void main(String[] args) throws

                InterruptedException {

                System.setProperty("webdriver.chrome.drive

                r",

                                "C:\\Users\\10657527\\Downloads\\chromedriver_win32
(1)\\chromedriver.exe");
                WebDriver driver = new ChromeDriver();
                driver.manage().window().maximize();
                driver.get("https://adactin.com/HotelApp/i
                ndex.php");

                driver.findElement(By.id("username")).sendKeys("Venkat");
                driver.findElement(By.id("password")).sendKeys("Venkat@123");

                WebDriverWait wait =new WebDriverWait(driver, 60);




        wait.until(ExpectedConditions.elementToBeClickable(driver.findElement(By.id("login
        "))));



                driver.findElement(By.id("login")).click();
                }
}
```

## Fluent wait:

- The fluent wait is used to tell the web driver to wait for a condition, as well asthe frequency with which we want to check the condition before throwing an "ElementNotVisibleException" exception.
- Frequency: Setting up a repeat cycle with the time frame to verify/check the condition atthe regular interval of time
- Let's consider a scenario where an element is loaded at different intervals of time. The element might load within 10 seconds, 20 seconds or even more then that if we declare an explicit wait of 20 seconds. It will wait till the specified time before throwing an exception.In such scenarios, the fluent wait is the ideal wait to use as this will try to find the element at

different frequency until it finds it or the final timer runs out.

### Syntax:

```
Wait wait = new FluentWait(WebDriver reference).withTimeout(timeout, SECONDS).pollingEvery(timeout, SECONDS).ignoring(Exception.class);
```

Ex:

```
Wait wait = new FluentWait(driver).withTimeout(5, TimeUnit.MILLISECONDS).pollingEvery(60, TimeUnit.SECONDS).ignoring(Exception.class);
```

## FILE UPLOADING USING ROBOT CLASS AND SENDKEYS

***Using Robot class:***

*Ex:*

```
public class Login {

	public static void main(String[] args) throws

		InterruptedException, Throwable {

		System.setProperty("webdriver.chrome.driver",

				"C:\\Users\\10657527\\Downloads\\chromedriver_win32 (1)\\chromedriver.exe");
		WebDriver driver = new
		ChromeDriver();
		driver.manage().window().maximize
		();
		driver.get("http://demo.guru99.com/t
		est/upload/");

		String path="C:\\Users\\10657527\\Desktop\\Venkatraman.docx";

		driver.findElement(By.name("uploadfile_0
		")).click();Thread.sleep(2000);
		Robot robot
		= new
```

```
                Robot();
                robot.setAut
                oDelay(3000
                );
                StringSelection selection = new StringSelection(
                                path);
                Toolkit.getDefaultToolkit().getSystem
                Clipboard()

                                .setContents(selection, null);
                // press ctrl+vsss
                robot.keyPress(KeyEvent.VK_CONTROL
                );robot.keyPress(KeyEvent.VK_V);
                robot.setAutoDelay(3000);
                // release ctrl+v
                robot.keyRelease(KeyEvent.VK_CONTR
                OL);robot.keyRelease(KeyEvent.VK_V);
                // press
                enter
                robot.setAu
                toDelay(30
                00);
                robot.keyPress(KeyEvent.VK_ENTER
                );
                robot.keyRelease(KeyEvent.VK_ENT
                ER);



        }

}
```

*Using sendKeys:*

***Ex:***

```
public class Login {

        public static void main(String[] args) throws

                InterruptedException, Throwable {

                System.setProperty("webdriver.chrome.driver",
```

```
                                    "C:\\Users\\10657527\\Downloads\\chromedriver_win32
(1)\\chromedriver.exe");
                    WebDriver driver = new
                    ChromeDriver();
                    driver.manage().window().maximize
                    ();
                    driver.get("http://demo.guru99.com/t
                    est/upload/");

                    String

                    path="C:\\Users\\10657527\\Desktop\\Venkatraman.d

                    ocx";

                    driver.findElement(By.name("uploadfile_0")).sendKe

                    ys(path);;



          }

}
```

# Broken links:

- If any link is failed load it's destination page then that link is called as broken links.
- It is not possible to verify broken links by using selenium.
- We can verify the broken links by using Java
- In order to verify the broken links we use a class called URL which is available in java.net package

**Sample web page:**

```
<a href="http://www.qspiders.com">Qspiders</a>

<a href="www.qspider.com">Qspider</a>

<a href="http://www.qsp.com">Qsp</a>
```

**Example1: Typical java program to verify broken links**

```
public class JavaDemo

{
```

```
        public static void main(String[] args) throws IOException
        {
                URL url = new URL("http://www.qspiders.com");
                HttpURLConnection con = (HttpURLConnection) url.openConnection();

                int code = con.getResponseCode(); //if code is 200, then
                link is not brokenSystem.out.println(code);

                String msg = con.getResponseMessage(); //if msg is Ok, then
                link is not brokenSystem.out.println(msg);
        }
}
```

**Example 2:**

```
public class Login {

        public static void main(String[] args) throws
                InterruptedException, Throwable {

                System.setProperty("webdriver.chrome.driver",
                                "C:\\Users\\10657527\\Downloads\\chromedriver_win32
(1)\\chromedriver.exe");
                WebDriver driver = new
                ChromeDriver();
                driver.manage().window().
                maximize();
                driver.get("https://www.goo
                gle.com/");

List<WebElement> allLinks = driver.findElements(By.xpath("//a"));

int broken=0, notBroken=0;

for(WebElement link:allLinks)
{
String href = link.getAttribute("href"); String text = link.getText(); System.out.println("Link: "+text);
System.out.println("URL: "+href);
```

```
url.openConnection();

try
{
URL url = new URL(href);
HttpURLConnection con = (HttpURLConnection)
int code = con.getResponseCode();
if(code==200)
{System.out.println("Links is not broken.");
notBroken++;

}
else
{
System.out.println("Link is broken1:");
System.out.println(con.getResponseMessage());
broken++;
}
}
catch (Exception e)
{
System.out.println("Link is broken2:"); broken++;
}

System.out.println("==============================================");
}

System.out.println("Total number of links: "+allLinks.size());
System.out.println("Number of broken links: "+broken);
System.out.println("Number of non broken links: "+notBroken);
```

```
Thread.sleep(2000); driver.close();
}
}
```

# P A G E  O B J E C T  M O D E L

---

QUESTIONS(Theory)

---

**1. What is meant by POM?**

*Page Object Model, also known as POM, is a design pattern in Selenium that creates an object repository for storing all web elements.

*It is useful in reducing code duplication and improves test case maintenance.

**2. What are the types of POM?**

*With PageFactory

*Without PageFactory

**3. What is meant by Annotations?**

Annotations are the lines of codes that can control how the method below them will be executed.

*They  are used to provide supplement information about the program.

*Annotation Start with "@" Annotation do not change action of a compiled program .

*Annotation help to associate metadata to the program elements i.e. instance variable, constructors, methods, classes.

**list of annotations**

@FindBy

@FindBys

@FindAll

@cacheLookup

**4. What are the page factory Annotations?**

In Page Factory, Annotations are used to give descriptive names for WebElements to improve code readability. And annotation @FindBy is used to identify Web Elements in the page.

*PageFactory is a way of implementing the “Page Object Model”.

*Here, we follow the principle of separation of Page Object Repository and Test Methods.

*It is an inbuilt concept of Page Object Model which is very optimized.

**5.What is mean by pojo class?**

*POJO stands for Plain Old Java Object. It is an ordinary Java object, not bound by any special restriction other than those forced by the Java Language Specification and not requiring any classpath.

*POJOs are used for increasing the readability and re-usability of a program.

*POJOs have gained the most acceptance because they are easy to write and understand.

6.**How will you generate the pojo class?**

```
public class Employee {


   public String EmpName;
   public String EmpAddress;


   public Employee(String EmpName, String EmpAddress ) {
      this.EmpName = EmpName;
      this.EmpAddress = EmpAddress ;


   }
//generate the getter
   public String EmpName() {
      return this. EmpName+ " " + this.EmpName;
   }


public String EmpAddress() {
      return this.EmpAddress+ " " + this.EmpAddress;
   }
}
```

7.Write the folder Structure used in POM?

*maven-project/pom. ...

*maven-project/src/main – contains source code and resources that become part of the artifact.

*maven-project/src/test – holds all the test code and resources.

*maven-project/src/it – usually reserved for integration tests used by the Maven Failsafe Plugin.

**8.What is the difference between @FindBys and @FindAll?**

*FindBys is used to mark a field on a Page Object to indicate that lookup should use a series of @FindBy tags in a chain as described in ByChained.

*It can be used on a types as well, but will not be processed by default.

* We can pass more than one locator it behaves like AND operator.

*If any one locator mistakes it will not enter the value, instead it will throw no such Element Exception

Example

```
 @FindBy
      @FindBy(id="email")
          private WebElement txtUserName;


          @FindBy(id="pass")
          private WebElement txtPassword;


          @FindBy(name="login")
          private WebElement btnClicklogin;
```

@FindAll

* We can pass more than one locator it is behaviour like OR operator.

*If any one locator mistakes it will  enter the value.

```
@FindAll
      @FindBy(id="email")
```

```
private WebElement txtUserName;


@FindBy(id="pass")
private WebElement txtPassword;


@FindBy(name="login")
private WebElement btnClicklogin;
```

**9.What is the use of POM?**

*A Project Object Model or POM is the fundamental unit of work in Maven.

*It is an XML file that contains information about the project and configuration details used by Maven to build the project.

*It contains default values for most projects.

*The POM contains information about the project and various configuration detail used by Maven to build the project(s).

* POM also contains the goals and plugins.

**10.What are the use of @cacheLookup?**

PageFactory annotation @CacheLookup is used to mark the WebElements once located so that the same instance in the DOM can always be used.

CacheLookup attribute can be used to instruct the InitElements() method to cache the element once its located and so that it will not be searched over and over again.

# D A T A  D R I V E N  F R A M E W O R K

ref.getSheet()--------> to read the sheet--->String(Sheet name)-------->Sheet(Interface)

ref.createSheet()----->to create a sheet in workbook------>String(sheet name)-------->Sheet(Interface)

ref.write()---> to write the workbook------->FileoutStream ref

Sheet--Interface

=====

ref.getRow()-------->to read or get the Row from the sheet------>Integer(index)---------->Row(Interface)

ref.getPhysicalNumberOfRows()---------> to find the number of Rows filled with data---------->Integer

ref.createRow()---------->to create a row  in the sheet---------->int(row index)------>Row(Interface)

Row--Interface

==========

ref.getCell()------->to read or get the cell from the Row ------->Integer(index)------->Cell(Interface)

ref.getPhysicalNumberOfCells()---------->to find the number of Cells present in each row filled with data---------->Integer

ref.createCell()-------->to create a cell in the row---------->int(cell index)-------->Cell(Interface)

Cell-Interface

==========

ref.getStringCellValue()---------->to read the string values from the cell------>String

ref.getCelltype()---------> this method will tell whether a numerical data or String data is filled in the cell------->Integer

this method output is Integer---(0 or 1)

When getCellType == 1----String data is filled in the cell

When getCellType == 0 ----Numerical data is filled in the cell

ref.getNumericCellValue()-------> this method will read or get the numeric value present in the cell---------->double

Type Casting (Convert Double data into Long data)

Syntax : "long ref = (long) doubleref;"

ref.getDateCellValue()----> to get the date from the cell

ref.setCellValue()----->to set value for the cell----> String(cell value)

how to read the date

==========

DateUtil-------Class---------isCellDateFormatted()-Static Method------->It will check the value present in the cell is according to the date format or not

------> If it is date format

-----> Cell--Interface

ref.getDateCellValue()-------> Date(Return type)----->It will give month, date and year along with it it give time

To change the format of the date

==========

SimpleDateFormat-----> class

format ----------> method--------> It will change already existing date format new prescribed.

how to Write a Excel Sheet

==========

Step1: Create object for file and specify the location of the file

Step 2: Create object for the workbook

Step 3: Create a Sheet

Step 4: Create a Row

Step 5: Create a cell

Step 6: Set the value for the cell

Step 7: Create a object FileOutPutStream to write a file

how to update the data in Excel

==========

Step 1: To read all the data present in Excel

Step 2: Check whether desired data present in the excel sheet or not

If data exists replace with new data

If data doesnt exist read the other datas

Step 3: Write the updated data back in the excel sheet

To write bulk of data in Excel Sheet

==========

Step 1: Get multiple data from user as input

Step 2: Add all the data in the list

Step 3: Iterate and get the data present in the User defined List

Step 4: Enclose the method setCellValue() inside the loop and pass (listref.get(i))

Step 5: Create object for FileOutputStream

Step 6: Write or update the file.

## Data Driven Interview Q&A

1. **What is DataDriven?**

Data Driven is a framework that is used to drive test cases and suites from an external data feed.

The data feed can be data sheets like xls, xlsx, and csv files. A Data Driven Framework in Selenium is a technique of separating the "data set" from the actual "test case" (code).

2. **What are the format available in Excel to read?**

Existing Excel workbook or worksheet with extension .xls
Existing Excel workbook or worksheet with extension
.xlsxComma-separated values text file with extension .csv
Text file, often with extension .txt or extension .asc

3. **What are Row and Column size of xls and xlsx?**

| Type | Rows | Columns |
|---|---|---|
| **Xls** | 65536 | 256 |
| **Xlsx** | 1048576 | 16384 |

4. **What are the jar files available to read/write in Excelsheet?**

commons-collections4-4.1.jarpoi-
3.8.jar

poi-ooxml-3.8.jar

poi-ooxml-schemas-3.8.jar
xmlbeans-2.6.0.jar

5. **What is the use of fileinputstream?**

FileInputStream is used to read data from a file in the form of sequence of bytes. It is meant for reading streams of raw bytes such as image data.

6. **What is meant by workbook?**

A workbook is a collection of one or more spreadsheets, also called worksheets, in a single file, in Microsoft excel. The workbook has number of sheets. A sheet is a central structure of a workbook, which represents a grid of cells. The Sheet interface
extends **java.lang.Iterable**.Sheets has rows and cell, which represents the column in the spreadsheet.

7. **Whether workbook is interface or class?**

Workbook is an interface implemented by HSSFWorkbook and XSSF Workbook.

8. **What is the method available to get particular sheet?**

We use the method called **getSheet()** to get the particular sheet.

Public class Exel {
Public static void main(String []args) { File file = new File (“ file path”);
FileInputStream stream = new FileInputStream(file); Workbook w = new Workbook (stream);
Sheet s = w.getSheet(“data”);

**9.** What is the purpose of getPhysicalnumberofcells?

This method is used to get the cell count or the number of defined cells and not the total number of cells in the particular sheet.

**10.** What method is used to get particular row?
getRow() is the method used to get the current row in the sheet. getRow(index) will give the particular row.

**11.** What method is used to get particular cell?
getCell() is the method used to get the current cell/column in the sheet. getCell(index) method will give the particular cell/column.

**12.** What is the purpose of getPhysicalNumberofRows?
getPhysicalNumberofRows() method is used to get the total number of defined rows or rowscount in the sheet.

**13.** What is the purpose of getcelltype?
getCellType() method is used to return the type of cell, for e.g int, string etc.

**14.** What are the possible output given by the getcelltype?
1= text; 0= number;

**15.** What is the purpose of getStringCellValue and its return type?
The purpose of the getStringCellValue() method is to get the value of the cell as a string. For numeric cells and the cells that has formulas that are not string, this will throw an exception and for the empty cells it will return an empty string.

The return type is string.

**16.** What is the purpose of getNumericCellValue and its return type?
The purpose of the getNumericCellValue() method is to get the value of the string as a number. The return type is double.

**17.** What method is used to replace the cell value?
setCellValue() method is used to replace the cell value.

**18.** What is the purpose of FileOutputstream?
FileOutputStream is an output stream used for writing data/streams of raw bytes to file or store data to the file. Usually this will be used to write primitive data into the file.

**19.** How will you write data in workbook?

Here are the basic steps for writing an Excel file:

1. Create a Workbook.
2. Create a Sheet.
3. Repeat the following steps until all data is processed:

a. Create a Row.
b. Create Cellsin a Row.

4. Write to an OutputStream.
5. Close the output stream.

Public class Sample {
Public static void main (String[]args){ File file = new File( path);
FileInputStream stream = new FileInputStream(file); Workbook w= new XSSFWorkbook (stream);
Sheet s = w.getSheet(data); Row r = s.getRow(1);
Cell c= r.getCell(1);
String name = c.getStringCellValue(); c.setCellValue(string);
FileOutputStream out = new FileOutputStream(file); w.write(out);
}

**20.** How will you create new Sheet, row and cell?
The new sheet will be created by createSheet() method. Sheet s = w.createSheet(sheet);
The new Row will be created by createRow() method Row r = s.createRow(0);
The new Cell will be created by createCell method Cell c = r.createCell(0);

**21.** What is the difference between Xls and Xlsx?

| XLS | XLSX |
|---|---|
| Xls is the older version of excel and is the default file format for 2003 versions | Xlsx is the latest version of excel and is the default file format for versions from 2007 onwards |
| It is based on Binary Interchange File Format(BIFF) and the information is directly stored into a binary format | It is based on on openXML format and the information is stored in a text file that uses XML to define all its parameters |
| It is faster especially for the files that require complex formula for a larger set of data | It is slower on files that requires complex formula for larger set of data |
| It is readable by all versions of excel | It is readable only on versions from 2007 onwards |
| It is capable to support macros | It doesn't support macros |
| It has 2^16 rows and 2^8 columns | It has 2^20 rows and 2^14 columns |

**22.** What is the difference between HSSF and XSSF workbook?

| HSSF | XSSF |
|---|---|
| Horrible SpreadSheet Format(HSSF) is used to read and write the Xls format of Ms Excel (1997-2007 versions) | XML Spreadsheet Format(XSSF) is sued to read and write the Xlsx format of Ms Excel ( 2007 and onwards versions) |
| It is a high-level class under the org.apache.poi.hssf.usermodel package. Implements Workbook interface | It is a class under org.apache.poi.xssf.usermodel package and implements Workbook interface. |
| This can export upto 65535 lines | This can export upto 1.04 million rows |

**23.** What is the difference between jxl and apache POI jar file?

| JXL | APACHE POI |
|---|---|
| Java JXL doesn't support XLSX format | Apache POI supports both XLSX and XLS formats |
| It doesn't support conditional formatting | It supports conditional formatting |
| It doesn't support richtext formatting | It supports RichText formatting |
| It only supports certain text rotations: horizontal/vertical, +/- 45 degrees and stacked | It supports any integer number of degrees plus stacked |
| It doesn't support drawing shapes | It supports drawing shapes |
| It doesn't support split panes | It supports split panes |
| | |

# J U N I T  F R A M E W O R K

## Unit Testing Framework

Unit Testing it is a testing done by developers for testing a single module without using the proper tetsing scripts

Junit Framework majorly helps in Setting up the verification or Validation point

how to integrate Junit Framework in Maven Project

---

Step 1: Create a Simple Maven Project

Step 2: Add dependancy

WebDriverManager --- 5.0.3

Selenium Java ------3.141.59

apache poi ooxml -----3.8 beta4

Jnuit dependancy -----4.2.0 or 4.12

Step 3: Create a class under src/test/java folder

## Annotations Used in Junit

---

Annotations are used above the methods as well as variables

@BeforeClass ----This annotation is used above the method, This method will execute only once before the execution of all the methods present in the class

Browser Launch or Browser Setup

@Before ----This annotation is used above the method, This method will execute each time before the execution of @Test annotation

Setting up verification point or Finding the time

@Test----This annotation is used above the method, This method will execute only once, This is the mandatory annoatation for execution

Main Functionality

@Afetr---This annotation is used above the method, This method will execute each time after the execution of @Test annotation

Setting up verification point or Finding the time

@AfterClass----This annotation is used above the method, This method will execute only once after the execution of all the methods present in the class

Closing Browser, Screenshot

Note:

No Private methods are allowed in Junit

Methods with @BeforeClass and @AfterClass annotation should be public static

If there are more than one @Test annotation then the methods will execute in the ascending order of the method name

## Assertion

==========

Verification or Validation

how to Achieve Verification

## (i) hard Assert(Junit)

(i)Assert.assertTrue(boolean);

Assert--Predefined class

assertTrue--Static method

if boolean = true----The following lines will execute

if boolean = false ---The following lines will not execute

Assert.assertTrue(true);

sysout("test");

(ii) Assert.assertFalse(boolean);

boolean = true-- The following lines will not execute

boolean = false -- The following lines will execute

(iii) Assert.assertEquals(expected, actual);

If expected = actual - The following lines will execute

If !expected = actual -- The following lined will not execute

If the following lines dont execute then the method will be failed

## (ii) SoftAssert(Testng)

@Ignore ---- This annotation is used to ignore the Test methods

## Suite Level Execution

======================

Suite -- Collection of test methods

@RunWith(Suite.class)

@Suite.suiteclasses({Testing.class,Testing2.class})

# T E S T N G  F R A M E W O R K

## TestNG

======
TestNext Generation Framework
TestNew Generation Framework
Disadvantages of JUnit
====================
(i) By using Junit framework only Unit Testing is possible
(ii) By using Junit framework, we cannot generate execution Reports, Reports will contain date and time of execution, no of test methods passed failed, ignored, skipped, test data
(iii) By using Junit only hard Assertion is possible
(iv) By using Junit Private Annotations are not possible (Achieving Encapsulation will be difficult).

## Advantages of TestNg over Junit

==================================
(i) By using Testng we can generate default html reports.
(ii) We can prioritize the test methods by using priority concept.
(iii) We can execute a particular test method more than once by using "invocation Count"
(iv) We can pass the parameters in Testng by using xml file
(v) The execution of Testng will start from "Testng.xml" file
(vi) We can pass bulk of data by using a class called as data Providers
(vii) We can Achieve Suite level execution in Testng (Collection of test methods is called as Suite)
(viii) We can achieve Cross Browser Testing in Testng to check the compatibility and stability of the application
(ix) We can Achieve Parallel execution in testng, to reduce the time of execution.
(x) We can Achieve both hard assert and Soft Assert in Testng
(xi) We can Achieve grouping of test methods in Testng
(x) We can Achieve Rerunning of failed test methods in Testng

Note:
======
(i) Private methods are also possible
(ii) @BeforeClass and @AfetrClass need not to be static

## Annotations used in Testng

============================
@BeforeSuite
@BeforeTest
@BeforeClass
@BeforeGroups
@BeforeMethod
@Test
@AfterMethod
@AfterGroups
@AfterClass
@AfterTest
@AfterSuite
Suite - Collection of Tests
Test - Collection of Classes
Class - Collection of groups
Groups - Collection of Test methods

possible Usage of Testng
=======================

Testng+ POM+Baseclass+DataDriven Framework
Testng+Cucumber

how to Setup a Testng Project
==========
Step1: Create a Simple Maven Project
Step 2: Add Dependencies
Testng - 6.14.3
Supporting jar (j commander - 1.7)
Step 3: Add Testng plugin
Help -- Eclipse MarketPlace ----Testng-----Install

# Priority

==========
We can change the order of execution by giving values to option called **priority**, The value ranges from negative to positive.
The execution flow is in the ascending order of the priority
The default value of priority is "0"

# Invocation Count

==========
If we want to run a particular test method multiple times then we have to go for invocation count
Default value of invocation count =1.

# Enable

==========
To skip particular test method we use
**Enabled = false**
The default value of Enabled is true.

# Suite Level Execution

==========
Collection of test methods is called as Suite Level execution
To convert Class level execution into Suite level execution
Select the Class----RC---TestNg---ConvertToTestNg----Generate testng.xml file
A new file with default name testng.xml will be generated

Explanation of Testng.xml file
==========
It will contain tags and fields
<tags>
<suite name = "xyz" >---</suite>---Collections of tests
<test name ="xyz"> ---- </test>--------Collection of classes
<classes>----</classes>
<Class name ="xyz">-----</class> -------Collection of test methods
<method>

Red colour - Error
Blue- Failed
Green - Passed

# Assertion in Testng

==========
(i) Both hard Assert as well as Soft Assert both are in Testng

| Hard Assert | Soft Assert |
|---|---|
| It is used to set the verification or validation point | It is used to set the verification or validation point |
| If verification is failed, then entire test method will be failed | If verification is failed then test method will get passed and the line of codes below the verification point will get executed. |
| **Methods in Assert** | |
| ref.AssertTrue(boolean) | ref.assertTrue (boolean) |
| ref.AssertFalse(boolean) | ref.assertAll(); |
| ref.AssertEquals(Expected, Actual, message) | ref.AssertEquals(Expected, Actual, message) |
| The methods present in the hard assert are static methods | Non static methods |
| We cannot convert a hard assert into soft assert | We can convert a soft assert into hard Assert. |
| During compile time, Assertion error can be handled | During compile time, AsserrtionError can be handled. |

**expectedExceptions acts as a catch block to handle the AssertionError**

# Parameters

==========
We can pass the data from testng.xml file
to do this we have to include a tagname called as <parameter name="username" value="ram">
Whenever we use parameter include @Parameters annotation above the test method, and this annotation should have the attribute value of the name.
If we want to change the data from xml file to Test method, then we use annotation called as @Optional, or the Attribute value from Testng.xml file and @parameters annotations doesnt match then it takes the data from the @Optional

# Data Providers

==========
DataProviders is used to pass bulk amount of data to a particular test method.
We can pass more number of positive and negative inputs.
DataProvider is a test method which will be present in both same class and different class
Step 1: Near @Test annotation ( dataproviders = "xyz")
Step 2: Create a seperate method for provide the data (This will contain combination of both positive and negative inputs)
By using data providers we can rerun particular test method with positive and negative input or test data combination

# Grouping in TestNg

==========
Smoke Testing - It is initial testing, done to check whether the build is testable or not, It is also called day 0 Check
Regression Testing - Whenever a new feature is added, the already existing feature should not lose its functionality
Sanity Testing - Before Release, we check the main functionality of the application.
There are 100 test methods
50 - Smoke Testing
25 - Regression Testing
25 - Sanity Testing
We can group a particular test method present in a class to execute together.
We have to include groups in @Test annotation
We have to include a tag called as <groups> in testng.xml file
We have to add a tag called as <include> to execute the test methods
We have to add a tag called as <exclude> not to execute the test methods

## Parallel Execution

==========
To reduce the Time of Execution.
There are three different levels of Parallel Execution
(i) Method Level Parallel Execution
If more than one Test method (@Test annotation) executes parallely
(ii) Class Level Parallel Execution
If more than one Class (collection of @Test methods and other annoations) executes parallely.
(iii) Test Level Parallel Execution
If more than one Test (Collection of Classes) Executes parallely
Whenever we go for parallel execution
Then add tag name parallel and mention level of Execution

## Cross Browser Testing

==========
To check the stability of the running the same code in different browser
To integrate Cross Browser Testing, Parallel Execution and parameter

## Rerun

==========
We can the rerun the failed test method only in TestNg
Two types
(i) Manual Rerun
(ii) Automatic Rerun
a) When we know which test method is failed
b) When we don't know which test method is failed

**Manual Rerun**
==========
In this case, the failed test method is executed multiple times manually.
Whenever a particular test method is failed, then automatically a testng-failed.xml will be generated in the test output folder
If we execute that particular xml file then failed test method alone will get executed.

# C U C U M B E R  F R A M E W O R K

BDD (Behaviour Driven Development) with the execution of TDD (TestDriven Development).
Used to convert the exact requirements into **plain English language.**
It support in sprint automation.
Non-Technical people can also understand the automation coverage.
Cucumber supports **Ruby, Java, Grovy, Python, .NET and PHP**.
We can use the **Gherkin language or Gherkin Keyword** (PlainEnglish).
Cucumber options.
Cucumber Reports.

**CUCUMBER ANNOTATIONS**

- @Given

- @When
- @Then
- @CucumberOptions

**BDD FRAMEWORK**

1. Cucumber
2. Jbehave
3. Nbehave
4. Specflow

**GHERKIN Keyword / GHERKIN Language**

- Feature
- Scenario
- Given
- When
- Then
- And
- But
- Scenario Outline
- Examples
- Background

**Feature**

1. Feature defines the logical test funtionality test [What you will test the feature file].
2. **Feature** keyword is present at the starting of the feature file.

**Scenario**

1. Business rule[Provides the scenario name /purpose of the scenario].

**Given**

1. Some precondition step.

**When**

1. Some key actions.

**Then**

1. To observe the outcomes or validation.

**Scenario Outline**

1. Used to execute the same scenario with multiple times wih different set of test datas.

**Examples**

1. Used to pass the parameters using the **| (pipe symbol) |** separate the each variables.

**But**

1. Used to add negative type comments.

**Background**

1. Define steps which is are common to all the tests in the feature file.

**And**

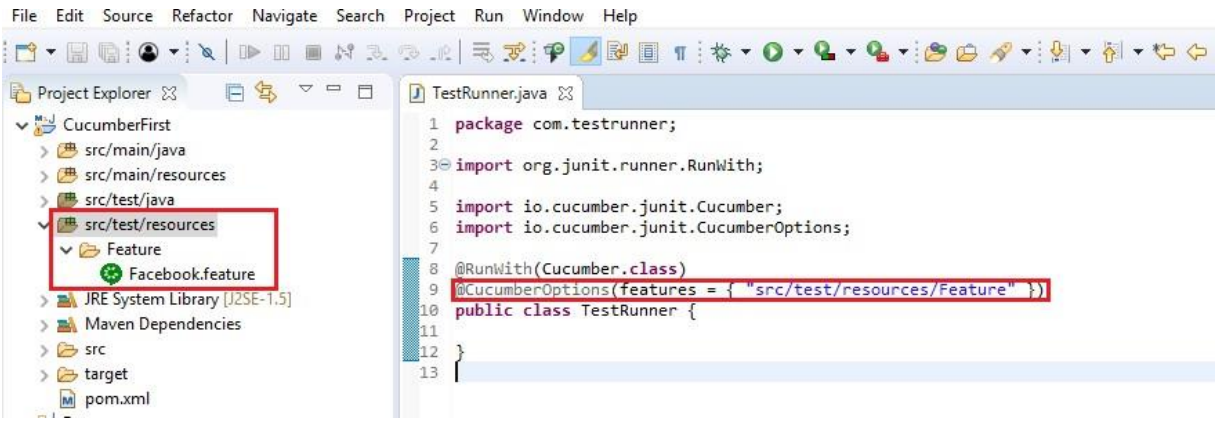
1. Used to add conditions in feature file.

**CUCUMBER OPTIONS**

- features
- glue
- dryRun
- monochrome
- plugin
- pretty
- tags
- strict

**features**

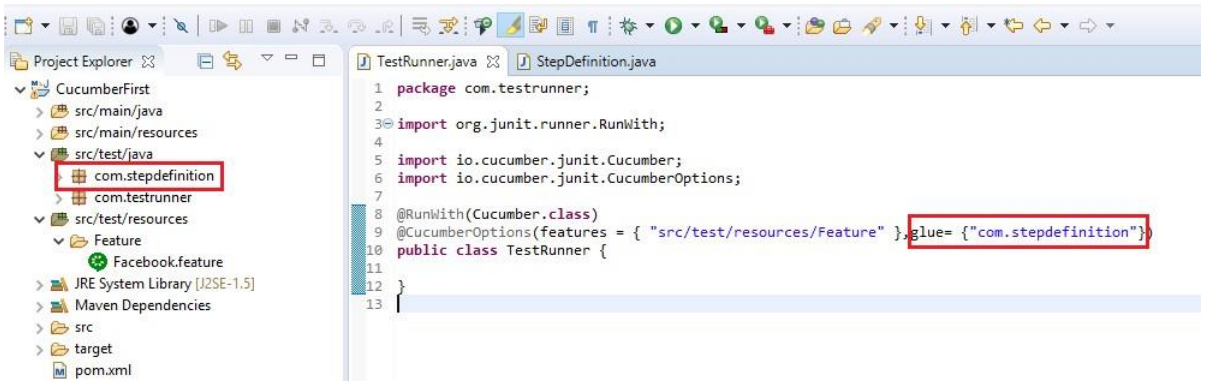
- **feature** option is used to locate the Feature folder in project folder structure.




**FEATURE**

## Glue

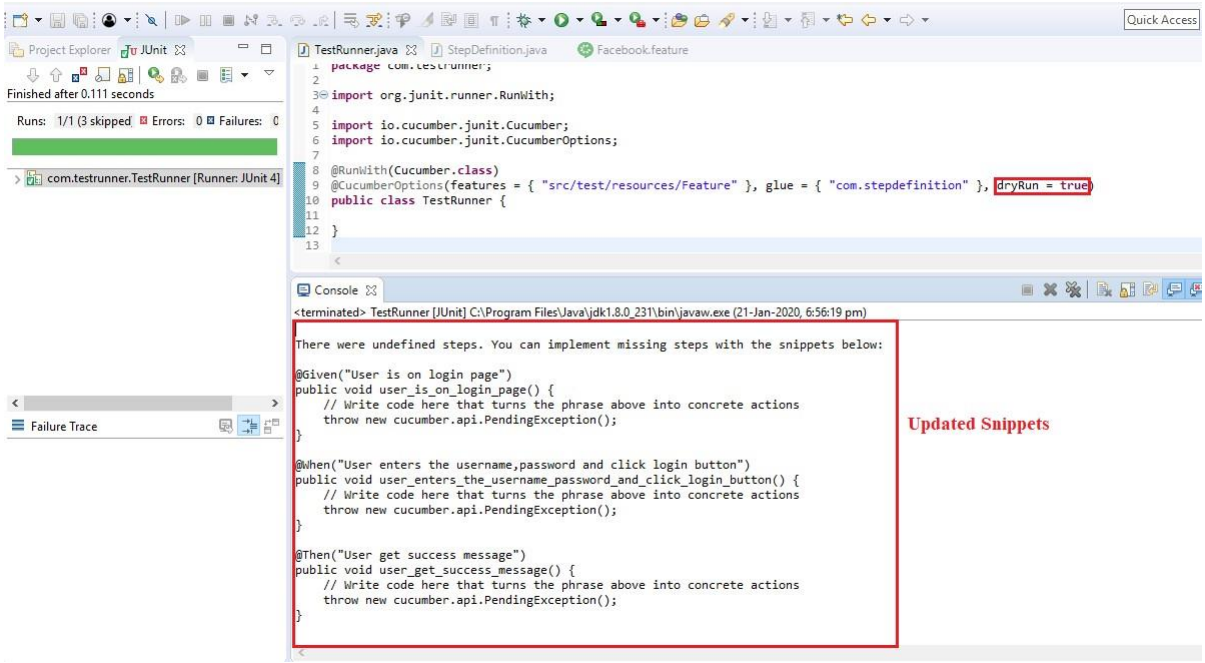
- **glue** option is used to locate the stepDefinition class.




**GLUE**

## dryRun

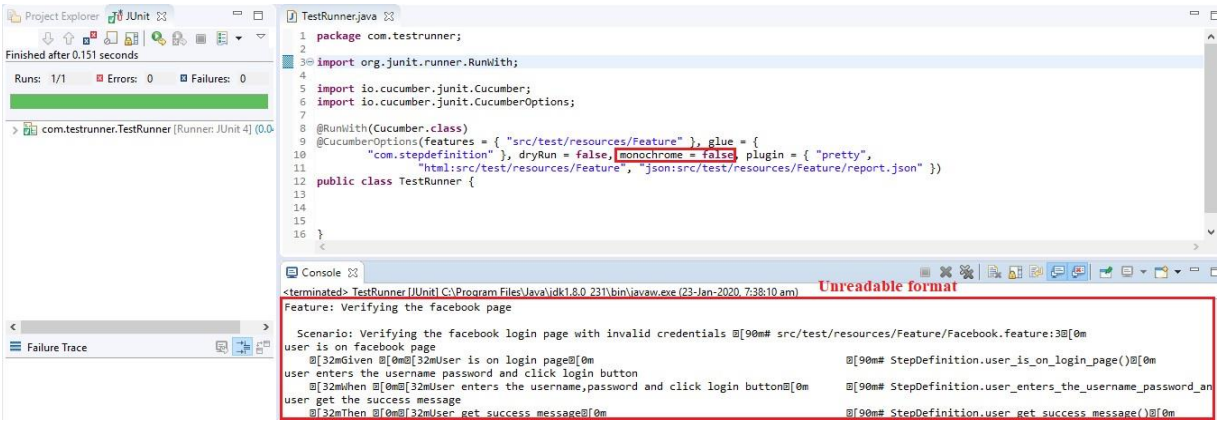
- **dryRun** option can either set as true or false.
- if it is set as **true** means check the every step mentioned in the feature file has corresponding code written in StepDefinition class or not.so in case any of the step is mismatch it update the snippets alone.



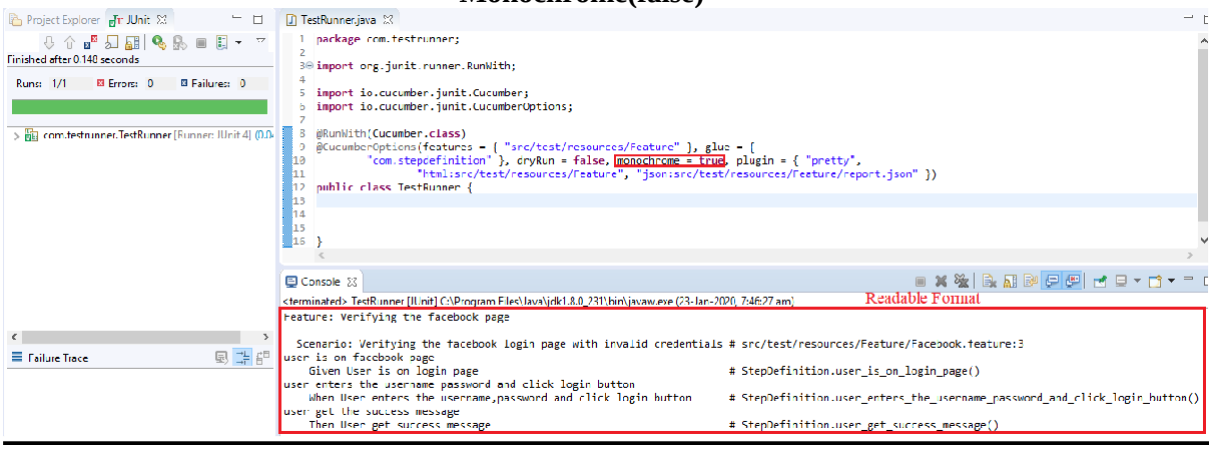

**dryRun**

## monochrome

- **monochrome** option can either set as **true** or **false.**
- If it is set as **true**, it means that the **console output** for the Cucumber test are much more readable.
- if it is set as **false**, then the **console output** is not as readable.




**Monochrome(false)**



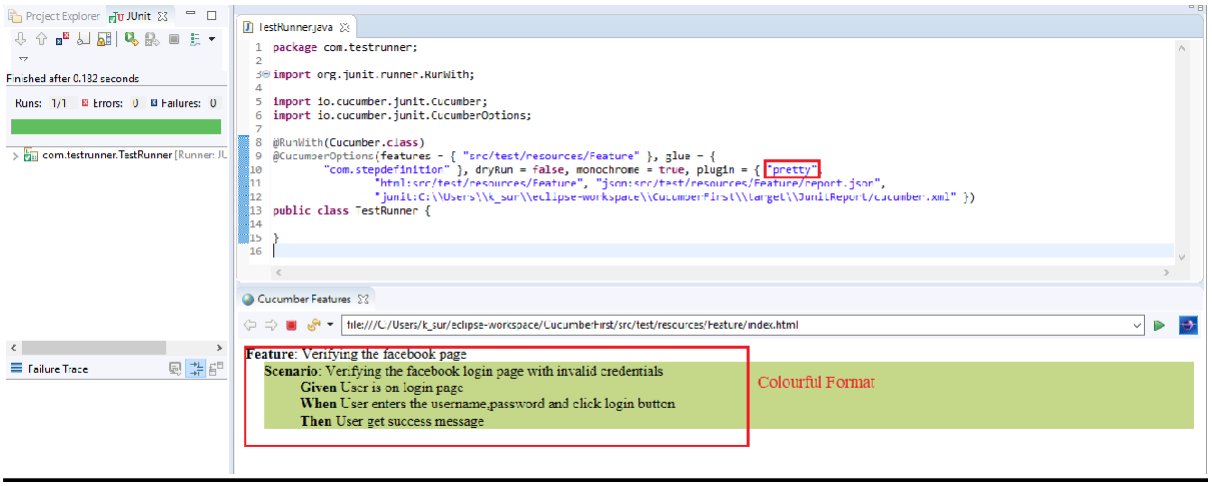

**Monochrome(true)**

**plugin**

- **plugin** option is used to specify different formatting options for the output reports(html,json,xml).

**pretty**

- **pretty** option is used to Print the **Gherkin** source with additional colors and stack traces for errors.




**Pretty**

**tags**

- ❖ **tags** option is for which tags in the feature file should be executed
- ❖ **tags = {"@<tag name>"}strict**
- ❖ **strict** option is **true** will fail execution, if there are undefined or pending steps

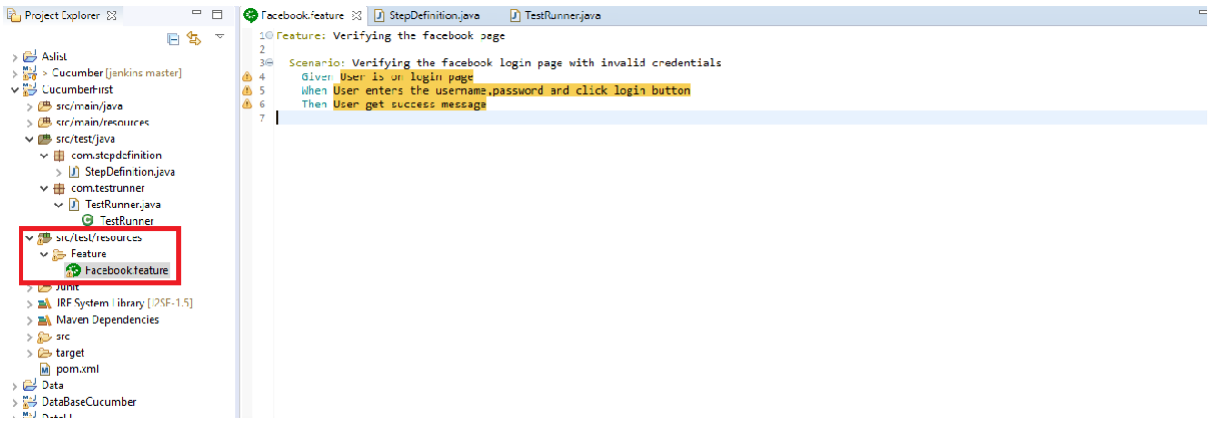
**Cucumber Options**

| Option | Purpose | Default Value |
|---|---|---|
| features | **set**: The paths of the feature files | {} |
| Glue | **set**: The paths of the step definition file | {} |
| Tags | **instruct**: What tags in the feature files should be executed | {} |
| dryRun | **true**: Checks if all the steps have the Step Definition | false |
| monochrome | **true**: Display the console output in much readable format | false |
| plugin | **set**: What all the report formats to use | {} |
| Strict | **true**: Will fail execution, if there are undefined or pending steps | false |

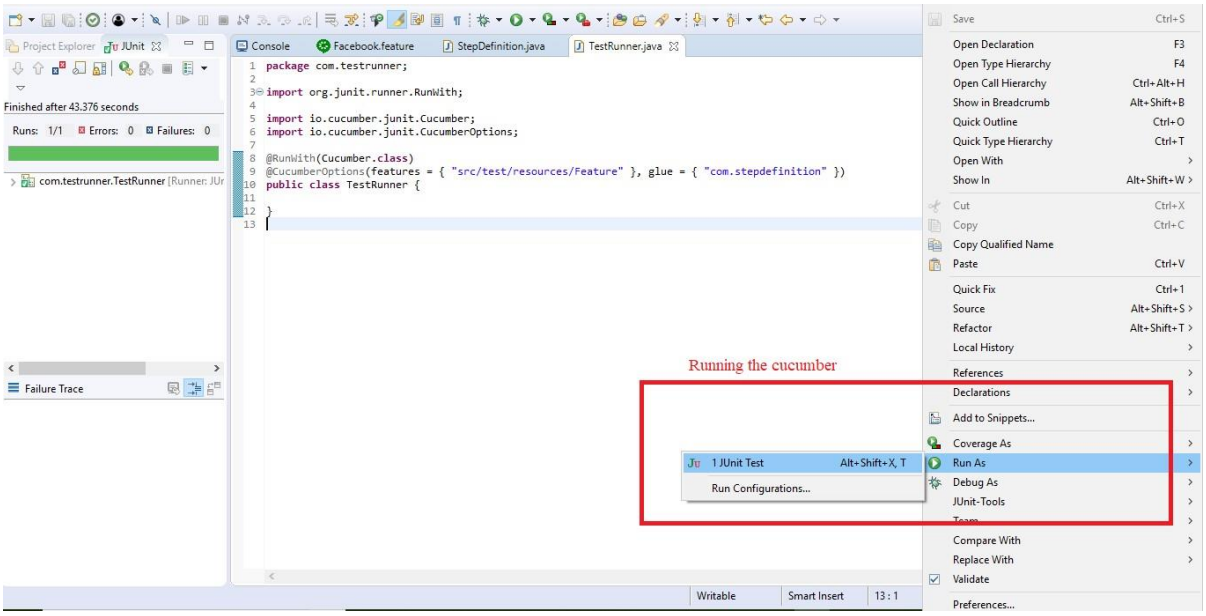
**Steps:**

1. Add the cucumber plugin [cucumber eclipse plugin] in Eclipse marketplace.
2. Create maven project
3. Add the cucumber dependency(cucumber-junit,cucumber-java,selenium-java).
4. Create feature file in (**src/test/resources**).
5. Create TestRunner class(**src/test/java**).
6. Generate the snippets.
7. Copy the step definition files.
8. Copy the generated snippets and paste in the Stepdefinition class.
9. Run the code using testrunner class (right click□run as□junit)

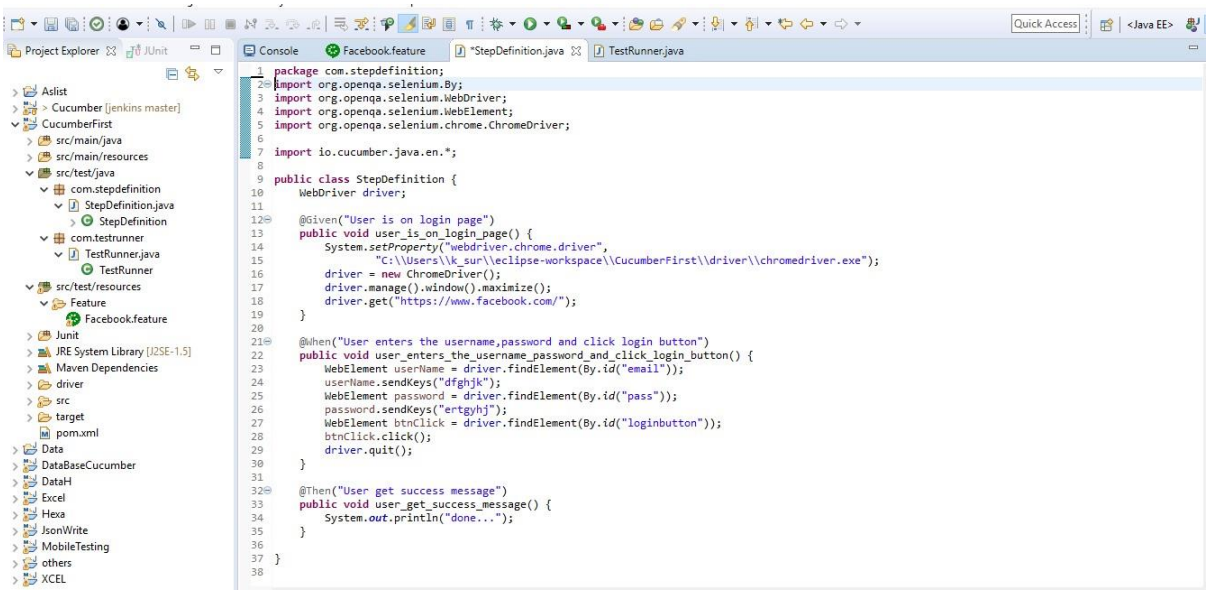
**1. CUCUMBER PROGRAM TO LAUNCH FACEBOOK**



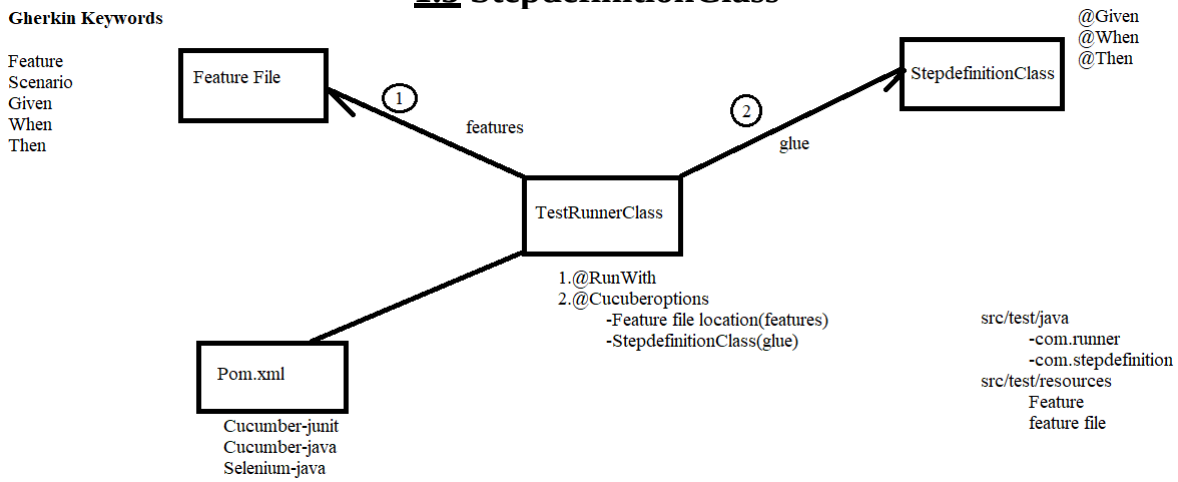

**1.1 Feature File**




## 1.2 TestRunnerClass



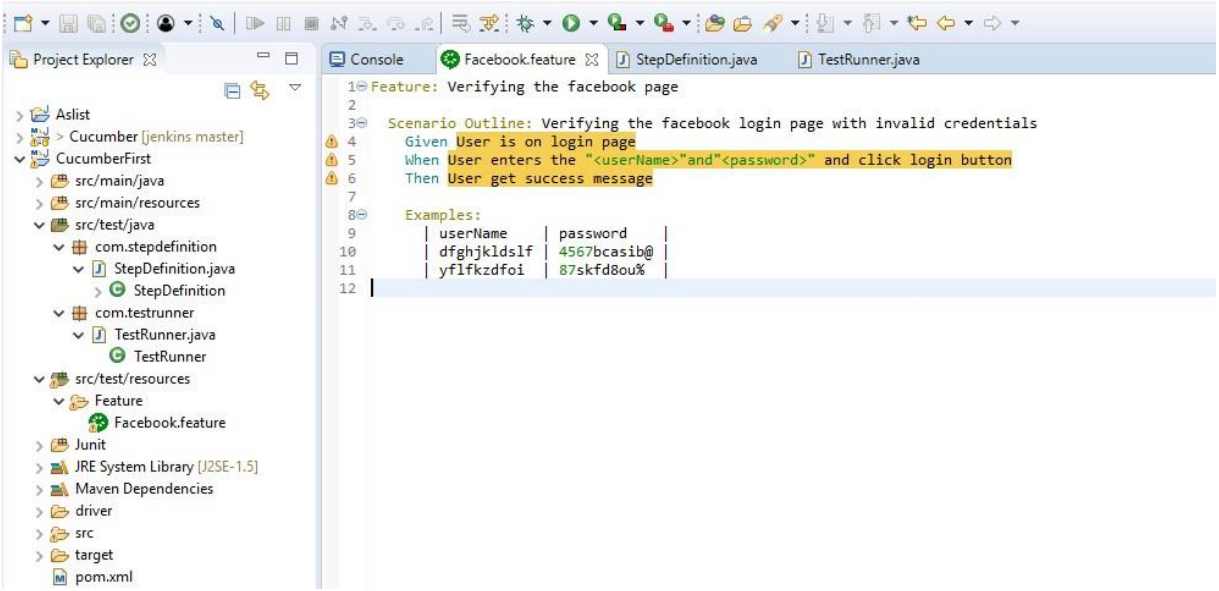
## 1.3 StepdefinitionClass



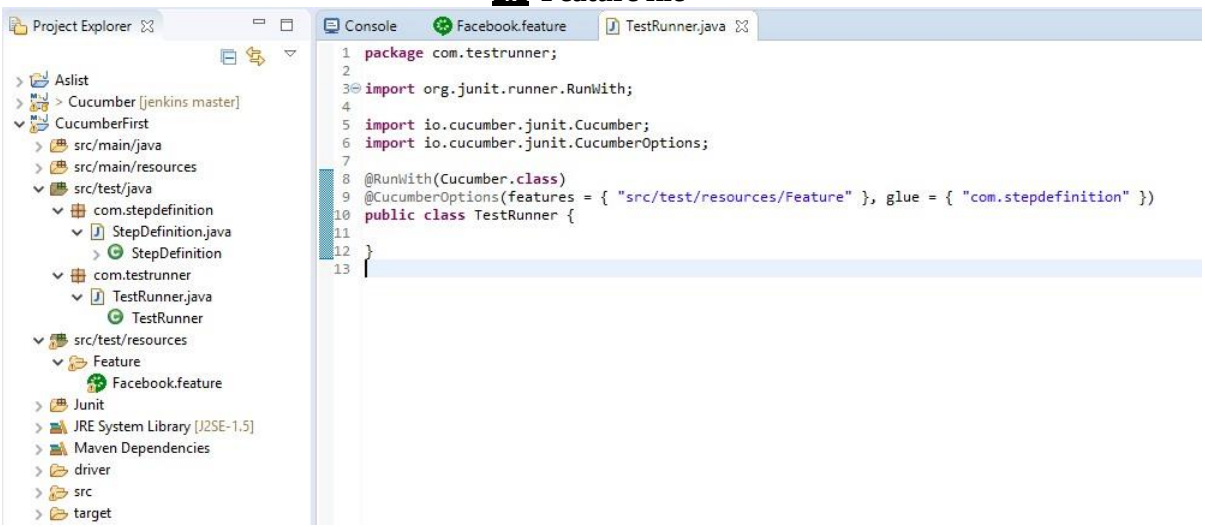

## 1.4 Cucumber Architecture

# 2.LAUNCH FACEBOOK AND PASSING THE PARAMETER USING EXAMPLE

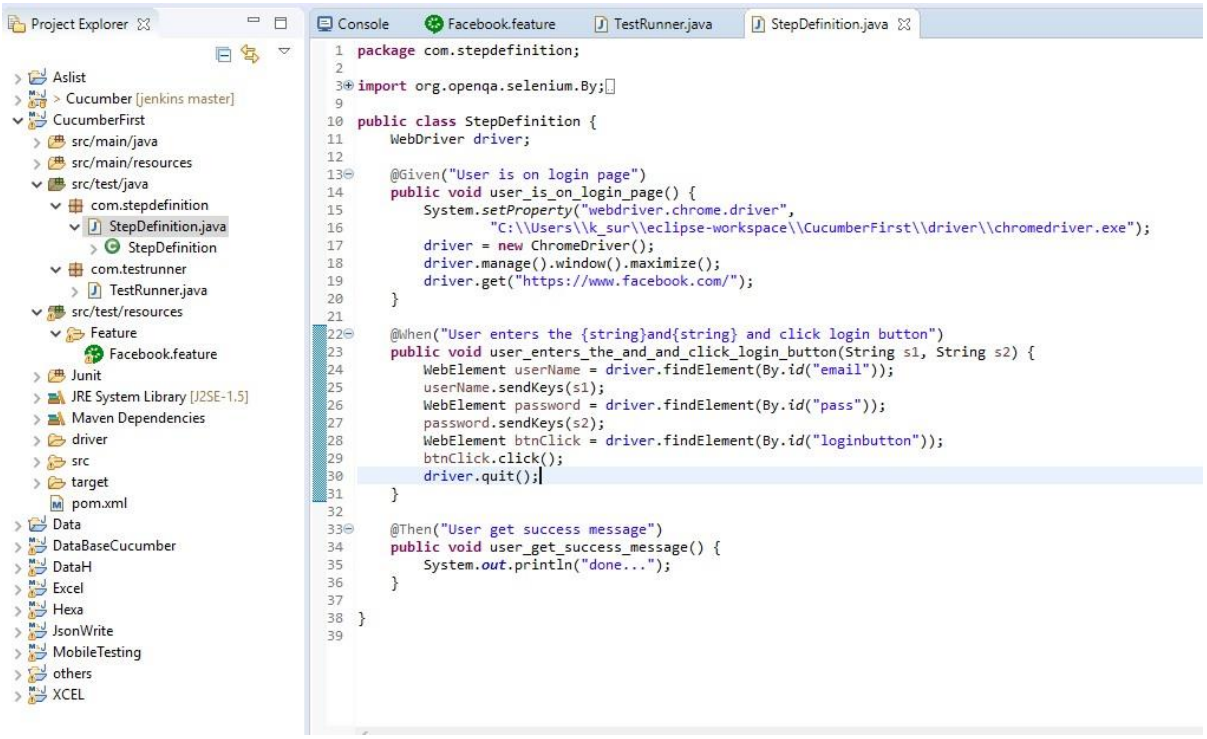
- In cucumber using **Scenario Outline and Example** keyword we can pass multiple parameters for same scenario.




**2.1 Feature file**




**2.2 TestRunnerClass**



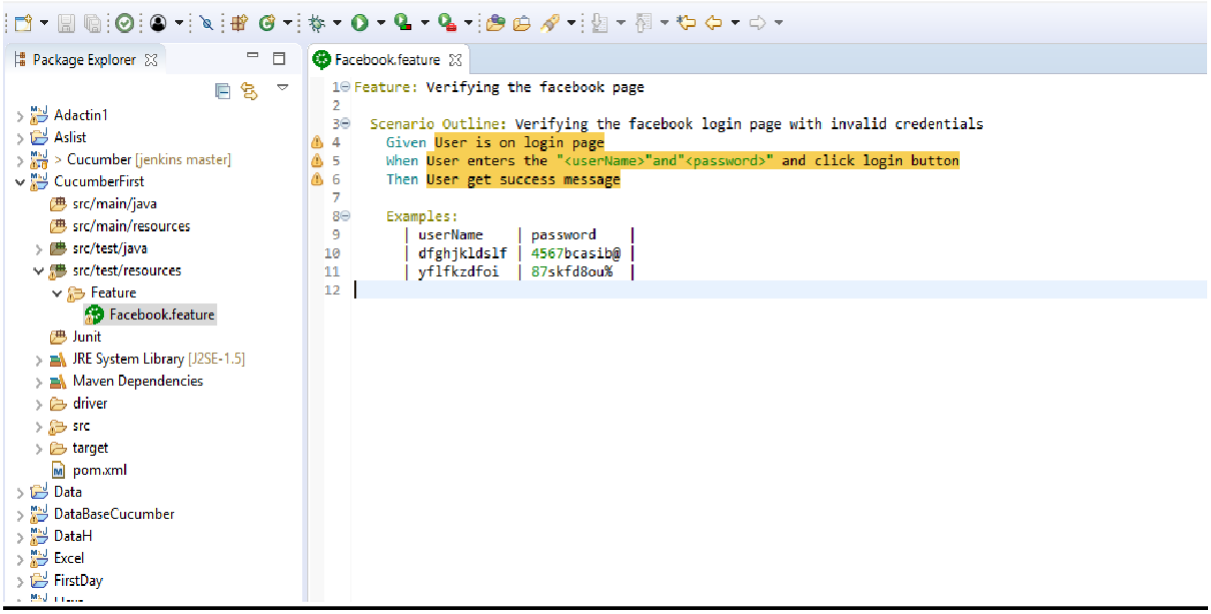

**2.3 StepdefinitionClass**

# 3. DESIGN PATTERNS

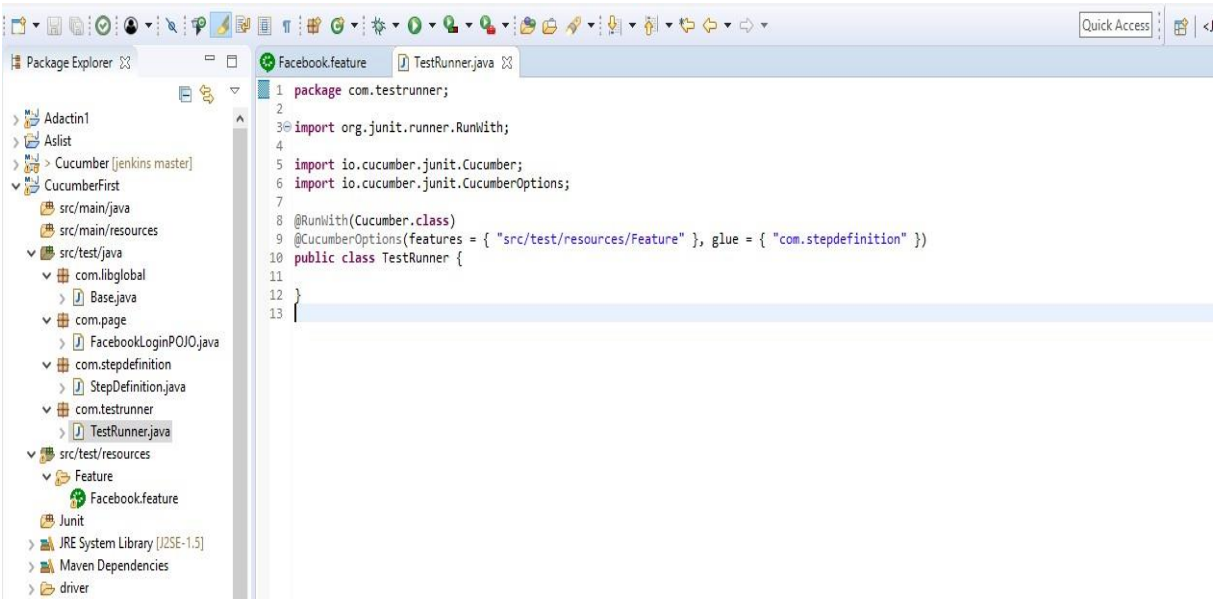
- Page Object Model(POM).
- Singleton.

## 3.1 CUCUMBER WITH BASE CLASS AND POM

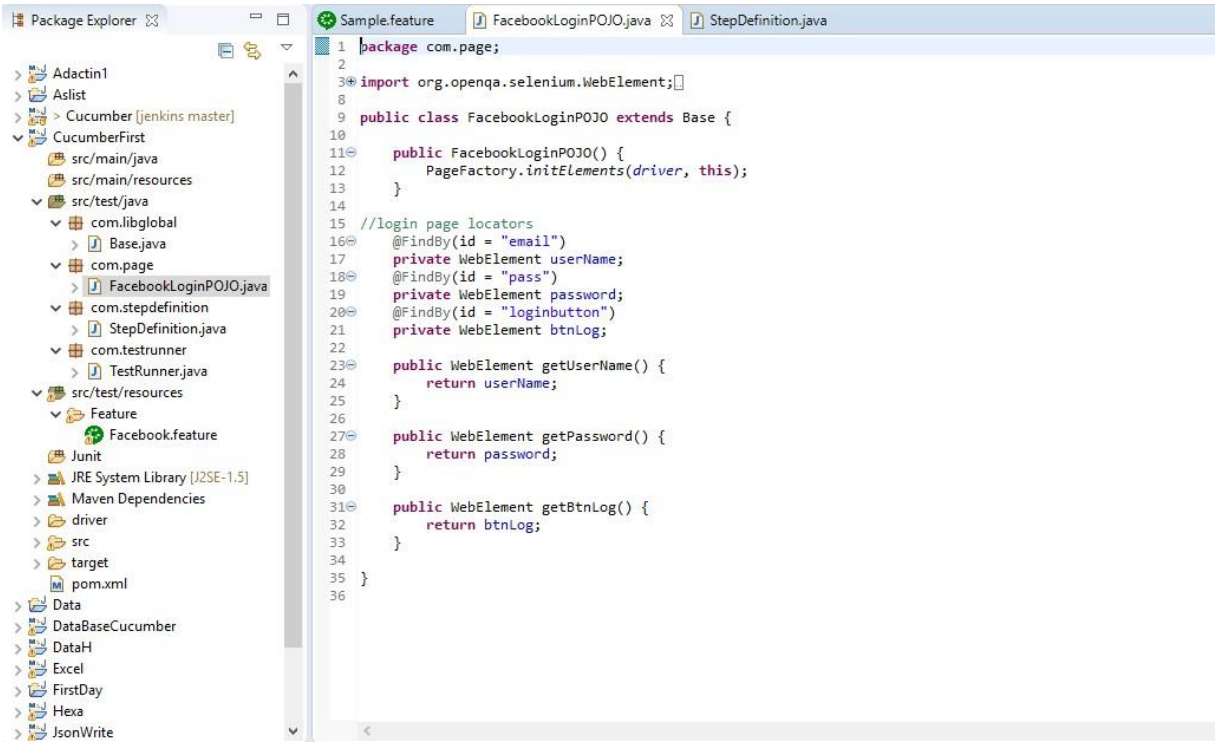
- It is widely **used** design pattern in **Cucumber** for enhancing test maintenance and reducing code duplication.



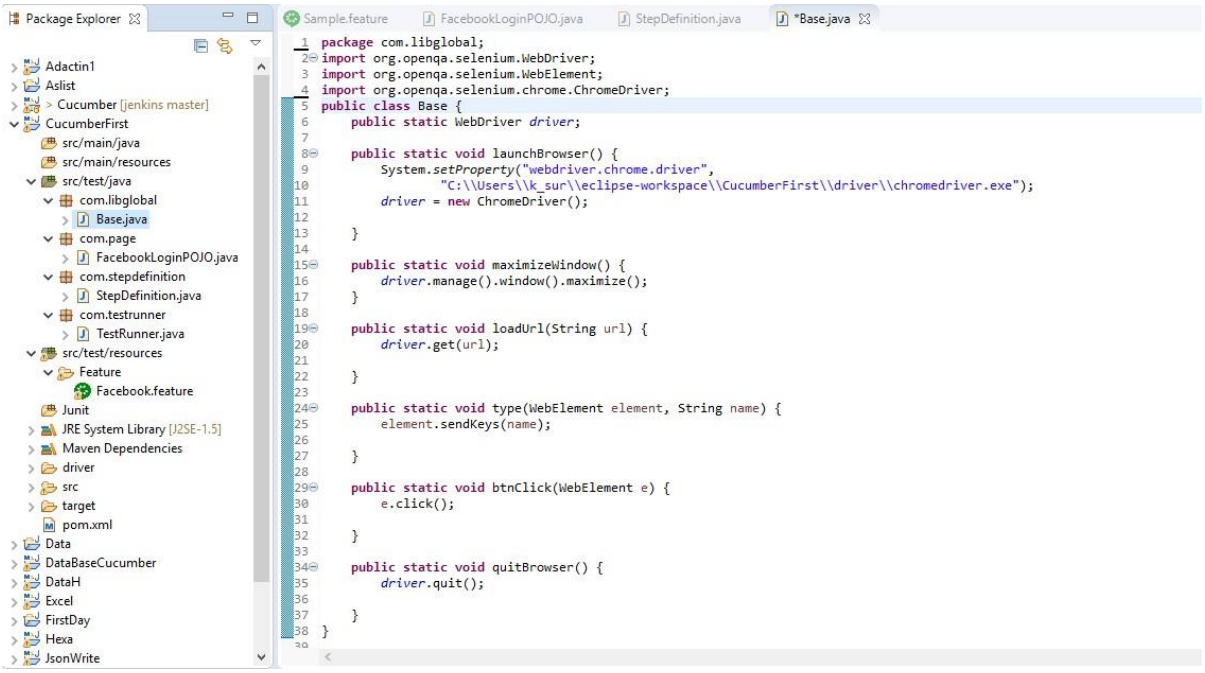

### 3.1.1 Feature file



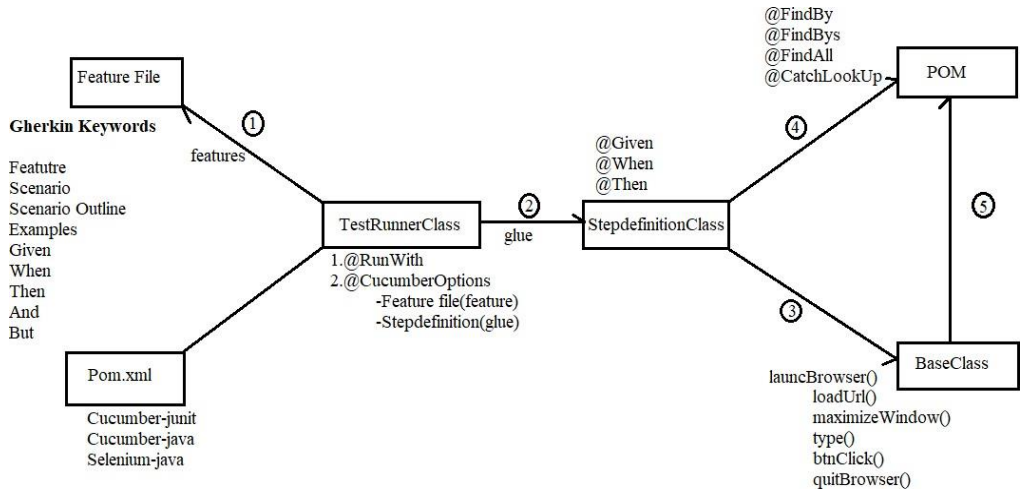

## 3.1.2 TestRunnerClass




## 3.1.3 POJO CLASS



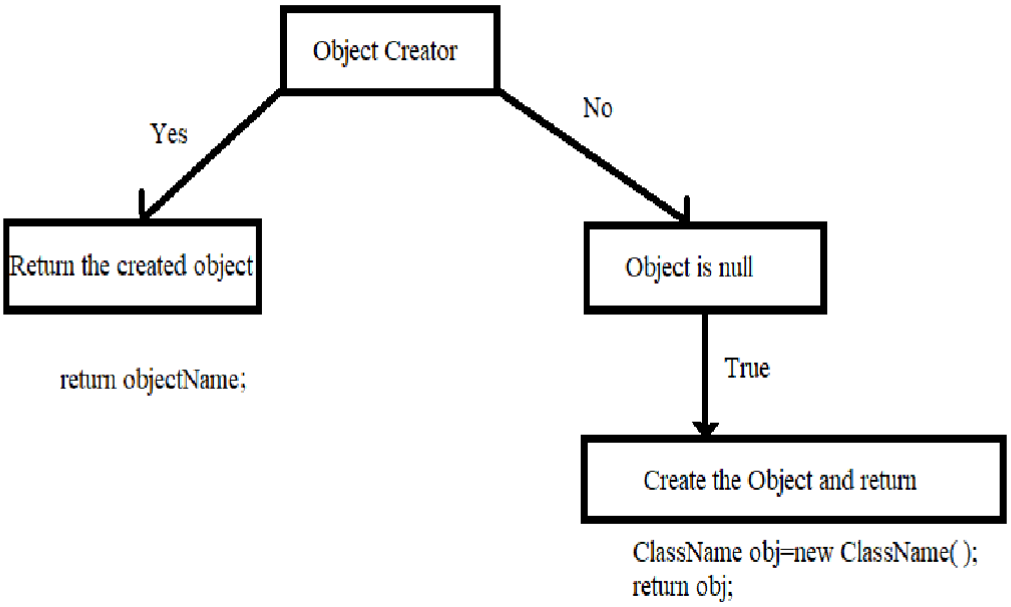

### 3.1.4 BaseClass




### 3.1.5 Architecture3.2

**CUCUMBER WITH SINGLETON**

- Reducing the object creation we go for **singleton** design pattern.

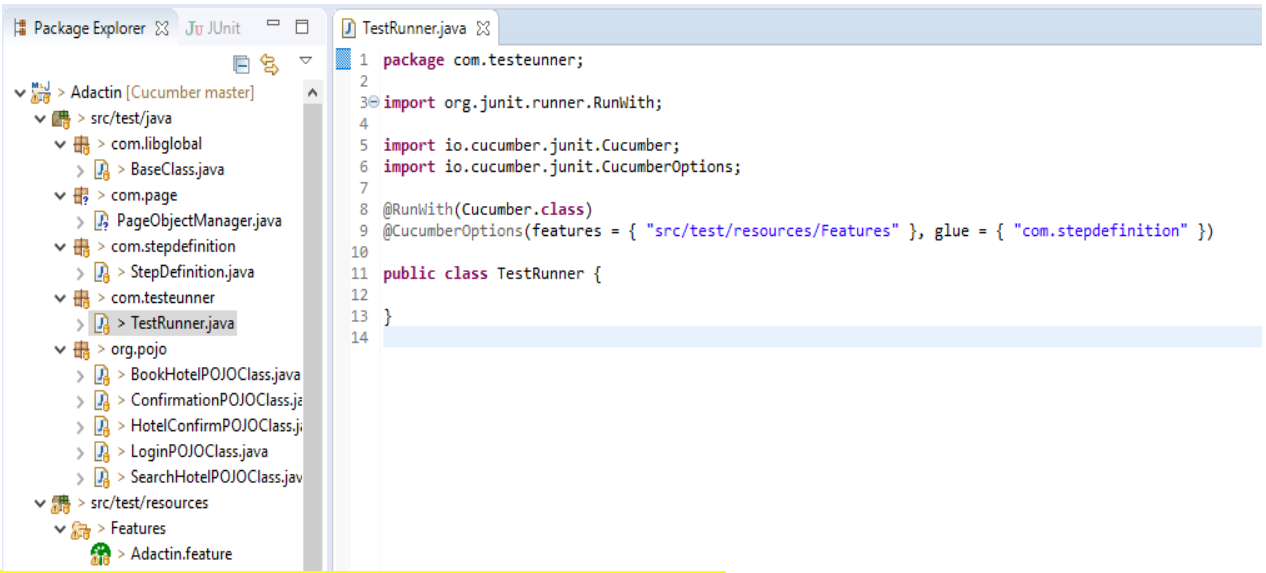



### 3.2.1 SINGLETON

Adactin.feature

```
Feature: Verifing the adactin web page

  Scenario Outline: Verify the login with valid credentials
    Given User is on adactin page
    When User enters "<username>"and"<password>"
    And User clicks login button
    Then User verify success message

    Examples:
      | username         | password |
      | nitishselvakumar | s1740034 |

  Scenario Outline: Verify the Search Hotel page
    When User select "<Location>"and"<Hotels>"and"<Room Type>"and"<Number of Rooms>"and"<Adults per Room>"and"<Children per Room>"
    And User click search button

    Examples:
      | Location | Hotels      | Room Type | Number of Rooms | Adults per Room | Children per Room |
      | Sydney   | Hotel Creek | Standard  | 1 - One         | 1 - One         | 1 - One           |

  Scenario: verify the Select Hotel page
    When User select the Hotel
    And User click the continue button

  Scenario Outline: verify the search hotel page
    When User enter "<First Name>"and"<Last Name>"and"<Billing Address>"and"<Credit Card No>"and"<Credit Card Type>"and"<Select month>"and"<Select year>"and"<CVV Number>"
    And User click BookNow button

    Examples:
      | First Name | Last Name | Billing Address | Credit Card No   | Credit Card Type | Select month | Select year | CVV Number |
      | nitish     | selva     | Chennai         | 1234567890123456 | VISA             | January      |        2022 |        826 |

  Scenario: verify the confirmation page
    When User click search hotel
```

### 3.2.3 Feature File




### 3.2.4 TestRunnerClass

```
package com.libglobal;

import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;import
org.openqa.selenium.support.ui.Select; public class BaseClass {

	public static WebDriver driver;

	public static void launchBrowser() {
		System.setProperty("webdriver.chrome.driver",

	"C:\\Users\\k_sur\\Desktop\\Nitish\\nitish\\Cucumber\\driver\\chromedriver.exe");
		driver = new ChromeDriver();

	}
	public static void maximizeWindow() {

		driver.manage().window().maximize();
	}
	public static void loadUrl(String url) {

		driver.get(url);

	}
	public static void type(WebElement e, String value) {
		e.sendKeys(value);

	}
	public static void btnClick(WebElement e) {e.click();

	}
	public static void getTittle() {

		String title = driver.getTitle();
		System.out.println(title);
	}
	public static void getUrl() {

		String currentUrl = driver.getCurrentUrl();
		System.out.println(currentUrl);
	}
	public static void quitBrowser() {
```

```
		driver.quit();

	}
	public static void selectByVisibleText(WebElement element, String text) {

		Select s = new Select(element);
		s.selectByVisibleText(text);
	}}
```

### 3.2.5 BaseClass

```
package org.pojo;

import java.util.List;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
import com.libglobal.BaseClass;

public class BookHotelPOJOClass extends BaseClass {

	public BookHotelPOJOClass() { PageFactory.initElements(driver,
		this);

	}
	@FindBy(id = "first_name") private
	List<WebElement> fname;
	@FindBy(id = "last_name") private
	List<WebElement> lname;
	@FindBy(id = "address")
	private List<WebElement> Address;
	@FindBy(id = "cc_num")
	private List<WebElement> ccnum;
	@FindBy(id = "cc_type")
	private List<WebElement> cardtype;
	@FindBy(id = "cc_exp_month") private
	List<WebElement> month; @FindBy(id
	= "cc_exp_year") private
	List<WebElement> year; @FindBy(id =
	"cc_cvv")
	private List<WebElement> cvv;
	@FindBy(id = "book_now") private
	List<WebElement> book;
	public List<WebElement> getFname() {
		return fname;

	}
	public List<WebElement> getLname() {
		return lname;

	}
	public List<WebElement> getAddress() {
		return Address;

	}
	public List<WebElement> getCcnum() {
		return ccnum;

	}
	public List<WebElement> getCardtype() {
		return cardtype;

	}
```

```
	public List<WebElement> getMonth() {
		return month;

	}
	public List<WebElement> getYear() {
		return year;

	}
	public List<WebElement> getCvv() {

		return cvv;

	}
	public List<WebElement> getBook() {
		return book;

	}
}
```

### 3.2.6 BookHotelPOJOClass

```
package org.pojo;

import java.util.List;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
import com.libglobal.BaseClass;

public class ConfirmationPOJOClass extends BaseClass {

	public ConfirmationPOJOClass() {
		PageFactory.initElements(driver, this);

	}
	@FindBy(id = "search_hotel")
	private List<WebElement> searchhotel;

	public List<WebElement> getSearchhotel() {
		return searchhotel;

	}
}
```

### 3.2.7 ConfirmationPOJOClass

```
package org.pojo;

import java.util.List;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
import com.libglobal.BaseClass;

public class HotelConfirmPOJOClass extends BaseClass {

	public HotelConfirmPOJOClass() {
		PageFactory.initElements(driver, this);

	}
```

```
	@FindBy(id = "radiobutton_0")
	private List<WebElement> radio;
	@FindBy(id = "continue")
	private List<WebElement> Continue;

	public List<WebElement> getRadio() {
		return radio;

	}
	public List<WebElement> getContinue() {
		return Continue;

	}
}
```

### 3.2.8 HotelConfirmPOJOClass

```
package org.pojo;

import java.util.List;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
import com.libglobal.BaseClass;

public class LoginPOJOClass extends BaseClass {

	public LoginPOJOClass() { PageFactory.initElements(driver,
		this);

	}
	@FindBy(id = "username") private
	List<WebElement> user;@FindBy(id
	= "password") private
	List<WebElement> pass;@FindBy(id
	= "login")
	private List<WebElement> log;

	public List<WebElement> getUser() {
		return user;

	}
	public List<WebElement> getPass() {
		return pass;

	}
	public List<WebElement> getLog() {

		return log;

	}
}
```

### 3.2.8 LoginPOJOClass

```
package org.pojo;

import java.util.List;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy; import
org.openqa.selenium.support.PageFactory;import
```

```
com.libglobal.BaseClass;
public class SearchHotelPOJOClass extends BaseClass {

	public SearchHotelPOJOClass() {

		PageFactory.initElements(driver, this);

	}
	@FindBy(id = "location")
	private List<WebElement> location;
	@FindBy(id = "hotels")
	private List<WebElement> hotels;
	@FindBy(id = "room_type")
	private List<WebElement> roomtype;
	@FindBy(id = "room_nos")
	private List<WebElement> rooms;
	@FindBy(id = "adult_room")
	private List<WebElement> adult_room;
	@FindBy(id = "child_room")
	private List<WebElement> child_room;
	@FindBy(id = "Submit")
	private List<WebElement> submit;

	public List<WebElement> getLocation() {
		return location;

	}
	public List<WebElement> getHotels() {
		return hotels;

	}
	public List<WebElement> getRoomtype() {
		return roomtype;

	}
	public List<WebElement> getRooms() {
		return rooms;

	}
	public List<WebElement> getAdult_room() {
		return adult_room;

	}
	public List<WebElement> getChild_room() {
		return child_room;

	}
	public List<WebElement> getSubmit() {
		return submit;

	}
}
```

### 3.2.9 SearchHotelPOJOClass


## PageObjectManager

```
package com.page;

import org.pojo.BookHotelPOJOClass; import
org.pojo.ConfirmationPOJOClass;import
org.pojo.HotelConfirmPOJOClass;import
org.pojo.LoginPOJOClass; import
org.pojo.SearchHotelPOJOClass;
public class PageObjectManager {

	LoginPOJOClass loginPage;
	SearchHotelPOJOClass serchPage;
	HotelConfirmPOJOClass hotelConfirmPage;
	BookHotelPOJOClass bookingPage;
	ConfirmationPOJOClass confirmPage;
	public LoginPOJOClass getLoginPage() {
		return (loginPage == null) ? loginPage = new LoginPOJOClass() : loginPage;

	}
	public SearchHotelPOJOClass getSerchPage() {
		return (serchPage == null) ? serchPage = new SearchHotelPOJOClass() : serchPage;
	}
	public HotelConfirmPOJOClass getHotelConfirmPage() {
		return (hotelConfirmPage == null) ? hotelConfirmPage = new HotelConfirmPOJOClass() :
hotelConfirmPage;
	}
	public BookHotelPOJOClass getBookingPage() {
		return (bookingPage == null) ? bookingPage = new BookHotelPOJOClass() : bookingPage;
	}
```

```
	public ConfirmationPOJOClass getConfirmPage() {
		return (confirmPage == null) ? confirmPage = new ConfirmationPOJOClass() :

confirmPage;
	}
}
```

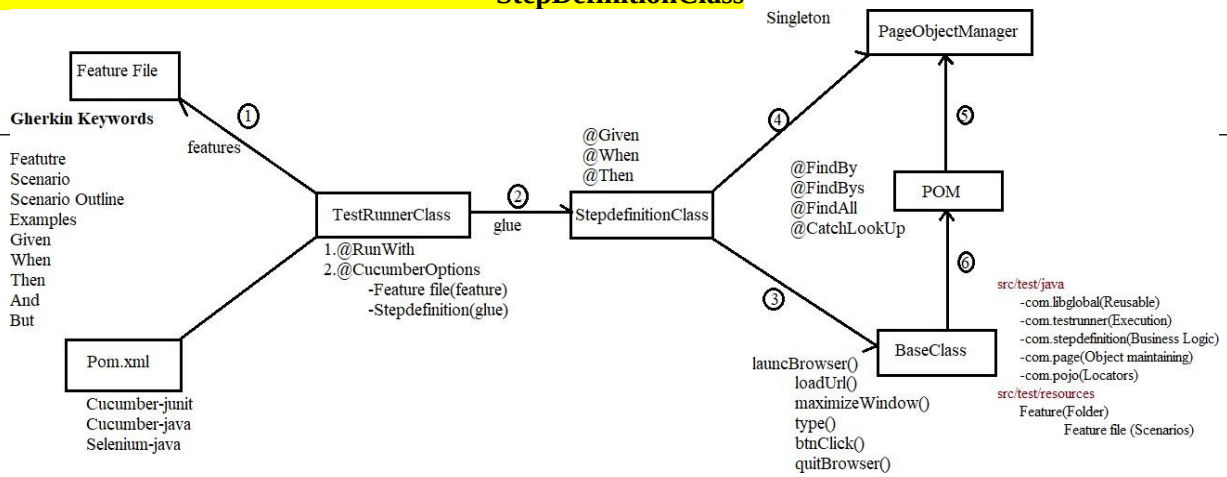
### 3.2.10 PageObjectManager

```
Package com.stepdefinition;

import com.libglobal.BaseClass;

import com.page.PageObjectManager;

import io.cucumber.java.en.*;

public class StepDefinition extends BaseClass { PageObjectManager page =

	new PageObjectManager();


	@Given("User is on adactin page")
	public void user_is_on_adactin_page() { launchBrowser();
		loadUrl("https://adactin.com/HotelApp/");
		maximizeWindow();

	}
	@When("User enters {string}and{string}")
	public void user_enters_and(String userName, String password) {
```

```
		type(page.getLoginPage().getUser().get(0), userName);
		type(page.getLoginPage().getPass().get(0), password);
	}
	@When("User clicks login button")
	public void user_clicks_login_button() {

		btnClick(page.getLoginPage().getLog().get(0));
	}
	@Then("User verify success message")
	public void user_verify_success_message() {
		System.out.println("Successfuly login");
	}
	@When("User select {string}and{string}and{string}and{string}and{string}and{string}")
	public void user_select_and_and_and_and_and(String location, String hotel, String roomType,String
numOfRooms,String adultsPerRoom, String childperRoom) {
		selectByVisibleText(page.getSerchPage().getLocation().get(0), location);
		selectByVisibleText(page.getSerchPage().getHotels().get(0), hotel);
		selectByVisibleText(page.getSerchPage().getRoomtype().get(0), roomType);
		selectByVisibleText(page.getSerchPage().getRooms().get(0), numOfRooms);
```

```
		selectByVisibleText(page.getSerchPage().getAdult_room().get(0), adultsPerRoom);

		selectByVisibleText(page.getSerchPage().getChild_room().get(0), childperRoom);

	}
	@When("User click search button")
	public void user_click_search_button() {

		btnClick(page.getSerchPage().getSubmit().get(0));
	}
	@When("User select the Hotel")
	public void user_select_the_Hotel() {

		btnClick(page.getHotelConfirmPage().getRadio().get(0));
	}
	@When("User click the continue button")
	public void user_click_the_continue_button() {

		btnClick(page.getHotelConfirmPage().getContinue().get(0));
	}
	@When("User enter
{string}and{string}and{string}and{string}and{string}and{string}and{string}and{string}")
	public void user_enter_and_and_and_and_and_and_and(String firstNmae, String lastName, String
billingAddress,String creditCardNo, String creditCardType, String expiryMonth, String expiryYear, String cvvNumber) {
		type(page.getBookingPage().getFname().get(0), firstNmae);
		type(page.getBookingPage().getLname().get(0), lastName);
		type(page.getBookingPage().getAddress().get(0), billingAddress);
		type(page.getBookingPage().getCcnum().get(0), creditCardNo);
		selectByVisibleText(page.getBookingPage().getCardtype().get(0), creditCardType);
		selectByVisibleText(page.getBookingPage().getMonth().get(0), expiryMonth);
		selectByVisibleText(page.getBookingPage().getYear().get(0), expiryYear);
		type(page.getBookingPage().getCvv().get(0), cvvNumber);
	}
	@When("User click BookNow button")
	public void user_click_BookNow_button() {

		btnClick(page.getBookingPage().getBook().get(0));
	}
	@When("User click search hotel")
	public void user_click_search_hotel() throws InterruptedException {Thread.sleep(5000);
		btnClick(page.getConfirmPage().getSearchhotel().get(0)); quitBrowser();

	}
}
```

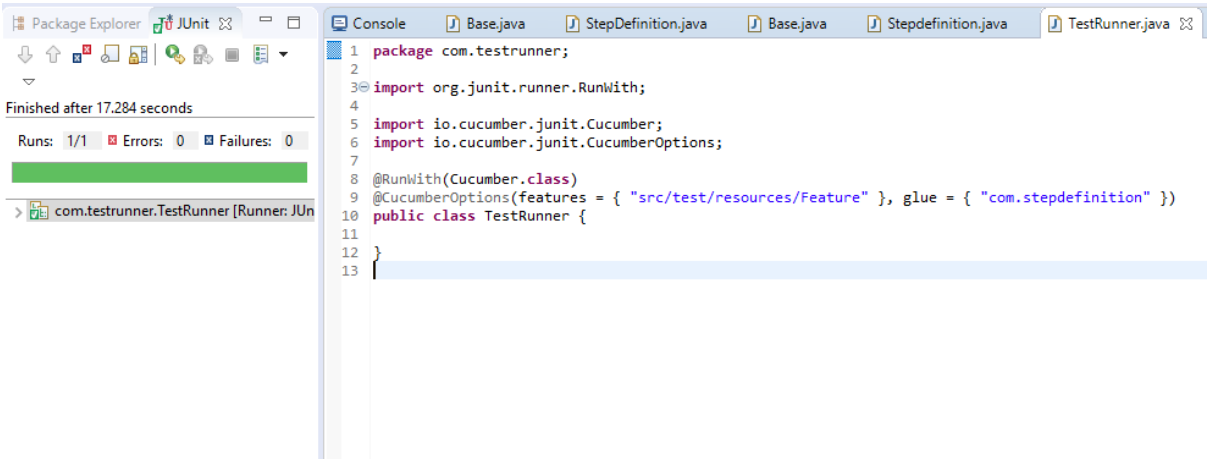
### 3.2.11 StepDefinitionClass



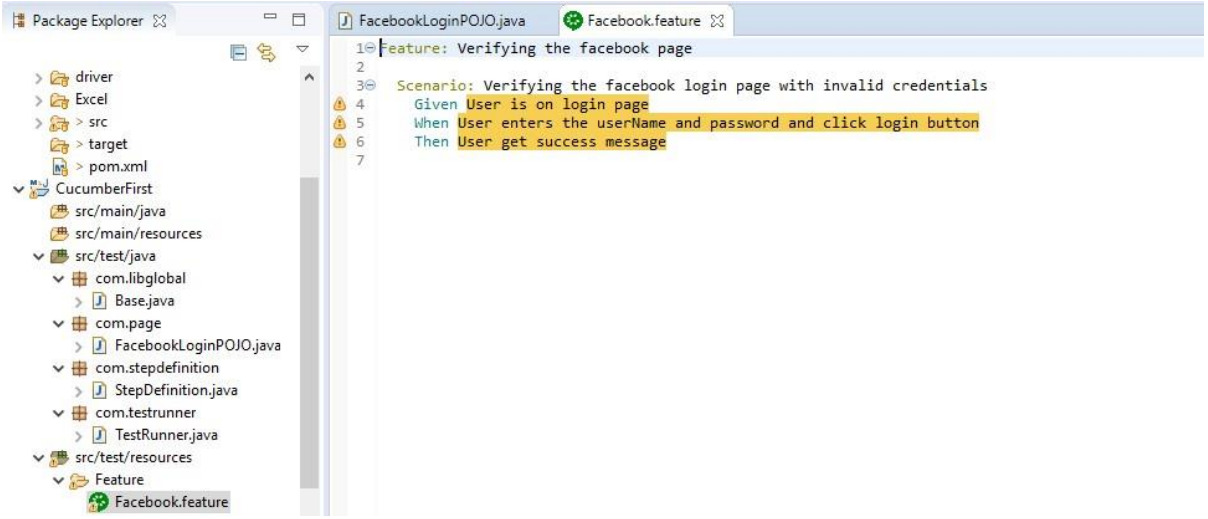
## 3.2.12 Cucumber Architecture

**TO PASS THE PARAMETER USING SQL IN CUCUMBER**

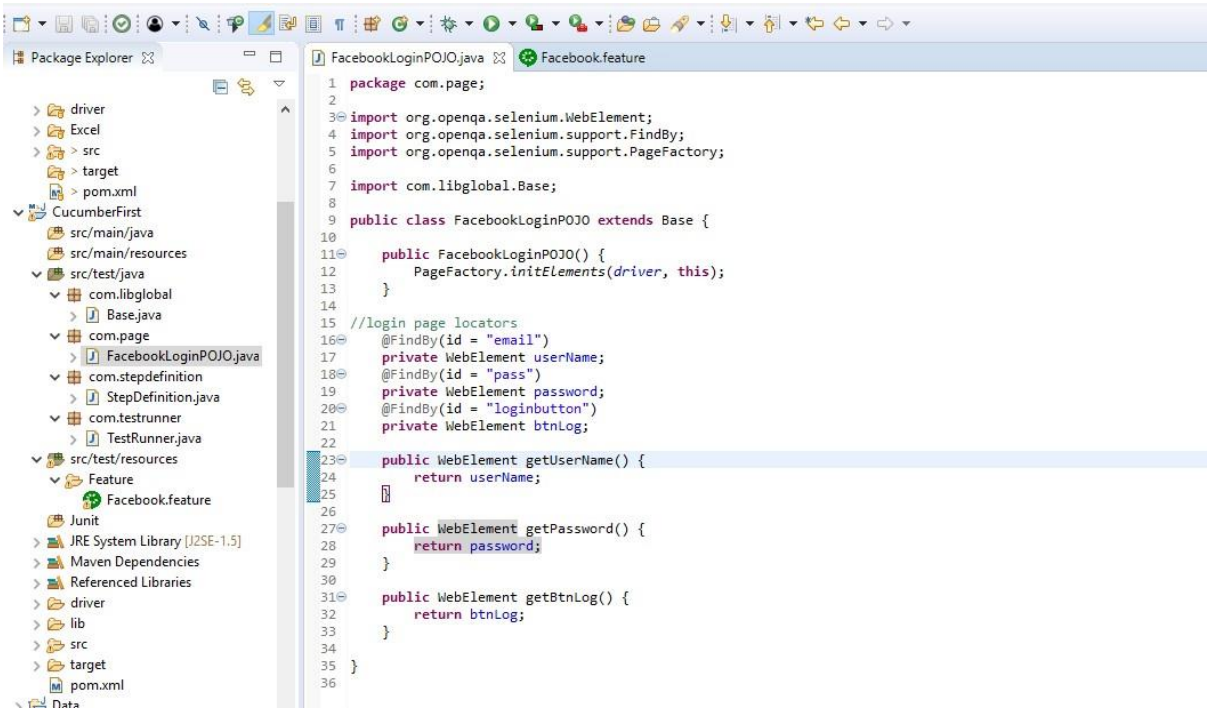
1. Create maven project and add ojdbc jar.
2. Pass the JDBC connection in base class.




**TestRunnerClass**




**Feature File**




**PojoClass**

```
package com.libglobal;

import   java.sql.Connection; import
java.sql.DriverManager; import
java.sql.PreparedStatement;import
java.sql.ResultSet;

import java.sql.SQLException;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class Base {
```

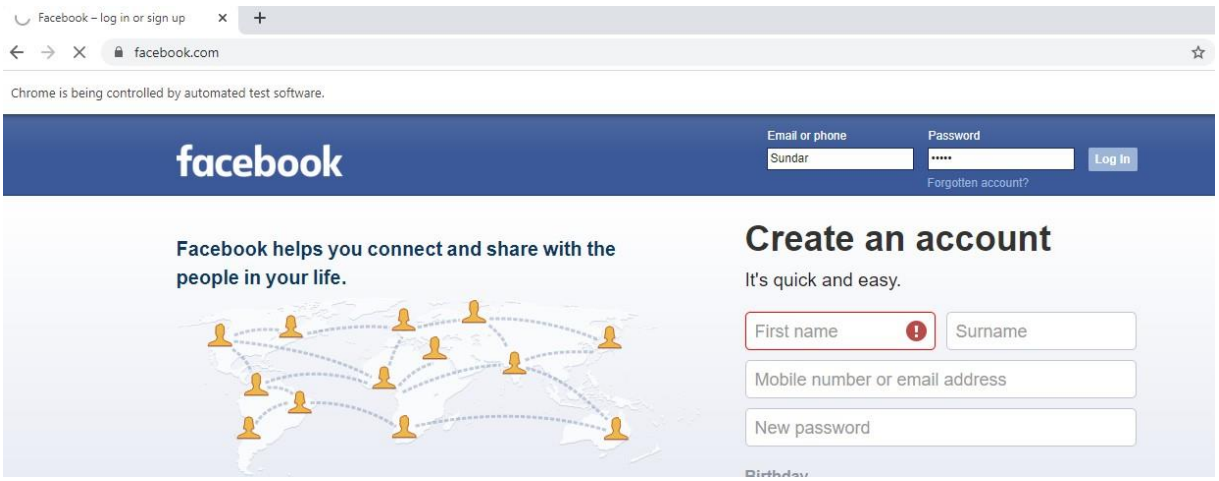
```
	public static void launchBrowser() {
		System.setProperty("webdriver.chrome.driver",

				"C:\\Users\\k_sur\\eclipseworkspace\\CucumberFirst\\driver\\chromedriver.exe");
		driver = new ChromeDriver();

	}
	public static void maximizeWindow() {

		driver.manage().window().maximize();
	}
	public static void loadUrl(String url) {

		driver.get(url);

	}
	public static void type(WebElement element, String name) {
		element.sendKeys(name);

	}
	public static void btnClick(WebElement e) {e.click();

	}
	public static void quitBrowser() {
```

```
        driver.quit();

    }
    public static String getDataFromSql(String query) throws Throwable {Connection con =
        null;

        String name = null;try {

            Class.forName("oracle.jdbc.driver.OracleDriver");
        con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "hr", "admin");
            PreparedStatement ps = con.prepareStatement(query);ResultSet
            rs = ps.executeQuery();
            while (rs.next()) {

                name = rs.getString("first_name");
            }
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
          } finally {

            try {
                con.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        return name;

    }
}
```

**BaseClass**

```
package com.stepdefinition;

import com.libglobal.Base;

import com.page.FacebookLoginPOJO;

import io.cucumber.java.en.*;


public class StepDefinition extends Base {@Given("User

	is on login page")

	public void user_is_on_login_page() {

		launchBrowser();// launch the browser
		maximizeWindow();// maximize the window
		loadUrl("https://www.facebook.com/");

	}
	@When("User enters the userName and password and click login button")
	public void user_enters_the_userName_and_password_and_click_login_button() throws Throwable {
		FacebookLoginPOJO f = new FacebookLoginPOJO();
		// get data from sql
		type(f.getUserName(), getDataFromSql("select first_name from employees where
first_name='Sundar'"));
		type(f.getPassword(), getDataFromSql("select first_name from employees where
first_name='Ellen'"));
		btnClick(f.getBtnLog());// clicks the login button
		quitBrowser();// quits the browser

	}
	@Then("User get success message")
	public void user_get_success_message() {
		System.out.println("done...");
```

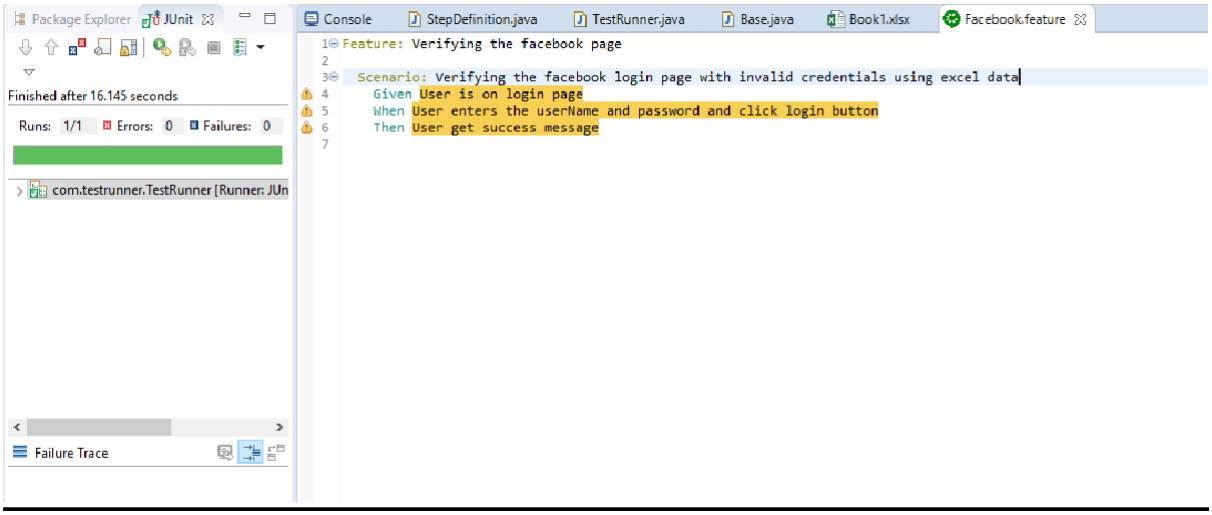
**StepDefinitionClass**



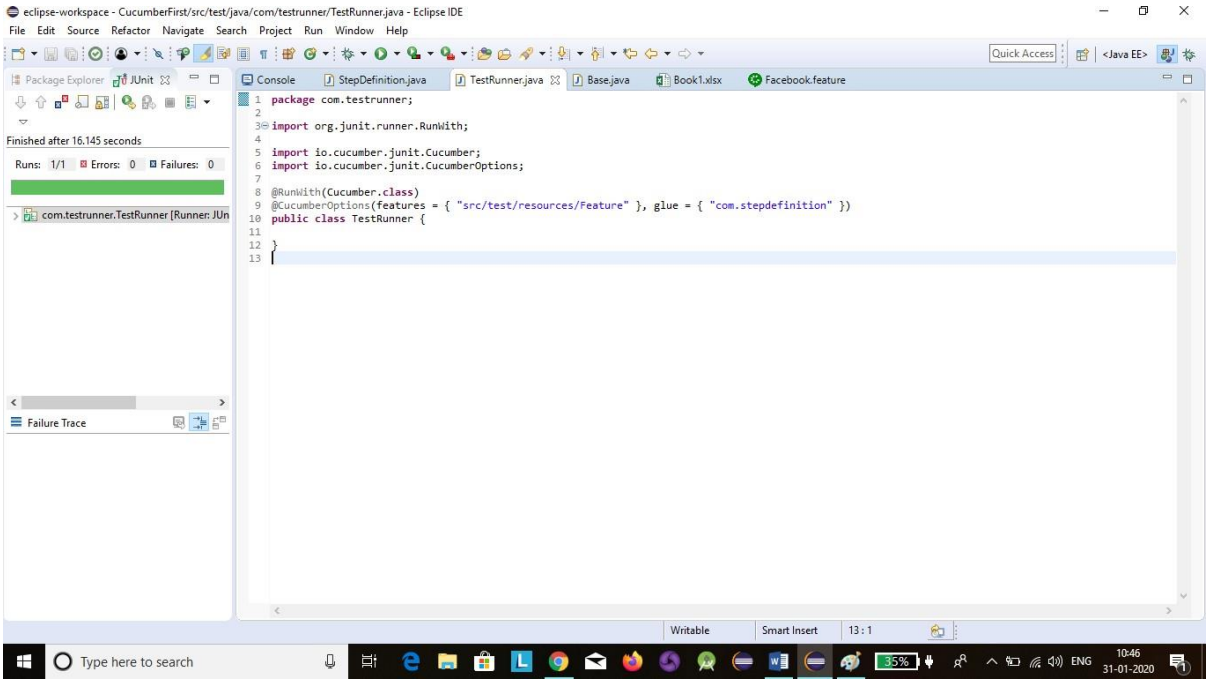

**Output**

## TO PASSING THE PARAMETER USING EXCEL IN CUCUMBER

1. Create Maven project add poi-ooxml,commons-io dependencies.
2. Create Folder and copy paste the Excel.




**Feature file**

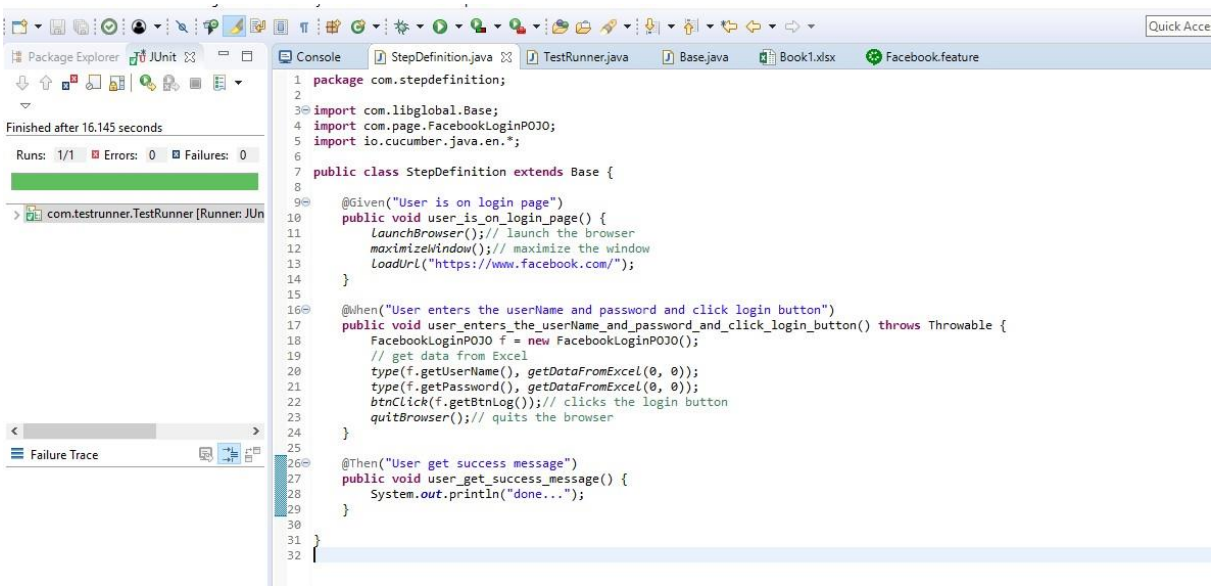



**TestRunnerClass**

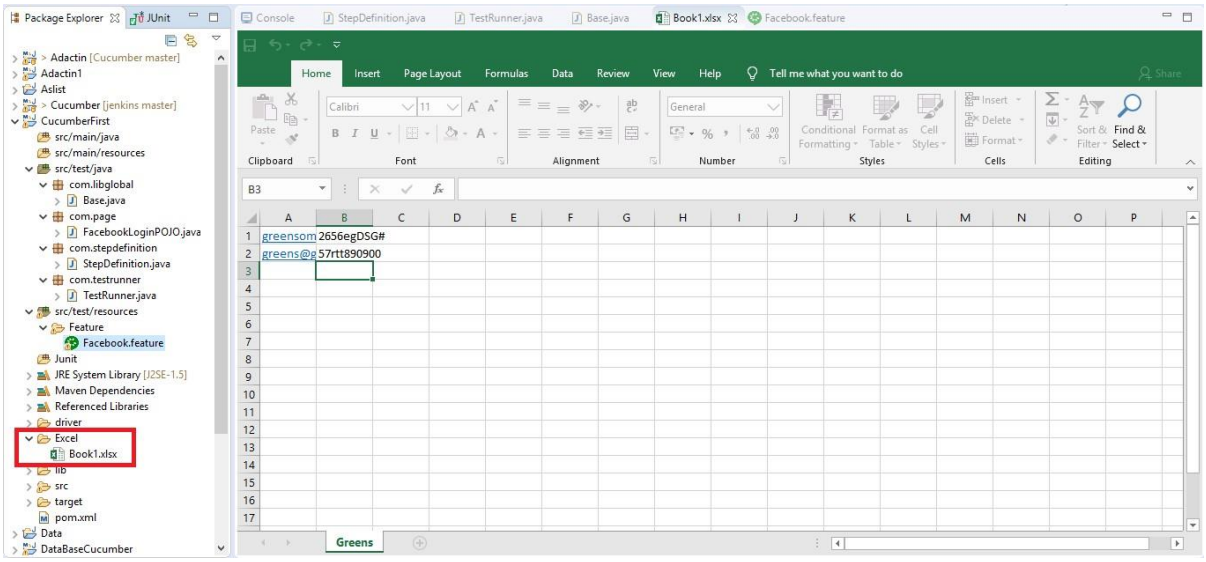
```
package com.libglobal;

import java.io.File;

import java.io.FileInputStream; import
java.io.IOException; import
java.text.SimpleDateFormat;import
java.util.Date;

import org.apache.poi.ss.usermodel.Cell; import
org.apache.poi.ss.usermodel.DateUtil;import
org.apache.poi.ss.usermodel.Row; import
org.apache.poi.ss.usermodel.Sheet; import
org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class Base {

	public static WebDriver driver;

	public static void launchBrowser() {
		System.setProperty("webdriver.chrome.driver",

					"C:\\Users\\k_sur\\eclipse-
workspace\\CucumberFirst\\driver\\chromedriver.exe");
		driver = new ChromeDriver();

	}
	public static void maximizeWindow() {

		driver.manage().window().maximize();
	}
	public static void loadUrl(String url) {

		driver.get(url);

	}
	public static void type(WebElement element, String name) {
		element.sendKeys(name);

	}
	public static void btnClick(WebElement e) {e.click();

	}
	public static void quitBrowser() {

		driver.quit();

	}
	@SuppressWarnings({ "deprecation", "resource" })
	public static String getDataFromExcel(int row, int cell) throws IOException {String value = null;

		File loc = new

File("C:\\Users\\k_sur\\eclipseworkspace\\CucumberFirst\\Excel\\Book1.xlsx");
```

```
                FileInputStream stream = new FileInputStream(loc);
                Workbook w = new XSSFWorkbook(stream);
                Sheet s = w.getSheet("Greens");Row r
                = s.getRow(row);
                Cell c = r.getCell(cell); int type =

                c.getCellType();if (type == 1) {

                        value = c.getStringCellValue();
                } else if (type == 0) {

                        if (DateUtil.isCellDateFormatted(c)) {

                                Date dateCellValue = c.getDateCellValue(); SimpleDateFormat sim =
                                new SimpleDateFormat("dd-mm-yyyy");value =
                                sim.format(dateCellValue);
                        } else {

                                double numericCellValue = c.getNumericCellValue();
                                long l = (long) numericCellValue;value =
                                String.valueOf(l);

                        }
                }
                return value;

        }
}
```

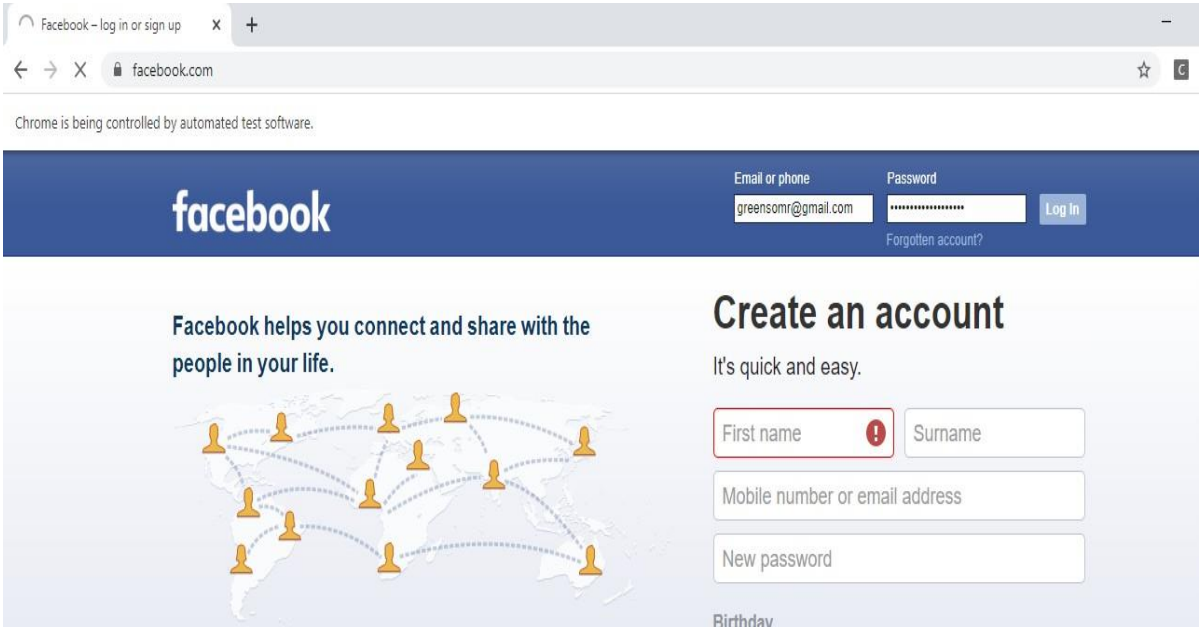
**BaseClass**




**StepDefinitionClass**




**Excel File**

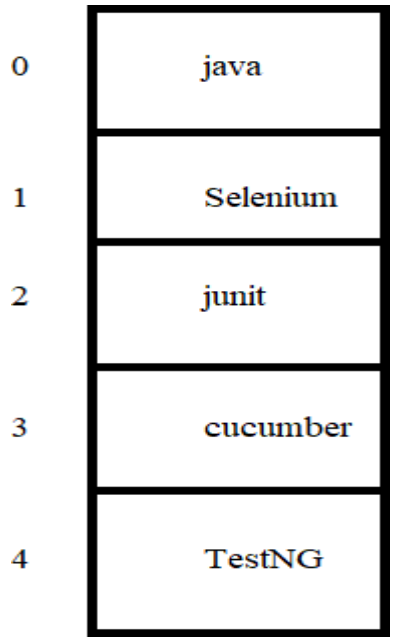



## Output

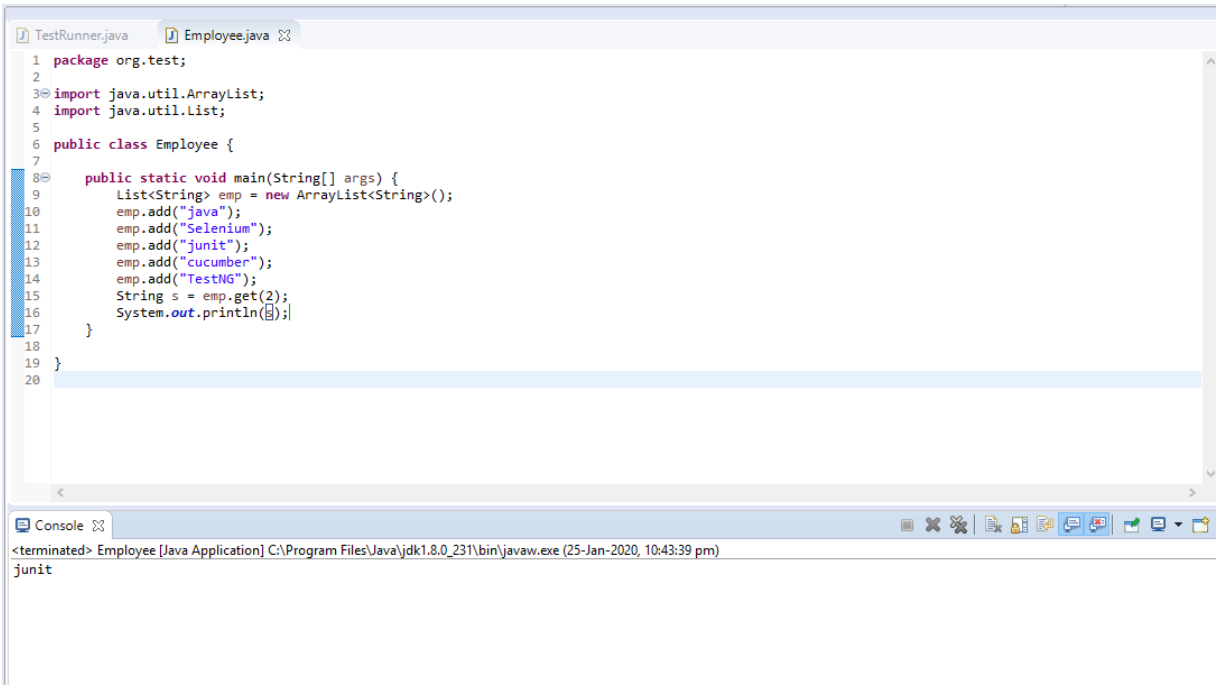
**DATA TABLE IN CUCUMBER**

1. asList(1D-Without Header).
2. asLists(2D-Without Header).
3. asMap(1D-With Header).
4. asMaps(2D-With Header).

**1.1D-WITHOUT HEADER(asList)**

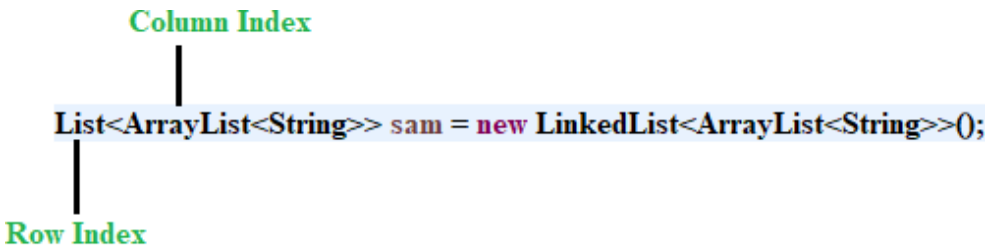



**1D –List Diagram**




**1D-Without Header**

## 2.2D-WITHOUT HEADER(asLists)




| | 0 | 1 | 2 |
|---|---|---|---|
| 0 | bala | bala@gmail .com | selenium |
| 1 | karthi | karthi@gmail .com | python |
| 2 | kiruba | kiruba@gmail .com | sql |

**2D-List Diagram**

Sample.java

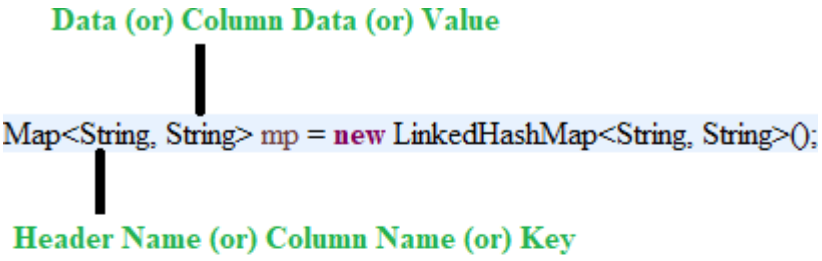
```
package org.test;

import java.util.ArrayList;
import java.util.LinkedList;
import java.util.List;

public class Sample {

    public static void main(String[] args) {
        // Outer list
        List<ArrayList<String>> sam = new LinkedList<ArrayList<String>>();
        // Inner list
        ArrayList<String> s1 = new ArrayList<String>();
        s1.add("bala");
        s1.add("bala@gmail.com");
        s1.add("selenium");
        ArrayList<String> s2 = new ArrayList<String>();
        s2.add("karthi");
        s2.add("karthi@gmail.com");
        s2.add("python");
        ArrayList<String> s3 = new ArrayList<String>();
        s3.add("kiruba");
        s3.add("kiruba@gmail.com");
        s3.add("sql");
        sam.add(s1);
        sam.add(s2);
        sam.add(s3);
        // get the values from outer list
        ArrayList<String> list = sam.get(1);
        // get the values from inner list
        String string = list.get(1);
        System.out.println(string);

    }
}
```

Console

```
<terminated> Sample (3
karthi@gmail.com
```

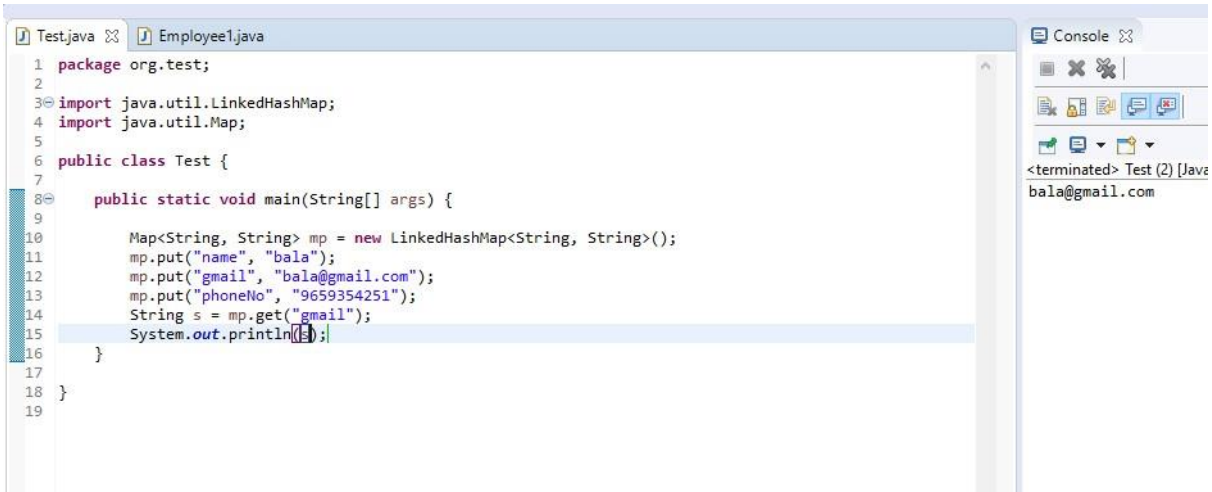
**2D-Without Header**

## 3.1 D-WITH HEADER(asMap)





**1D-Map Diagram**




**1D-With Header**

## 3.2 D-WITH HEADER(asMaps)

Key(or)Column Header

```
List<LinkedHashMap<String, String>> li = new LinkedList<LinkedHashMap<String, String>>();
```

Index

Value(or)Column value

| | name | gmail | phoneNo |
|---|---|---|---|
| 0 | nitish | nitish2198 @gmail.com | 7896541230 |
| 1 | merin | prbmerin @gmail.com | 9321654870 |

**2D-Map Diagram**

Test.java | AsMaps.java

```
1  package org.test;
2
3  import java.util.LinkedHashMap;
6
7  public class AsMaps {
8
9      public static void main(String[] args) {
10         // outter list declaration
11         List<LinkedHashMap<String, String>> li = new LinkedList<LinkedHashMap<String, String>>();
12         // inner map declaration
13         LinkedHashMap<String, String> mp1 = new LinkedHashMap<String, String>();
14         mp1.put("name", "nitish");
15         mp1.put("gmail", "nitish2198@gmail.com");
16         mp1.put("phoneNo", "7896541230");
17         LinkedHashMap<String, String> mp2 = new LinkedHashMap<String, String>();
18         mp2.put("name", "merin");
19         mp2.put("gmail", "prbmerin@gmail.com");
20         mp2.put("phoneNo", "9321654870");
21         li.add(mp1);
22         li.add(mp2);
23         LinkedHashMap<String, String> asMaps = li.get(1);
24         String s = asMaps.get("gmail");
25         System.out.println(s);
26     }
27
28 }
29
```

Console

```
<terminated> AsMaps [Java
prbmerin@gmail.com
```

**2D-With Header**

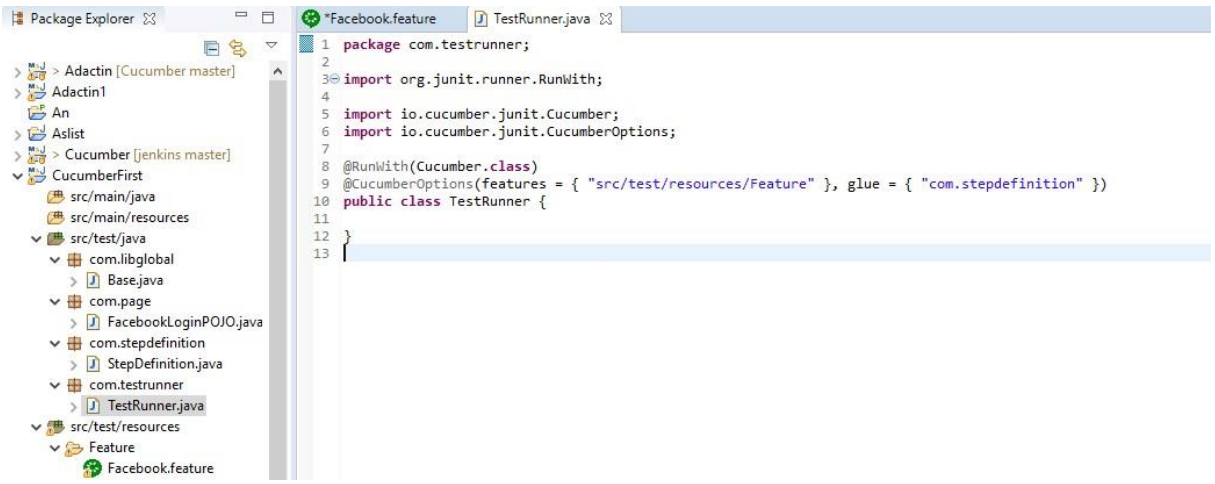
Package Explorer

- Adactin [Cucumber master]
- Adactin1
- An
- Aslist
- Cucumber [jenkins master]
- CucumberFirst
  - src/main/java
  - src/main/resources
  - src/test/java
  - src/test/resources
    - Feature
      - Facebook.feature
  - Junit
  - JRE System Library [J2SE-1.5]
  - Maven Dependencies
  - Referenced Libraries
  - driver
  - Excel
  - lib
  - src
  - target
  - pom.xml
- Data
- DataBaseCucumber
- DataH
- Excel
- FirstDay

*Facebook.feature

```
Feature: Verifying the facebook page

  Background:
    Given User is on login page

  Scenario: Verifying the facebook login page with invalid credentials
    When User enters the userName and password using data table ONE DIMENTIONAL LIST and click login button
      #ONE DIMENTIONAL LIST (OR) asList
      | bala@gmail.com |
      | !2678@gmsK     |
    Then User get success message

  Scenario: Verifying the facebook login page with invalid credentials
    When User enters the userName and password using data table TWO DIMENTIONAL LIST and click login button
      #TWO DIMENTIONAL LIST (OR) asListS
      | bala@gmail.com   | !2678@gmsK |
      | kiruba@gmail.com | 2&@hmbmK   |
    Then User get verification message

  Scenario: Verifying the facebook login page with invalid credentials using data table
    When User enters the username and password ONE DIMENTIONAL MAP and click login button
      #ONE DIMENTIONAL MAP (OR) asMap
      | userName | bala@gmail.com |
      | password | !2678@gmsK     |
    Then User get message

  Scenario: Verifying the facebook login page with invalid credentials
    When User enters the userName using data table TWO DIMENTIONAL MAP and password and click login button
      #TWO DIMENTIONAL MAP (OR) asMaps
      | userName         | password   |
      | bala@gmail.com   | !2678@gmsK |
      | kiruba@gmail.com | 2&@hmbmK   |
    Then User get some message
```

**Feature File**




## TestRunnerClass

```
package com.libglobal;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class Base {

	public static WebDriver driver;

	public static void launchBrowser() {
		System.setProperty("webdriver.chrome.driver",

					"C:\\Users\\k_sur\\eclipse-
workspace\\CucumberFirst\\driver\\chromedriver.exe");
		driver = new ChromeDriver();

	}
	public static void maximizeWindow() {

		driver.manage().window().maximize();
	}
	public static void loadUrl(String url) {

		driver.get(url);

	}
	public static void type(WebElement element, String name) {
		element.sendKeys(name);

	}
	public static void btnClick(WebElement e) {e.click();

	}
	public static void quitBrowser() {

		driver.quit();

	}
}
```

**BaseClass**

```
package com.page;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
import com.libglobal.Base;

public class FacebookLoginPOJO extends Base {

	public FacebookLoginPOJO() {
		PageFactory.initElements(driver, this);

	}
//login page locators
	@FindBy(id = "email") private
	WebElement userName;
	@FindBy(id   =   "pass") private
	WebElement password;
	@FindBy(id = "loginbutton")
	private WebElement btnLog;
	public WebElement getUserName() {

		return userName;

	}
	public WebElement getPassword() {

		return password;

	}
	public WebElement getBtnLog() {

		return btnLog;
```

**Login POJO**

```
package com.stepdefinition;

import java.util.List;

import java.util.Map;

import com.libglobal.Base;

import com.page.FacebookLoginPOJO;

import io.cucumber.java.en.*;

public class StepDefinition extends Base {
	FacebookLoginPOJO f;

	@Given("User is on login page")
	public void user_is_on_login_page() { launchBrowser();
		loadUrl("https://www.facebook.com/");
		maximizeWindow();

	}
	@When("User enters the userName and password using data table ONE DIMENTIONAL LIST and clicklogin button")
	public void user_enters_the_userName_and_password_using_data_table_ONE_DIMENTIONAL_LIST_and_click_login_button(
			io.cucumber.datatable.DataTable dataFromList) {
		f = new FacebookLoginPOJO();
		List<String> list = dataFromList.asList(); type(f.getUserName(),
		list.get(0));// 1d without headertype(f.getPassword(), list.get(1));// 1d
		without headerbtnClick(f.getBtnLog());
	}
	@Then("User get success message")
	public void user_get_success_message() {
		quitBrowser(); System.out.println("done...");

	}
	@When("User enters the userName and password using data table TWO DIMENTIONAL LIST and clicklogin button")
	public void user_enters_the_userName_and_password_using_data_table_TWO_DIMENTIONAL_LIST_and_click_login_button(
			io.cucumber.datatable.DataTable dataFromLists) {
		f = new FacebookLoginPOJO();
		List<List<String>> lists = dataFromLists.asLists(); type(f.getUserName(),
		lists.get(0).get(0));// 2d without headertype(f.getPassword(),
		lists.get(0).get(1));// 2d without headerbtnClick(f.getBtnLog());
	}
	@Then("User get verification message")
	public void user_get_verification_message() {quitBrowser();
		System.out.println("done...");

	}
```

```
	@When("User enters the username and password ONE DIMENTIONAL MAP and click login button")
	public void user_enters_the_username_and_password_ONE_DIMENTIONAL_MAP_and_click_login_button(
			io.cucumber.datatable.DataTable dataFromMap) {
		f = new FacebookLoginPOJO();
		Map<String, String> map = dataFromMap.asMap(String.class, String.class);
		type(f.getUserName(), map.get("userName"));// 1d with header type(f.getPassword(),
		map.get("password"));// 1d with header btnClick(f.getBtnLog());
	}
	@Then("User get message")
	public void user_get_message() { quitBrowser();
		System.out.println("done...");

	}
	@When("User enters the userName using data table TWO DIMENTIONAL MAP and password and clicklogin
button")
	public void

user_enters_the_userName_using_data_table_TWO_DIMENTIONAL_MAP_and_password_and_click_login_button(
			io.cucumber.datatable.DataTable dataFromMaps) {
		f = new FacebookLoginPOJO();
		List<Map<String, String>> maps = dataFromMaps.asMaps();
		type(f.getUserName(),	maps.get(1).get("userName"));//	2d	with	header
		type(f.getPassword(),	maps.get(0).get("password"));//	2d	with	header
		btnClick(f.getBtnLog());
	}
	@Then("User get some message")
	public void user_get_some_message() {
		quitBrowser();
		System.out.println("done...");

	}
}
```

# H O O K S

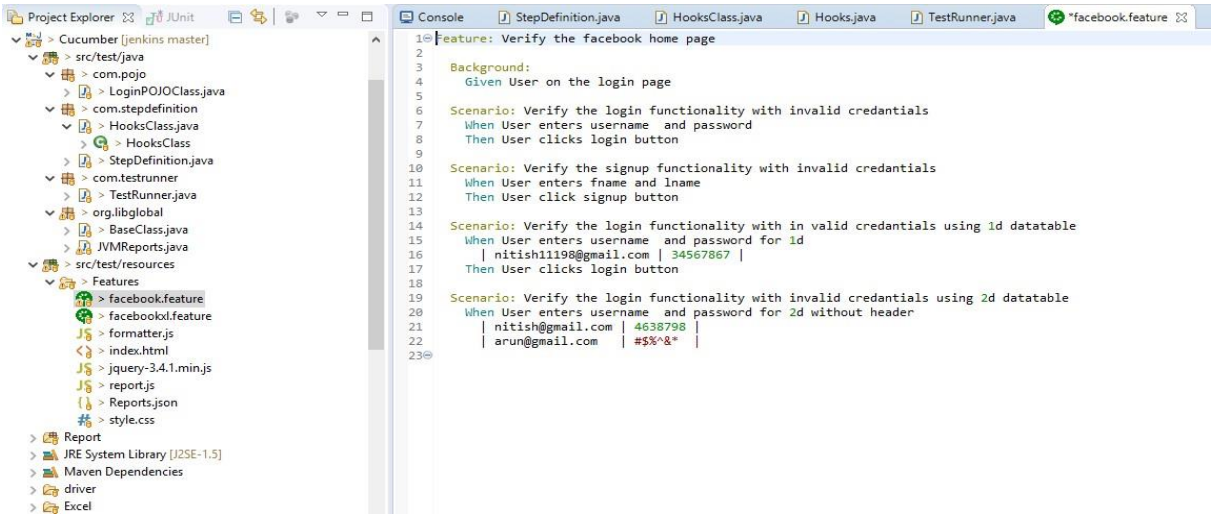
- Cucumber supports **hooks** which are blocks of code that run **before** or **after** each scenario.**Cucumber Hooks** allows us to better manage the code workflow and helps as to reduce the code redundancy.
- In hooks we will be having two anatations **@Before** and **@After**.
- In Hooks we can priortize the execution order.
- In Hooks we can use tags (taged hooks).

**@Before**

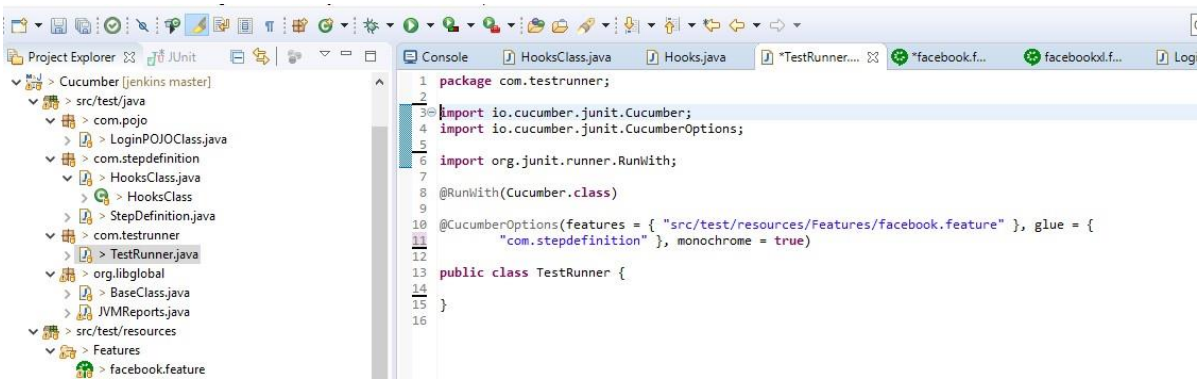
- Which is going to execute before each scenario like
    1. Launch browser.
    2. Load url.
    3. Delete cookies.
    4. Maximize window.
    5. Waits.

**@After**

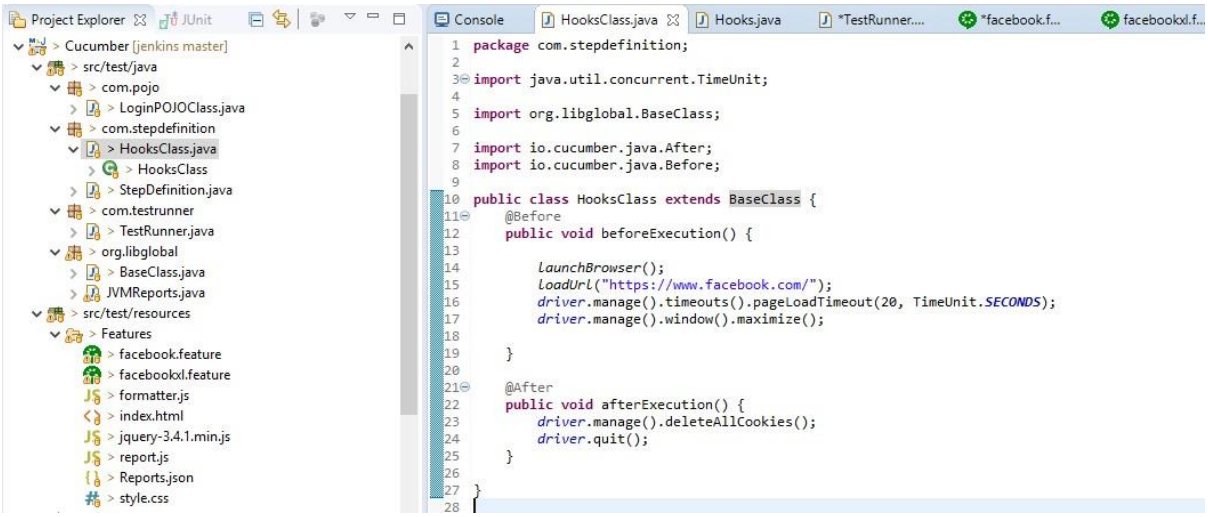
- Which is going to execute after each scenario like
    1. Close browser.
    2. Quit browser.
    3. Screenshot.
    4. Closing the database connection.




**Feature File One**




**TestRunner Class**




**HooksClass**

```
package org.libglobal;

import java.io.File;

import java.io.FileInputStream; import
java.io.IOException; import
java.text.SimpleDateFormat;import
java.util.Date;

import org.apache.commons.io.FileUtils; import
org.apache.poi.ss.usermodel.Cell; import
org.apache.poi.ss.usermodel.DateUtil;import
org.apache.poi.ss.usermodel.Row; import
org.apache.poi.ss.usermodel.Sheet; import
org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.openqa.selenium.OutputType; import
org.openqa.selenium.TakesScreenshot;import
org.openqa.selenium.WebDriver; import
org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;
public class BaseClass {

	public static WebDriver driver;

	public static void launchBrowser() {
		System.setProperty("webdriver.chrome.driver",

				"C:\\Users\\k_sur\\Desktop\\Nitish\\cucumber
jenkins\\jenkins\\driver\\chromedriver.exe");
		driver = new ChromeDriver();

	}
```

```
	}
	public static void loadUrl(String url) {

		driver.get(url);

	}
	public static void type(WebElement e, String value) {
		e.sendKeys(value);

	}
	public static void btnClick(WebElement e) {e.click();

	}
	public static String getTittle() { String title =
		driver.getTitle();return title;

	}
	public static String getUrl() {

		String Url = driver.getCurrentUrl();
		return Url;

	}
	public static void quitBrowser() {

		driver.quit();

	}
	public static void selectByVisibleText(WebElement element, String text) {

		new Select(element).selectByVisibleText(text);
	}
	public static String getTextAttribute(WebElement element) {String
		attribute = element.getAttribute("value"); return attribute;
	}
	@SuppressWarnings({ "deprecation", "resource" })
	public static String getDataFromExcel(int row, int cell) throws IOException {String value = null;

		File loc = new File("C:\\Users\\k_sur\\eclipse-
workspace\\Cucumber\\Excel\\integration.xlsx");
		FileInputStream stream = new FileInputStream(loc);
		Workbook w = new XSSFWorkbook(stream);
		Sheet s = w.getSheet("Sheet1");Row r
		= s.getRow(row);
		Cell c = r.getCell(cell);
```

```
                int type = c.getCellType();
                if (type == 1) {
                        value = c.getStringCellValue();
                } else if (type == 0) {

                        if (DateUtil.isCellDateFormatted(c)) {

                                Date dateCellValue = c.getDateCellValue(); SimpleDateFormat sim =
                                new SimpleDateFormat("dd-mm-yyyy");value =
                                sim.format(dateCellValue);
                        } else {

                                double numericCellValue = c.getNumericCellValue();
                                long l = (long) numericCellValue;value =
                                String.valueOf(l);

                        }
                }
                return value;

        }
}
```

**BaseClass**

```
package com.pojo;

import org.libglobal.BaseClass;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
public class LoginPOJOClass extends BaseClass {

        public LoginPOJOClass() { PageFactory.initElements(driver,
                this);

        }
        @FindBy(id = "email")
        private WebElement username;

        @FindBy(id = "pass")
        private WebElement password;

        @FindBy(id = "loginbutton")
        private WebElement login;

        @FindBy(id = "u_0_m")
        private WebElement fname;

        @FindBy(id = "u_0_o")
        private WebElement lname;

        @FindBy(id = "u_0_13")
        private WebElement signup;
```

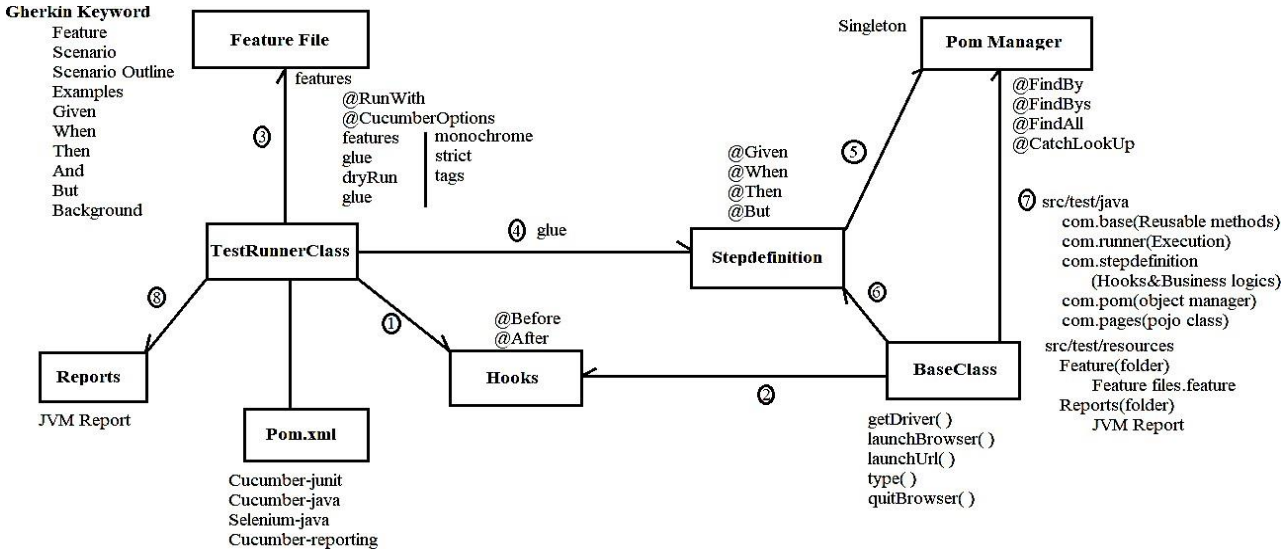
```
	public WebElement getUsername() {

		return username;

	}
	public WebElement getPassword() {

		return password;

	}
	public WebElement getLogin() {

		return login;

	}
	public WebElement getFname() {

		return fname;

	}
	public WebElement getLname() {

		return lname;

	}
	public WebElement getSignup() {

		return signup;
```

## LoginPojoClass

```
package com.stepdefinition;

import java.io.IOException;

import java.util.List;

import org.libglobal.BaseClass;

import com.pojo.LoginPOJOClass;

import io.cucumber.java.en.*;

public class StepDefinition extends BaseClass {LoginPOJOClass

	l;


	@Given("User on the login page")
	public void user_on_the_login_page() {

	}
	@When("User enters username   and password")
	public void user_enters_username_and_password() {l = new
		LoginPOJOClass();

		type(l.getUsername(), "nitish11198@gmail.com");
		type(l.getPassword(), "345678");
	}
	@Then("User clicks login button")
	public void user_clicks_login_button() {
```

```
		l = new LoginPOJOClass();
		btnClick(l.getLogin());

	}
	@When("User enters fname and lname")
	public void user_enters_fname_and_lname() {l = new
		LoginPOJOClass();

		type(l.getFname(), "nitish");
		type(l.getLname(), "kumar");
	}
	@Then("User click signup button")
	public void user_click_signup_button() {l = new
		LoginPOJOClass();

		btnClick(l.getSignup());

	}
	@When("User enters username   and password for {int}d")
	public void user_enters_username_and_password_for_d(Integer int1,
io.cucumber.datatable.DataTable d) {
		List<String> li = d.asList();l = new
		LoginPOJOClass();
		type(l.getUsername(), li.get(0));
		type(l.getPassword(), li.get(1));
	}
	@When("User enters username   and password for {int}d without header")
	public void user_enters_username_and_password_for_d_without_header(Integer int1,io.cucumber.datatable.DataTable d)
			{
		List<List<String>> list = d.asLists();l = new
		LoginPOJOClass();
		type(l.getUsername(), list.get(0).get(0));
		type(l.getPassword(), list.get(0).get(1));
	}
	@When("User enters the username and password")
	public void user_enters_the_username_and_password() throws IOException {l = new
		LoginPOJOClass();

		type(l.getUsername(), getDataFromExcel(1, 1));

		type(l.getPassword(), getDataFromExcel(1, 1));

	}
	@Then("User clicks the login button")
	public void user_clicks_the_login_button() {

		btnClick(l.getLogin());

	}
}
```

**StepDefinitionClass**




## Cucumber Architecture

**CUCUMBER TAGS**

- Using tags we can run the bulk of testcases.
- Using tags we can skip the bulk of testcase.

**Syntax**

- tags={"@tagname"}

**ignore**

- tags={"**~@tagname**"}

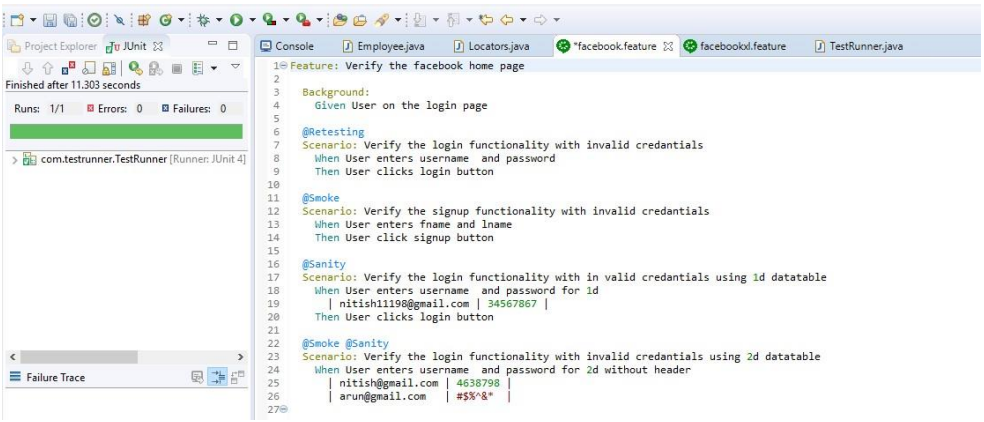
**groups using OR operator**

- using OR operator we can run more than one groups.
- Comma (, ) operator is used for OR.

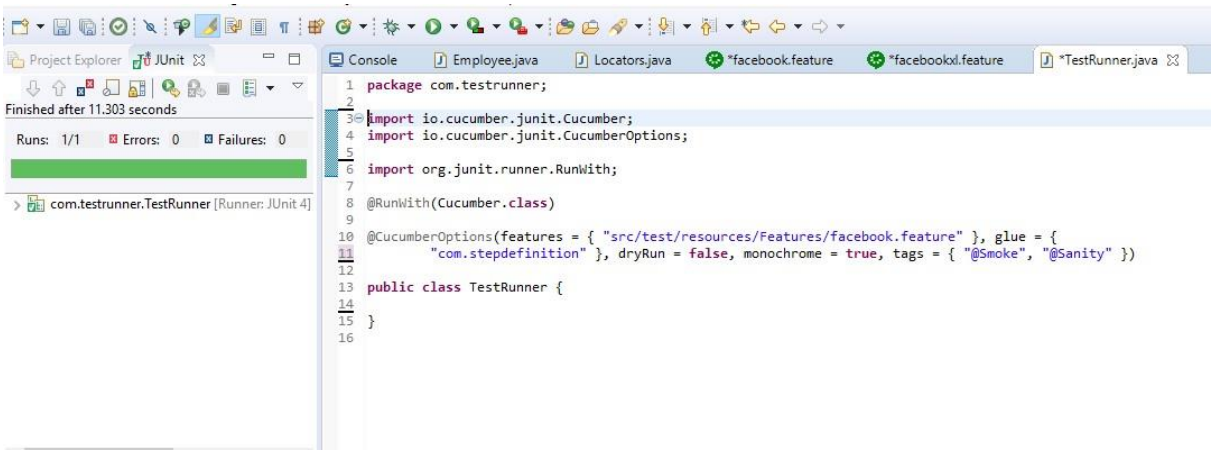
tags={"**@tagname1, @tagname2**"}

**groups using AND operator**

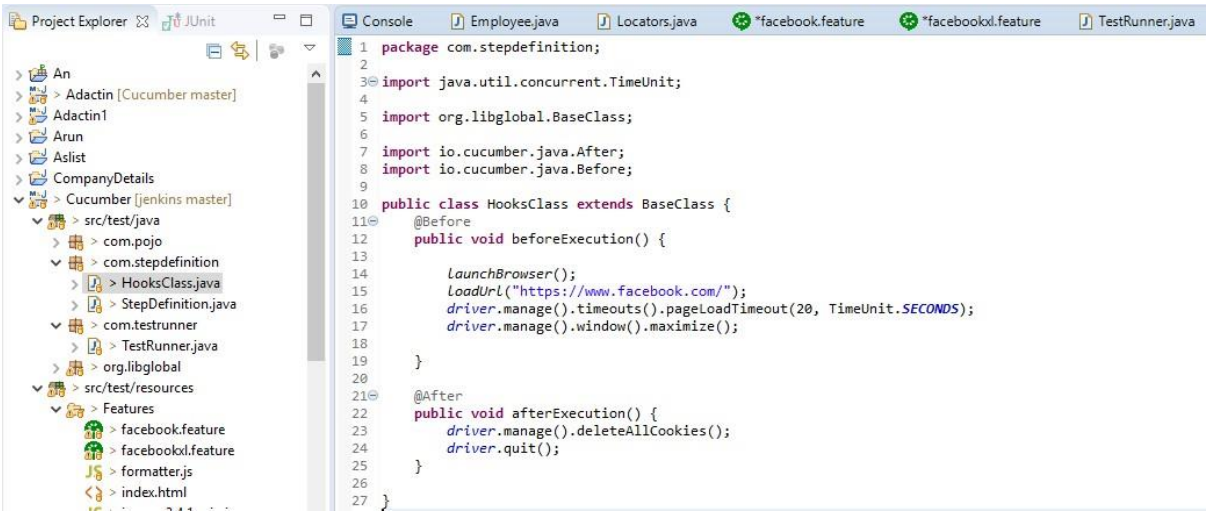
- Tags={"**@tagname1**","**@tagname2**"}




**Feature File one**




**TestRunner Class**




**Hooks Class**

```
package org.libglobal;

import java.io.File;

import java.io.FileInputStream; import
java.io.IOException; import
java.text.SimpleDateFormat;import
java.util.Date;

import org.apache.commons.io.FileUtils; import
org.apache.poi.ss.usermodel.Cell; import
org.apache.poi.ss.usermodel.DateUtil;import
org.apache.poi.ss.usermodel.Row; import
org.apache.poi.ss.usermodel.Sheet; import
org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.openqa.selenium.OutputType; import
org.openqa.selenium.TakesScreenshot;import
org.openqa.selenium.WebDriver; import
org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;
public class BaseClass {

	public static WebDriver driver;

	public static void launchBrowser() {
		System.setProperty("webdriver.chrome.driver",

					"C:\\Users\\k_sur\\Desktop\\Nitish\\cucumber
jenkins\\jenkins\\driver\\chromedriver.exe");
		driver = new ChromeDriver();

	}
	public static void maximizeWindow() {

		driver.manage().window().maximize();
```

```
	public static void loadUrl(String url) {

		driver.get(url);

	}
	public static void type(WebElement e, String value) {
		e.sendKeys(value);

	}
	public static void btnClick(WebElement e) {e.click();

	}
	public static String getTittle() { String title =
		driver.getTitle();return title;

	}
	public static String getUrl() {

		String Url = driver.getCurrentUrl();
		return Url;

	}
	public static void quitBrowser() {

		driver.quit();

	}
	public static void selectByVisibleText(WebElement element, String text) {

		new Select(element).selectByVisibleText(text);
	}
	public static String getTextAttribute(WebElement element) {String
		attribute = element.getAttribute("value"); return attribute;
	}
	@SuppressWarnings({ "deprecation", "resource" })
	public static String getDataFromExcel(int row, int cell) throws IOException {String value = null;

		File loc = new File("C:\\Users\\k_sur\\eclipse-
workspace\\Cucumber\\Excel\\integration.xlsx");
		FileInputStream stream = new FileInputStream(loc);
		Workbook w = new XSSFWorkbook(stream);
		Sheet s = w.getSheet("Sheet1");Row r
		= s.getRow(row);
		Cell c = r.getCell(cell);
		int type = c.getCellType();
```

```
                if (type == 1) {
                        value = c.getStringCellValue();
                } else if (type == 0) {

                        if (DateUtil.isCellDateFormatted(c)) {

                                Date dateCellValue = c.getDateCellValue(); SimpleDateFormat sim =
                                new SimpleDateFormat("dd-mm-yyyy");value =
                                sim.format(dateCellValue);
                        } else {

                                double numericCellValue = c.getNumericCellValue();
                                long l = (long) numericCellValue;value =
                                String.valueOf(l);

                        }
                }
                return value;

        }
}
```

**BaseClass**

```
package com.pojo;

import org.libglobal.BaseClass;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
public class LoginPOJOClass extends BaseClass {

        public LoginPOJOClass() { PageFactory.initElements(driver,
                this);

        }
        @FindBy(id = "email")
        private WebElement username;

        @FindBy(id = "pass")
        private WebElement password;

        @FindBy(id = "loginbutton")
        private WebElement login;

        @FindBy(id = "u_0_m")
        private WebElement fname;

        @FindBy(id = "u_0_o")
        private WebElement lname;

        @FindBy(id = "u_0_13")
        private WebElement signup;

        public WebElement getUsername() {

                return username;
```

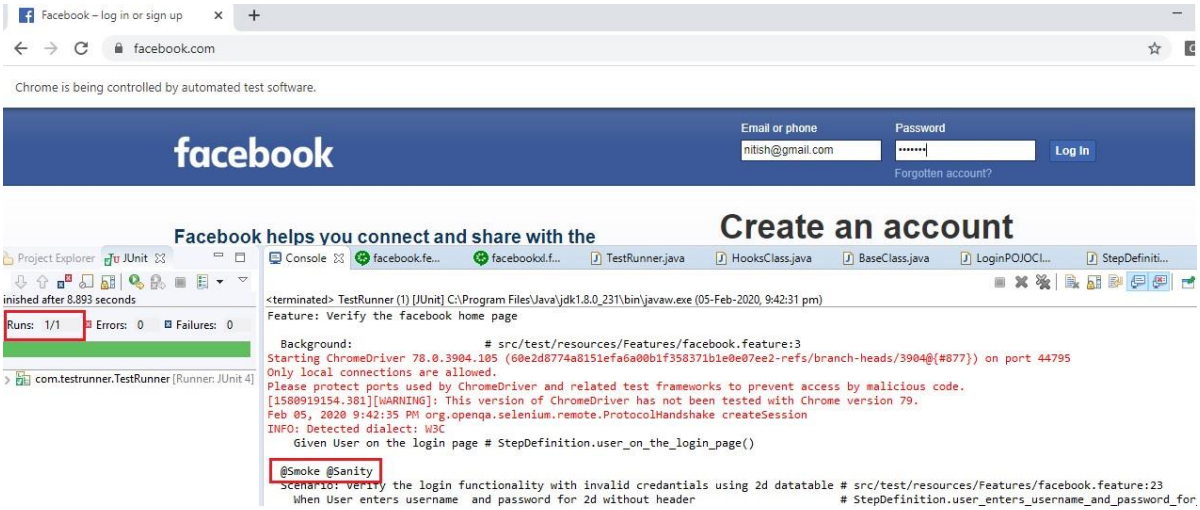
```
public WebElement getPassword() {

        return password;

}
public WebElement getLogin() {

        return login;

}
public WebElement getFname() {

        return fname;

}
public WebElement getLname() {

        return lname;

}
public WebElement getSignup() {

        return signup;
```

## Pojo Class

```
package com.stepdefinition;

import java.io.IOException;

import java.util.List;

import org.libglobal.BaseClass;

import com.pojo.LoginPOJOClass;

import io.cucumber.java.en.*;

public class StepDefinition extends BaseClass {LoginPOJOClass

    l;


    @Given("User on the login page")
    public void user_on_the_login_page() {

    }
    @When("User enters username   and password")
    public void user_enters_username_and_password() {l = new
            LoginPOJOClass();

            type(l.getUsername(), "nitish11198@gmail.com");
            type(l.getPassword(), "345678");
    }
    @Then("User clicks login button")
    public void user_clicks_login_button() {l = new
            LoginPOJOClass();

            btnClick(l.getLogin());
```

```
	}
	@When("User enters fname and lname")
	public void user_enters_fname_and_lname() {l = new
		LoginPOJOClass();

		type(l.getFname(), "nitish");
		type(l.getLname(), "kumar");
	}
	@Then("User click signup button")
	public void user_click_signup_button() {l = new
		LoginPOJOClass();

		btnClick(l.getSignup());

	}
	@When("User enters username   and password for {int}d")
	public void user_enters_username_and_password_for_d(Integer int1,
io.cucumber.datatable.DataTable d) {
		List<String> li = d.asList();l = new
		LoginPOJOClass();
		type(l.getUsername(), li.get(0));
		type(l.getPassword(), li.get(1));
	}
	@When("User enters username   and password for {int}d without header")
	public void user_enters_username_and_password_for_d_without_header(Integer int1,io.cucumber.datatable.DataTable
			d) {
		List<List<String>> list = d.asLists();l = new
		LoginPOJOClass();
		type(l.getUsername(), list.get(0).get(0));
		type(l.getPassword(), list.get(0).get(1));
	}
	@When("User enters the username and password")
	public void user_enters_the_username_and_password() throws IOException {l = new
		LoginPOJOClass();

		type(l.getUsername(), getDataFromExcel(1, 1));

		type(l.getPassword(), getDataFromExcel(1, 1));

	}
	@Then("User clicks the login button")
	public void user_clicks_the_login_button() {

		btnClick(l.getLogin());

	}
}
```

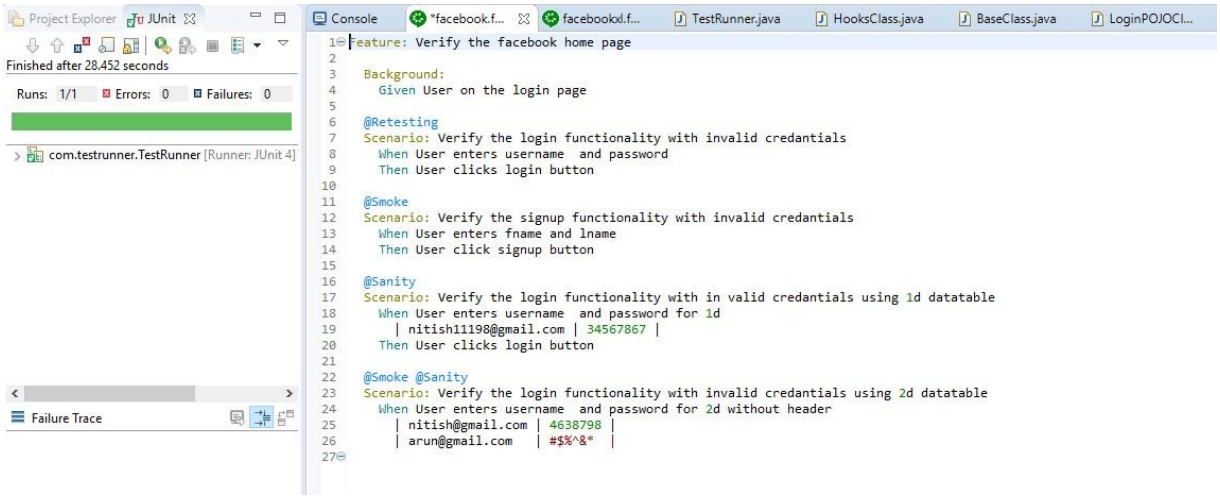
**Stepdfinition Class**



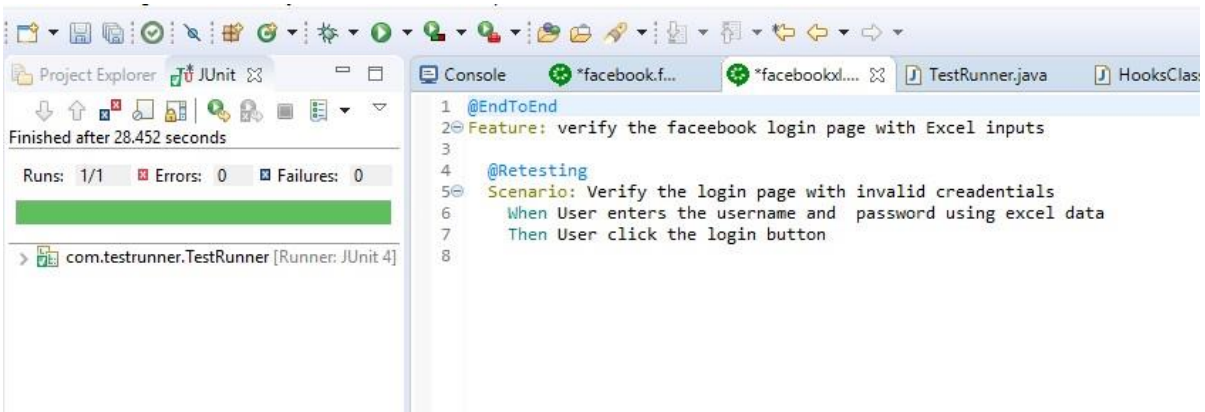

**Output**

**Feature File Grouping**

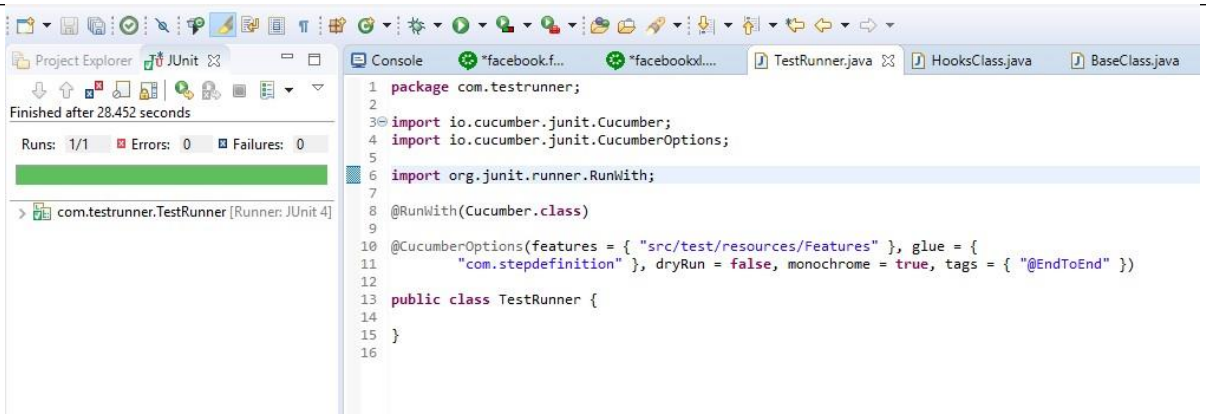
- tags={"**@EndToEnd**"}



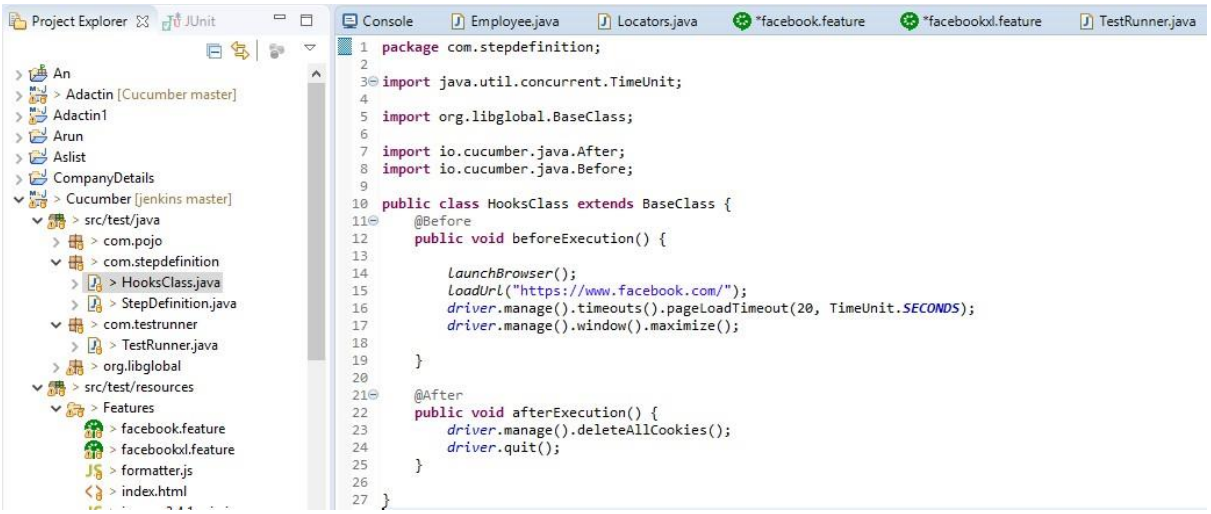

**FeatureFile One**




**Feature File Two**




**TestRunnerClass**




**Hooks Class**

```
package org.libglobal;

import java.io.File;

import java.io.FileInputStream; import
java.io.IOException; import
java.text.SimpleDateFormat;import
java.util.Date;

import org.apache.commons.io.FileUtils; import
org.apache.poi.ss.usermodel.Cell; import
org.apache.poi.ss.usermodel.DateUtil;import
org.apache.poi.ss.usermodel.Row; import
org.apache.poi.ss.usermodel.Sheet; import
org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.openqa.selenium.OutputType; import
org.openqa.selenium.TakesScreenshot;import
```

```
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;
public class BaseClass {

    public static WebDriver driver;

    public static void launchBrowser() {
        System.setProperty("webdriver.chrome.driver",
```

```
                                        "C:\\Users\\k_sur\\Desktop\\Nitish\\cucumber
jenkins\\jenkins\\driver\\chromedriver.exe");
                driver = new ChromeDriver();

        }
        public static void maximizeWindow() {

                driver.manage().window().maximize();
        }
        public static void loadUrl(String url) {

                driver.get(url);

        }
        public static void type(WebElement e, String value) {
                e.sendKeys(value);

        }
        public static void btnClick(WebElement e) {e.click();

        }
        public static String getTittle() { String title =
                driver.getTitle();return title;

        }
        public static String getUrl() {

                String Url = driver.getCurrentUrl();
                return Url;

        }
        public static void quitBrowser() {

                driver.quit();

        }
        public static void selectByVisibleText(WebElement element, String text) {

                new Select(element).selectByVisibleText(text);
        }
        public static String getTextAttribute(WebElement element) {String
                attribute = element.getAttribute("value");
```

```
                return attribute;

        }
        @SuppressWarnings({ "deprecation", "resource" })
        public static String getDataFromExcel(int row, int cell) throws IOException {String value = null;

                File loc = new File("C:\\Users\\k_sur\\eclipse-
workspace\\Cucumber\\Excel\\integration.xlsx");
                FileInputStream stream = new FileInputStream(loc);
                Workbook w = new XSSFWorkbook(stream);
                Sheet s = w.getSheet("Sheet1");Row r
                = s.getRow(row);
                Cell c = r.getCell(cell);
                int type = c.getCellType();
                if (type == 1) {
                        value = c.getStringCellValue();
                } else if (type == 0) {

                        if (DateUtil.isCellDateFormatted(c)) {

                                Date dateCellValue = c.getDateCellValue(); SimpleDateFormat sim =
                                new SimpleDateFormat("dd-mm-yyyy");value =
                                sim.format(dateCellValue);
                        } else {



                }
                return value;

      double numericCellValue = c.getNumericCellValue();
      long l = (long) numericCellValue;value = String.valueOf(l);


        }
}
```

**BaseClass**

```
package com.pojo;

import org.libglobal.BaseClass;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
public class LoginPOJOClass extends BaseClass {

        public LoginPOJOClass() { PageFactory.initElements(driver,
                this);

        }
        @FindBy(id = "email")
        private WebElement username;

        @FindBy(id = "pass")
        private WebElement password;

        @FindBy(id = "loginbutton")
        private WebElement login;

        @FindBy(id = "u_0_m")
        private WebElement fname;
```

```
	@FindBy(id = "u_0_o")
	private WebElement lname;

	@FindBy(id = "u_0_13")
	private WebElement signup;

	public WebElement getUsername() {

		return username;

	}
	public WebElement getPassword() {

		return password;

	}
	public WebElement getLogin() {

		return login;

	}
	public WebElement getFname() {

		return fname;

	}
	public WebElement getLname() {

		return lname;

	}
	public WebElement getSignup() {

		return signup;

	}
}
```

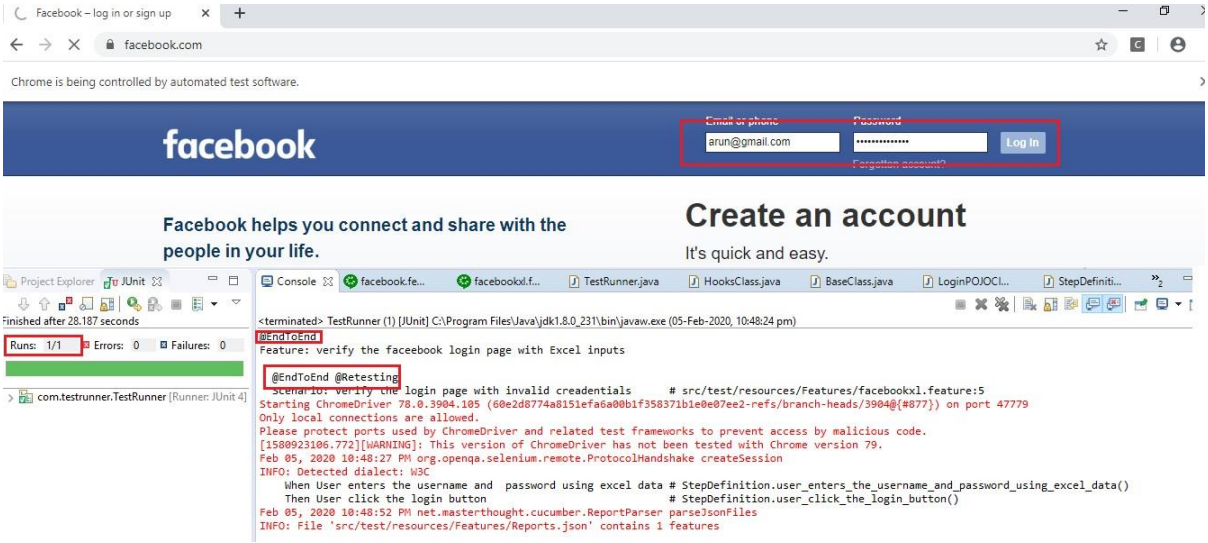
**Pojo Class**

```
package com.stepdefinition;

import java.io.IOException;

import java.util.List;

import org.libglobal.BaseClass;

import com.pojo.LoginPOJOClass;

import io.cucumber.java.en.*;

public class StepDefinition extends BaseClass {LoginPOJOClass

	l;
```

```
	public void user_on_the_login_page() {

	}
	@When("User enters username   and password")
	public void user_enters_username_and_password() {l = new
		LoginPOJOClass();

		type(l.getUsername(), "nitish11198@gmail.com");
		type(l.getPassword(), "345678");
	}
	@Then("User clicks login button")
	public void user_clicks_login_button() {l = new
		LoginPOJOClass();

		btnClick(l.getLogin());

	}
	@When("User enters fname and lname")
	public void user_enters_fname_and_lname() {l = new
		LoginPOJOClass();

		type(l.getFname(), "nitish");
		type(l.getLname(), "kumar");
	}
	@Then("User click signup button")
	public void user_click_signup_button() {l = new
		LoginPOJOClass();

		btnClick(l.getSignup());

	}
	@When("User enters username   and password for {int}d")
	public void user_enters_username_and_password_for_d(Integer int1,
io.cucumber.datatable.DataTable d) {
		List<String> li = d.asList();l = new
		LoginPOJOClass();
		type(l.getUsername(), li.get(0));
		type(l.getPassword(), li.get(1));
	}
	@When("User enters username   and password for {int}d without header")
	public void user_enters_username_and_password_for_d_without_header(Integer int1,io.cucumber.datatable.DataTable
			d) {
		List<List<String>> list = d.asLists();l = new
		LoginPOJOClass();
		type(l.getUsername(), list.get(0).get(0));
		type(l.getPassword(), list.get(0).get(1));
	}
	@When("User enters the username and password")
```

```
	public void user_enters_the_username_and_password() throws IOException {l = new
		LoginPOJOClass();

		type(l.getUsername(), getDataFromExcel(1, 1));

		type(l.getPassword(), getDataFromExcel(1, 1));

	}
	@Then("User clicks the login button")
	public void user_clicks_the_login_button() {

		btnClick(l.getLogin());

	}
	@When("User enters the username and   password using excel data")
	public void user_enters_the_username_and_password_using_excel_data() throws IOException {l = new
		LoginPOJOClass();
		type(l.getUsername(), getDataFromExcel(1, 1));

		type(l.getPassword(), getDataFromExcel(1, 1));

	}
	@Then("User click the login button")
	public void user_click_the_login_button() {l = new
		LoginPOJOClass(); btnClick(l.getLogin());

	}
```

**Stepdfinition Class**




**Output**

# G I T

Version Control System (VCS)

==========

Versioning

Tracking

Cloning

Two Types

(i) Distributed Version Control System(DVCS)

(ii) Centralized Version Control System(CVCS)

Git and Github

svn

## Git

====

how to integrate git in our System

Download git for Windows ----Install

Verification ----Commandprompt ---- git --version

How to Transfer files from Local disk to Remote Repository

---

Step 1: Go to the Project folder ---Right Click -----Click git Bash here

Step 2: Initialize a Empty git local repository along with Staging area

In git Bash here

git init

Step 3: To move the files from Local disk to staging area

git add .

Step 4: To check the current status

git status

if no commits yet is displayed -- Files are in Staging area

UnTracked Files -----Files are in Local Disk

Step 5: To move the files from Staging area to Local Repository

git commit -m "message"

Step 6: To Check the current status

git status

if Working Tree is clean ---- Files are in local Repository

Files to be committed ----Files are in Staging Area

Step 7: To move the files from Local Repository to Remote Repository

a) Create a folder or Repository in Git hub

b) Mention the Repository in Git Bash here

git remote add origin master url

c) To push the files from local repository to remote repository

git push -u origin master

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

how to get the project from Remote repository to local disk

==========

Step 1: Create a folder in local disk

Step 2: Create a Branch in git hub

Step 3: Copy the url

Step 4: In local disk inside the newly created folder open git Bash here

git clone url

Step 5: To get the project into Eclipse

Open Eclipse---File-----import----Existing maven Project(Maven Project)

import ---Existing java Project

Step 6: Switch the Branch From master to Current Branch

git checkout branchnmae

git status---To know the current Branch

Step 7: Work on your module

Step 8: Push the work from your local disk to remote repository to your branch

git add .

git commit -m "message"

git status

git push -u origin branchnme

Compare & Pull request---Compare (Code in Branch is not in master)

pull request ( We have to raise a request to accept our code from Branch to Master)

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

how to configure our local repository to remote repository

==========

git config --global user.email "xyz@gmail.com"

git config --global user.name "xyz"

how to resolve the conflicts (GIT Conflicts)

==========

Step 1: Get the updated from the master

git pull origin master

Now in Eclipse the code worked by different branch or different resource will be displayed

head -- Latest work

tail --- previous work

Step 2: Resolve conflicts by deleting unwanted code and keep the verified code

Now push the back to the branch and create a pull request

The code will be merged

git log --- to know all commits

git branch-- to know all branch

# M A N U A L  T E S T I N G

**Why we go for Manual Testing?**

1. Human eye captures more defects.
2. Everything cannot be automated like Captcha, Image, Animation, Video, etc.
3. Report prepared by the manual testers will be always detailed because manual testers are strongin

   Domain knowledge. While automation testers are strong in technical knowledge.

**Software Testing**:

- Functional Testing
- Non-Functional Testing

1. Functional Testing:

Checking the functionalities of the application.

- Manual Testing (Testing the functionalities by manual)
- Automation Testing (Testing the functionalities by using tools/scripts)
- Webservices Testing (Validating request and response) - API Testing
  **Automation Tools**:
  - Selenium - Web Based Applications
  - Appium - Mobile Applications
  - QTP (Quick Test Professional)/UFT (Unified Functional Testing) - Desktop Related

    Applications [Ex: MS Office Applications, Web Browsers, etc.]

  - Test Complete - Both Web Based and Desktop Based Applications

2. Non-Functional Testing:

Checking the non-functional aspects of the application.

**Types of Non-Functional Testing**:

1. Performance Testing:
   - They add virtual number of users accessing the application simultaneously and at a same timeto check

     the performance of the application.

*Tools Used*:

- J-Meter Open-Source Tool
- Load Runner Paid Version

*Load/Stress Testing*:

- Process of adding load and making the application stress is called Load/Stress Testing.

2. Usability Testing:

- To check how good the application is user friendly.

3. Accessibility Testing:
   - To check how good the application is accessible to different users.
   - [Note: This testing mainly focus on physically challenged peoples]
   - Tools Used: JAWS and NVDA

**Principles of Software Testing**:

1. Testing shows the presence of defect not their absence.
2. Exhaustive testing is not possible (Testing all the combinations of valid or invalid data is impossible).
3. Early testing saves time and money.
4. Defects cluster together (Concentrating the module even it is small which has more defects and testing it thoroughly will raise of more defect). It was found that 80% ofdefects comes from 20% of the modules.
5. Beware of pesticide paradox (Aware of which module the testing is required. Don'tSpend time in testing the module which doesn't have any defects).
6. Testing is context dependent (Whatever the application we are going to test, we have toact based on the context as what the application related to).
7. Absence of error is a fallacy (Instead of telling bug free, make sure the environments are matching with the client requirement).

**SDLC Overview**:

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software.

**Phases of SDLC**:

1. Requirement gathering and analysis
2. Design
3. Development
4. Testing >>STLC
5. Deployment
6. Operation and Maintenance

**Software Development Life Cycle**:

1. Requirement gathering and analysis:
   - A Contract will be signed between client and marketing team.
   - Business Analyst (BA) will try to get the requirement from the client and prepare a documentcalled

- Business Requirement Document (BRD) or Software Requirements Specification (SRS)

  Document

2. Design:
   - Then a document will be prepared called Design Specification Document, which will givedetails

     about how the front and back end could interact.

3. Development: (Dev Environment)
   - Then Developers will develop the software by the use of any coding language and they willdo a

     test called Unit Testing.

4. Testing: (QA or Testing Environment)
   - Once developer coded for the functionality of the software then they will move their code or build

     into testing environment called Testing and once developer moved the code into testingstage, tester

     will test the applications as per the flow STLC.

5. Deployment: (Live Environment)
   - Once a program has passed the testing phase, it is ready for deployment.
   - Deploy the application in the live environment.
   - Typically, it happens at Non-Peak Hours.
6. Operation and Maintenance:
   - Maintenance of software can include software upgrades, repairs, and fixes of the software if itbreaks.

**STLC Overview**:

- Software Testing Life Cycle (STLC) is defined as a sequence of activities conducted toperform

  Software Testing.

- Each of these stages have a definite Entry and Exit criteria;

*Entry Criteria*:

Entry Criteria gives the prerequisite items that must be completed before testing can begin.

*Exit Criteria*:

Exit Criteria defines the items that must be completed before testing can be concluded.

**Phases of STLC**:

- Requirement Analysis
- Test Planning
- Test Design
- Test execution
- Sign Off

1. Requirement Analysis:

Entry Criteria: Contracts, BR Document

- Understanding the requirements
- Clarification of doubts in query logs through walkthrough session
- Identify types of tests to be performed
- Get application access

Exit Criteria: Finalized scope.

2. Test Planning:

Entry Criteria: Finalized scope

- Preparation of test plan document for various types of testing
- Test tool selection
- Resource planning

Exit Criteria: Test Plan.

3. Test Design:

Entry Criteria: Test Plan

- Create test scenarios, test cases, automation script, RTM(Requiremnet Tracability Matrix)
- Create test data
- Review test cases with peer or lead
- Get Approval

Exit Criteria: Test Scenarios, Test Case, Test Data, RTM.

4. Test Execution:

Entry Criteria: Test Scenarios, Test Case, Test Data, RTM

- Execute tests as per plan
- Document test results, and log defects for failed cases
- Track the defects to closure
- Retest the Defect fixes

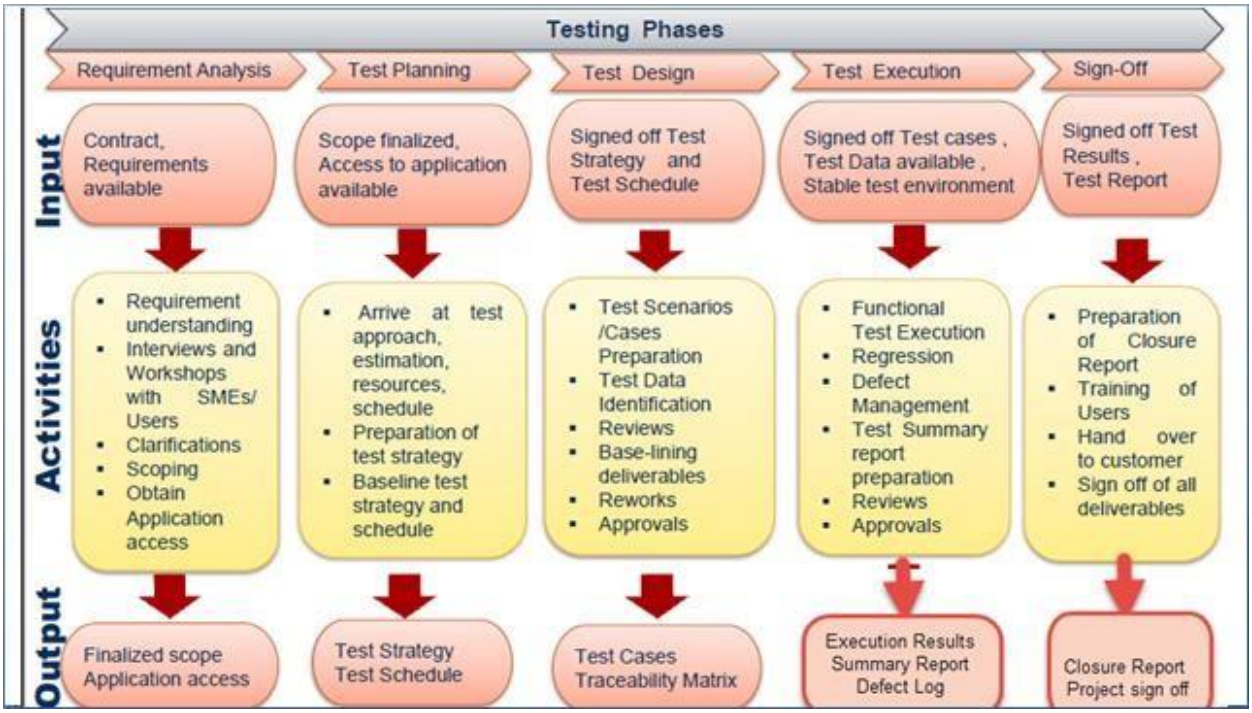
Exit Criteria: Execution Result, Defect Log.

5. Sign Off:

Entry Criteria: Execution Result, Defect Log

- Evaluate cycle completion
- Prepare Test closure report
- Test result analysis

Exit Criteria: Closure Document.




**Types of Testing**:

*Test Scenario*:

It is high level Testcase. It is an idea of what we are going to test.

*Test Case*:

It is an idea of how we are going to test, it includes test case (tc) name, test data, expected & actual results, status, etc.

*Test Data*:

The actual input which we are going to use in the application for testing.

*Defect*:

When the expected and actual is not matching, then it is a defect. In other words, when the application does not conform to the requirement specification.

*Bug*:

When the tester raises a defect to developer then it is called as bug.

**Different Types of Testing:**

*White Box Testing*:

White Box Testing is also called as Glass Box, Clear Box, and Structural Testing. It is based on applications internal coding knowledge. It is done by developers.

*Black Box Testing*:

Black Box Testing is a software testing method in which testers evaluate the functionality of the software under test without looking at the internal coding knowledge. It is done by testers.

*Positive Testing*:

It is to determine what system supposed to do. It helps to check whether the application is justifying the requirements or not.

*Negative Testing*:

It is to determine what system not supposed to do. It helps to find the defects from the software.

*Smoke Testing*:

Smoke Testing is done to make sure if the build we received from the development team is testable or not.It is also called as “Day 0” check. It is done at the “build level”. It helps not to waste the testing time to simply testing the whole application when the key features don’t work.

*Regression Testing*:

It is done to make sure the existing functionalities of the application is not impacted whenever any new functionalities are added to the application.

*Sanity Testing*:

Sanity Testing is done during the release phase to check for the main functionalities of the application without going deeper. It is also called as a subset of Regression testing. It is done at the “release level”. At times due to release time constraints rigorous regression testing can’t be done to the build, sanity testing does that part by checking main functionalities.

*Formal Testing*:

It is a process where the testers test the application by having pre-planned procedures and proper documentation.

*Informal Testing*:

It is a process where the testers test the application without having any pre-planned procedures and proper documentation.

*Monkey Testing*:

Perform abnormal action on the application deliberately in order to verify the stability of the application.

*Retesting*:

Once the developer fix the bug and we need to test again whether the bug is fixed or not.

*Beta Testing*:

Beta testing is done by a limited number of end users before delivery. Usually, it is done in the client place.

*Live Environment Testing*:

It is nothing but testing the application in live environment i.e., After Release.

| Test Data | Step No. | Test Step Description | Expected Result | Actual Result | Status | Created by |
|---|---|---|---|---|---|---|
| Username: any valid username Pwd: any valid pwd | Step 1 | Open the GC Broswer | GC Broswer should be opened | As Expected or As-Is | Pass | Merin |
| | Step 2 | Enter the Amzon URL and Clcik on Search | Amazon URL should be launched | As Expected or As-Is | Pass | |
| | Step 3 | Click the login link | Login page should be displayed | Login page is not displayed | Failed | |
| | Step 4 | Enter the Valid username | Username should be displayed | | Blocked | |
| | Step 5 | Enter the valid password | Password should be displayed as masked | | Blocked | |
| | Step 6 | Click on the Login button | Home Page should be displayed | | Blocked | |

**Testing Techniques**:

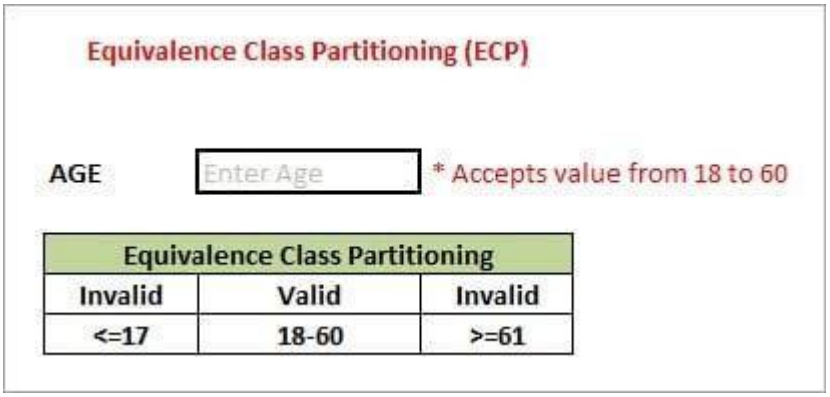
1. Equivalence case partitioning Technique
2. Decision Table Technique
3. State Transition Technique
4. Boundary Value Analysis
5. Error Guessing Technique
6. Ad-hoc Testing

**Equivalence partitioning Technique**:

- In equivalence partitioning, the test cases are equally divided based upon positive and negativeinputs.

Example: Consider that we have to select an age between 18-56,

- Here the Valid inputs are 18-56 and the invalid inputs are <=17 and =>57.
- Here we have one valid and two invalid inputs.




**Decision Table Technique**:

- In Decision table technique, we deal with combinations of inputs.
- To identify the test cases with decision table, we consider conditions and actions.
- We take conditions as inputs and actions as outputs.

Example: Login validation - Allow user to login only when both the username and password is correct.

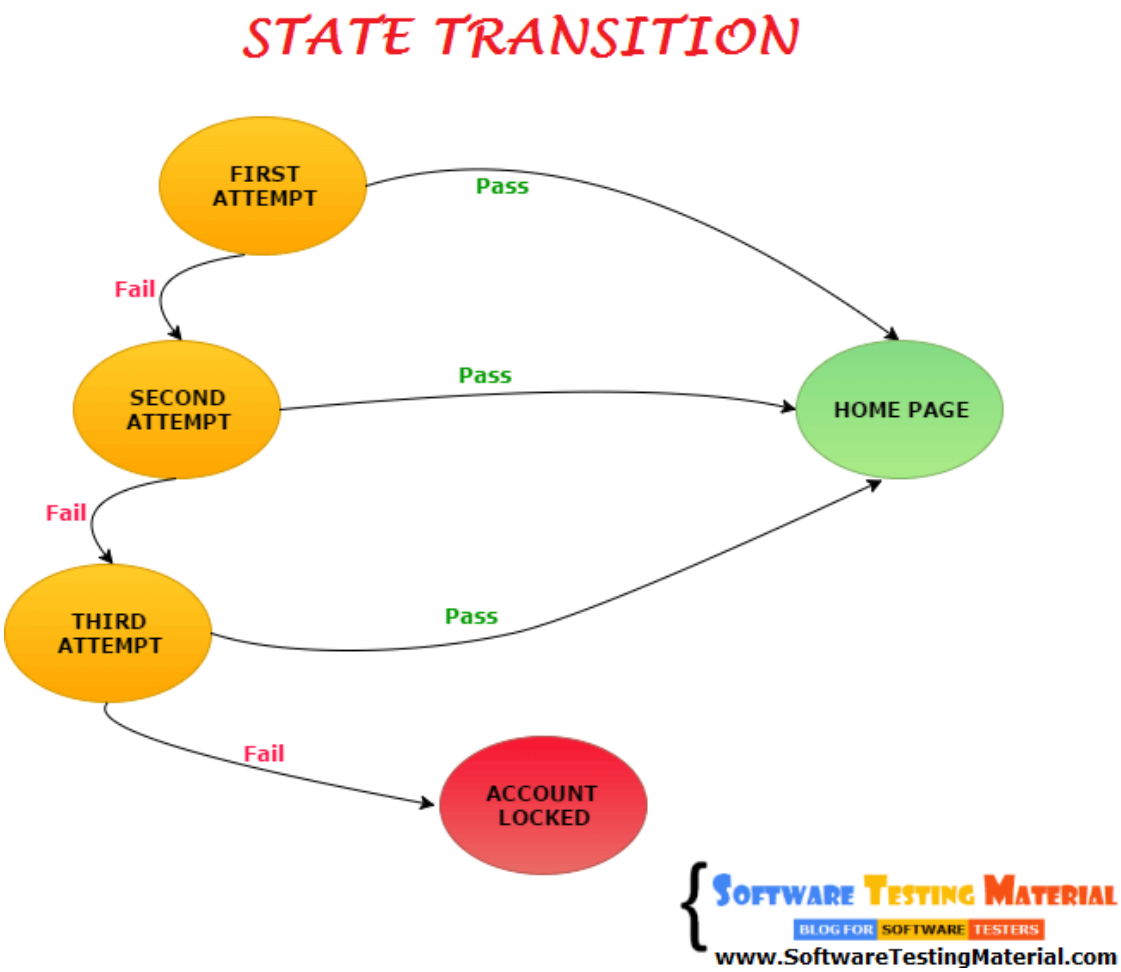
- Conditions 1: Enter valid username and valid password and Click Login.
  - Actions 1: Display home page and Execute.
- Conditions 2: Enter invalid username and valid password and Click Login.
  - Actions 2: Display Error message as invalid username.

| | | | | |
|---|---|---|---|---|
| **Email (condition1)** | T | T | F | F |
| **Password (condition2)** | T | F | T | F |
| **Expected Result (Action)** | Account Page | Incorrect password | Incorrect email | Incorrect email |

**State Transition Technique**:

- Using state transition testing, we pick test cases from an application where we need to test different system transitions.
- We can apply this when an application gives a different output for the same input,depending onwhat has happened in the earlier state.

Example: Login with invalid username and password three times keeps the account page blocked until change password.




**Boundary Value Analysis**:

- Boundary value analysis is based on testing the boundary values of valid and invalid partitions.
- Every partition has its maximum and minimum values and these maximum and minimum valuesare the boundary values of a partition.

Example: If we want to enter an amount between 100 to 1000.

Here we check based on boundaries for 100, we take 98, 99, 101, 102 and for 1000, we take 998, 999, 1001, 1002.

**Error Guessing Technique**:

- Error Guessing is used to find bugs in software application based on testers prior experience.
- Here we won't follow any specific rules.

**Ad-hoc Testing**:

- Ad-hoc testing is quite opposite to the formal testing.
- It is an informal testing type.
- In Ad-hoc testing, testers randomly test the application without following any documents and testdesign techniques.
- This testing is primarily performed if the knowledge of testers in the application under test is veryhigh.

- Testers randomly test the application without any test cases or any business requirement

  document.

**Defect Life Cycle**:

*New*:

When the defect is found and posted for the first time.
*Assigned*:

Once the defect is posted(bug), the testing team lead will assign it to Developing Team.
*Open*:

Once the developer opens the bug and works on to fix it.
*Fixed*:

When developer fix the bug with appropriate codes.
*Pending Retest*:

Once the developer fixes the bug, it will be moved to testing team.
*Retest*:

When the tester retesting whether the bug raised was fixed or not.
*Verified*:

If the tester confirms that the bug is fixed.
*Reopen*:

If the tester confirms that the bug is not yet fixed and found the same defect. Again, it will be moved to

Assigned Stage and the flow continuous.

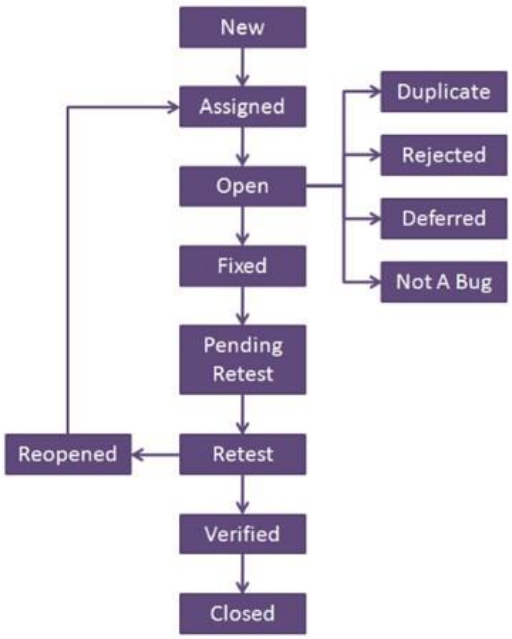
*Closed*:

Once the tester verified that the bug is fixed, then the defect raised will be closed.
*Duplicate*:

If the bug is already raised by someone, and if it is raised again means it is duplicate.
*Rejected/Not a Bug*:

When the developer feels that the bug is not genuine.
*Deferred*:

The bug which developer feels that it can be fixed later on the upcoming sprints.


**Levels of Testing**:

The testing done at various levels and by whom it is done is called Levels of Testing.

There are 4 levels of testing. If these 4 levels of testing are completed only, we can say the application is ready for release.

- Unit Testing
- Integration Testing
- System Testing
- Acceptance Testing

1. Unit Testing:
   - Testing an individual module in an application
   - Done by Developers
   - Test based on White Box Testing Technique.
2. Integration Testing:
   - Combining or merging two modules
   - Done by Testers
   - Test based on Black Box Testing Techniques
     - Big Bang Approach - Combing all combination of units
     - Top-down Approach - Higher level combinations of units
     - Bottom-up Approach - Lower level combinations of units
3. System Testing:
   - Complete Application Testing
   - Done by Testers
   - Test based on Black Box Testing Technique

**Types of System Testing**:

- Functional Testing - To test the functionalities of the application
- Usability Testing - To test whether the application is user friendly and it is easily understandable.
- Performance Testing - To test whether the application is not getting crashed or down when thereis load/stress
- Security Testing - To test whether the application is secure or not

If theses 4 levels of testing are completed only, we can say system testing is completed

4. Acceptance Testing:

- Getting approval from client
- Done by client side
- Testing will be done at Development/Testing Environment
- Satisfying client requirement.

**SDLC Methodologies**:

- Water Fall Model
- V-Model of Testing
- Iterative Model
- Agile Model

1. Waterfall Model:
   - Simple and easy to understand and use.
   - Each phase has specific deliverables and review processes.
   - Documentation and artifacts maintained properly.
   - Suitable for projects where requirements are well understood.

*Disadvantages*:

- Not suitable for projects where requirements are at a risk of changing.
- Cost of fixing defects is very high when detected at a later stage.
- Not a good model for complex and long projects.

2. V-Model:
   - The V-model is an SDLC model where execution of processes happens in a sequentialmanner in a V-shape.
   - It is also known as Verification and Validation model.
3. Iterative Model:
   - We are going to iterate each functionality in to each part and design, develop and test anddeploy.
4. Agile Model:
   - AGILE methodology is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project. Here both developmentand testing activities are concurrent unlike the Waterfall model.

*Why we go for Agile?*

- More Control
- Better Productivity
- Better Quality

- Higher Customer Satisfaction
- Higher return on investment

*Agile Manifesto (Principles)*:

- Individual and team interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
    - Responding to change over following a plan

[Note: Framework we are following in Agile is Scrum.]

**Agile**:

1. Roles
2. Artifacts/Documents
3. Ceremonies/Meeting

**Members/Role Players in Agile**:

*Product Owner*:

- Product Owner defines features of the product and decides release date.
- Product Owner prioritize the features according to the market value and profitability of the product.
- Product Owner can accept or reject work item result.

*Scrum Master*:

- Scrum Master manages the team and look after the team's productivity.
- Scrum Master is responsible for setting up the team, scrum meeting invite and removes obstaclesto progress.

*Scrum Team/Dev Team*:

- The team is usually about 5-9 members.
- The whole team involved in development and testing.
- Team manages its own work and organizes the work to complete the sprint or cycle.
- And attend the scrum meetings

**Different Artifacts used in Agile**:

*User story*:

They are short explanation of functionalities of the system under test.

*Acceptance Criteria*:

They are detailed explanation of functionalities of the system under test which is going to shows what we have to do.

*Product Backlog*:

- It is a collection of user stories captured for a scrum product.
- The product owner prepares and maintains the product backlog. It is prioritized by product owner.

*Sprint*:

It is a set period of time to complete the user stories, decided by product owner, usually 2-3 weeks of time.

*Sprint Backlog*:

It's a set of user stories to be completed in a sprint from the product backlog.

*Burnup and Burndown Chart*:

The burn-up chart illustrates the amount of completed work in a project whereas the burn-down chart depicts the amount of work remained to complete a project. Thus, the burn-up and burn-down charts are used to trace the progress of a project.

- Qtest - Test Management Tool
- JIRA - Defect Management Tool and Test Management Tool

*Agile Ceremonies or Meetings*:

1. Sprint Grooming:

- To understand the user stories.
- Product Owner will be explaining the user stories.
- Meeting Duration - 1 hour.
- Attended by - Product Owner, Scrum Master and Scrum Team.

2. Sprint Planning:

- Here the product owner will have already prioritized product backlog.
- Work is selected from the Product Backlog and pulled into the Sprint Backlog.
- Complexity of the user stories will be estimated based on Poker Card Technique.
- Meeting Duration - 1 to 3 hours.
- Attended by - Scrum Master and Scrum Team.

Before Sprint

3. Daily Scrum Meeting (or) Daily Stand-up calls:

- Stand-up is designed to quickly inform everyone of what's going on across the team.
- It's not a detailed status meeting. Following will be discussed in that meeting
- What did I complete yesterday?
- What will I work on today?
- Am I blocked by anything?
- Meeting Duration - 15 minutes.
- Attended by - Scrum Master and Scrum Team.

After Sprint

4. Sprint Review Meeting:

- After completion of every sprint, Team will give the showcase to the product owner.

- Scrum team will show the demo of app with current sprint's completed stories.
- Meeting Duration - 1 hour.
- Attended by - Product Owner, Scrum Master and Scrum Team.

5. Sprint Retrospective Meeting:

- Retrospectives help the team to understand what worked well–and what didn't.
- Team will rate the sprint out of 10 and discuss the below
- What went right?
- What went wrong?
- What can be done better for next sprint?
- Meeting Duration - 1 hour.
- Attended by - Scrum Master and Scrum Team.

**Defect Management Tool/Test Management Tool: JIRA**

- JIRA is a tool developed by Australian Company Atlassian.
- It is used for bug tracking, issue tracking, and project management.

*How to open an account in JIRA*:

- Type “JIRA account create” in Google,
- Click on “Try - Jira Software | Atlassian”,
- Click Try Free on Jira Software
- Create an account
- Select Scrum
- Enter Project Name

*How to Create a Bug*:

- Click Create or + Button
- Select Bug in Issue Type
- Give Summary - Defect Name
- Give Detailed explanation of the defect in Description
    - Environment: Testing Env
    - URL: www.aircanada.com/dev/46768596968
    - Steps to reproduce: 1.
      2.
      3.
    - Actual results:
    - Expected results:
    - Priority:
    - Severity:
    - Screenshots:
    - Assignee:

**Priority and Severity**:

*Severity: (Testers)*

Explaining how severe the defect is going to impact the application.

*Priority: (Developers or PO)*

How soon the defect is going to be fixed

- High Severity, High Priority - Login, Search, Payment, etc.
- High Severity, Low Priority - Links, websites, etc.
- Low Severity, Low Priority - Content validation

- Low Severity, High Priority - Logo, colors

**PROJECT DOMAIN**

**Travel Domain**:

- Car: Ola, Uber
- Bus: Redbus, Makemytrip
- Train: irctc
- Air: Air India, Air Canada

**Airline Domain**:

Select any airline which should not be familiar among us (eg: European and African Airlines)
*Modules*:

- Booking Module: Explain the complete booking flow to book a flight ticket
- Check In: It is a process that a particular airline accepts the passenger prior to travel
    - It is done at airport and online airline authority to enter the passenger to board their flight
    - Boarding Pass: Pass given by airline
- Flight Status: To check the status of the particular flight
- Login: Credentials given by the airline to the frequent customers and agents
    - Guest
    - Frequent Flyer
    - Agent
- Baggage: Maintaining the passenger's luggage
    - Carryon Baggage - Carry inside flights
    - Checkin Baggage - Other Bags based upon weights
- Upgrade: Passenger can travel from his lower class to upper class

**Booking Module**:

1. Home Page:

- Trips
    - Round Trip - Chennai to Madurai to Chennai
    - One Way - Chennai to Madurai
    - Multicity - Chennai to Madurai to Trichy to Bangalore to Mumbai
- Passenger
    - Adult (16-18+)
    - Youth (12-16 or 12-18)
    - Child (2-11)
    - Infant (0-2)
- Source and Destination
- Travel Date
- Promocode - Offer given by the airline for the particular routes or flights

2. Flight Select Result Page:

- Flight Name
- Travel Duration - Departure and Arrival Time
- Flight Details - Direct or connecting flight
- Cabin Selection
    - Economy
    - Premium Economy

- Business

- Fare Details

3. Passenger Information Page:

- Complete passenger details with first name, last name, gender, DOB, Mobile No, Email Addressand

  Frequent Flyer Program

4. Seat Selection Page:

- Available Seats
  - Regular Seats - Not payable
  - Preferred Seats - Payable for Extra Legroom, etc.

- Occupied Seats

5. Payment Page

- Different Types of payment to be made
  - Online Banking
  - Card payment

  - App Payment

6. Booking Confirmation Page:

- PNR (Passenger Name Record) / Booking Reference Number
- Seat Number
- Ticket in downloadable format

7. Manage My Booking Page:

- Cancel Booking by entering Last Name and PNR
- Change Seats
- Change Flight
- Change Cabin

**Ecommerce Domain**:

Online Purchasing Websites

E.g.:

- Amazon
- Flipkart
- Snapdeal

*Ecommerce Domain*:

Select any online purchasing websites which should not be familiar among us.

**Modules**:

- Purchasing Module - End to End purchasing a product
- Login Module - Credentials given by the website
- Cart Module - Managing our purchasing
- Product List Module - List of products available in the website

**Purchasing Module**:

1. Home Page:

- Hero Image

- Search Product
    - By providing product name
    - By clicking on product list
    - By clicking the offers available
    - By clicking on our preferred products available

2. Search Result Page:

- List of searched products available
- Product Details with different brands
- Image
- Price
- Offers
- Review
- Filters
- Multiple pages available

3. Product Details Page:

- Detailed information of the selected product
- Offers available for the selected product
- Additional Images of the selected product
- Review of the selected product
- Delivery Options
- Shipping Information
- In Stock/Out of Stock
- Add to Cart

4. Shopping Cart Page:

- Selected product is added
- No of products
- Total amount with delivery charges and discounts
- Add a product
- Remove a product
- Shipping amount with different shipping options

5. Payment Page

- Different Payment Options
- Purchase by Guest or Login
- If no login, give sign up
- Storing card details
- Session timeout if signed up for a long time
- Email/SMS confirmation

6. My Orders:

- Change the order
- Cancel the order
- Track the order
- Returns

**Banking Domain**:

- Process involved in Banks
- Select any of the country bank which should not be familiar among us except Asia.
- In banking there several business operations like
    - Core Baking system

- ATM Banking
- Internet Banking System
- Asset liability management system

**Modules**:

- Admin (Login)
- Personal Banking
- Corporate Banking

**Personal Banking**:

- Login Page
- Balance Enquiry
- Open Deposits
- Apply for Loans
- Apply Credit Card
- Pay Credit Card Bill
- Bill Payments
- Transaction Status

**Insurance Domain**:

- Process involved in Insurance Companies
- Select any of the country insurance which should not be familiar among us except Asia.

**Modules**:

- Login: Member, Employer, Doctors/Providers
- Plan Coverage
- Claims

**Plan Coverage**:

- Plans
- Coverage
- Find a Doctor
- Other Services

**Claims**:

- How to Submit a claim
- View Claims and statements

**Other Important Questions**:

1. Insprint in Agile:

- To develop and automate user stories in the same sprint, is called in-sprint automation.
- If there are 4 User stories in a sprint which could be automated then, We will write a skeletonscripts for the User stories which we are going to automate it in this sprint.
- In a 2 weeks sprint, for 6 days developer will Develop the application and on 7th day the build is delivered to Automation team.
- After that we will execute the scripts.

2. Outsprint or n-1 Sprint:

- To automate user stories for the previous sprint, is called out-sprint automation.
- Say there were 4 User stories from previous Sprint (Sprint1) which could be automated.
- I will automate it in Sprint 2, which is called as n-1 or outsprint automation.
- In my previous Sprint i.e., Sprint 1, I will be working on the framework setup.

3. Velocity:

Total Number of user story points consider in a particular sprint (We can say 40 Points)

4. Capacity:

- Total no of working days*Total no of resources*Total number of working hours(PD) = Capacity
    - Capacity= 10*7*8

5. Risk Based Testing:

   Identifying the functionalities which seems to cause failures to the application and then testing those functionalities.

6. Bug Leakage:

   Bug which is missed by the testing team while testing and if it is found by the end user or customer in Live Environment.

7. Bug Release:

   Releasing the application to the Live Environment with the known bugs then we call it as Bug Release.

8. Defect Age:

   It is the time interval between date of defect detection and date of defect closure.

- Defect Age = Date of defect closure – Date of defect detection
- For Example, A defect was found and reported on 17 May 2020 and it was fixed on 22 May2020. So the defect age is 5 days.

9. HotFix:

   A bug which needs to handle as more than high priority bug and needs to be fix it immediately.

10. RTM:

    Requirements Traceability Matrix (RTM) is used to trace the requirements are done by the testing side by matching the requirement with test cases.

11. Test Suite:

    Test Suite is a collection of test cases.

12. Test Strategy:

- It is a high-level document which captures the approach on how we go about testing the productand achieve the goals.
- It is developed by project manager.
- Documents like Test Plan are prepared by keeping this document as a base.

13. Test Environment:

It is the combination of hardware and software on which Test Team performs testing.

**Important Questions**:

1. 7 principles of software testing?
2. Draw and explain SDLC?
3. Draw and explain STLC?
4. What is the difference between severity and priority?
5. What you mean by test plan, test scenario, test case?
6. What you mean by RTM?
7. Draw and explain defect life cycle?
8. What is difference between functional and non-functional testing?
9. What you mean by usability testing?
10. What is mean by accessibility testing?
11. What are the different automation tools we have in market?
12. What is defect management tool used in your project?
13. Draw and explain water fall model?
14. Difference between regression and re testing?
15. Difference between black box and white box testing?
16. Explain equivalence case partitioning techniques with an example?
17. Explain Decision table with an example?
18. Explain state transition technique with an example?
19. Explain boundary value analysis with an example?
20. Explain error guessing?
21. Difference between formal and informal testing?
22. Explain Ad-hoc testing?
23. What is positive and negative testing?
24. What is test strategy?
25. What is test suite?
26. What is test environment?
27. What is test data?
28. What is test closure (or) sign off?
29. What is unit testing?
30. What is integration testing?
31. What is system testing?
32. What is beta testing?
33. What is smoke testing?
34. What is sanity testing?
35. What is monkey testing?
36. What is security testing?
37. What is performance testing?
38. What is walk- through?
39. What is bug leakage?
40. What is bug release?
41. What is Defect Age?
42. Give an example all the four categories of priority and severity?

**Others**

- What is the last user story/functionality you worked on your project?
- Write 5 defects which you raised in your project?

1. What is the current sprint you are working?
2. What is the total number of sprints so far in your project?
3. How many test cases we can execute in a day?
4. How many test cases we can prepare in a day?

5. How many automations test scripts we can execute in a day?
6. How many automations test scripts we can prepare in a day?

# O V E R A L L  D E P E N D E N C I E S

| Dependencies | Version |
|---|---|
| Selenium Java | 3.141.59 |
| WebDriverManager | 5.0.3 |
| Cucumber Java | 4.2.0 |
| Cucumber Junit | 4.2.0 |
| Cucumber Reporting | 5.6.1 |
| Apache Poi ooxml | 3.8-beta4 (Data Driven Framework) |
| TestNG | 6.14.3 |

```
<dependencies>
	<!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
	<dependency>
		<groupId>io.github.bonigarcia</groupId>
		<artifactId>webdrivermanager</artifactId>
		<version>5.0.3</version>
	</dependency>

	<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
	<dependency>
		<groupId>org.seleniumhq.selenium</groupId>
		<artifactId>selenium-java</artifactId>
		<version>3.141.59</version>
	</dependency>

	<!-- https://mvnrepository.com/artifact/io.cucumber/cucumber-java -->
	<dependency>
		<groupId>io.cucumber</groupId>
```

```
            <artifactId>cucumber-java</artifactId>
            <version>4.2.0</version>
        </dependency>

        <!-- https://mvnrepository.com/artifact/io.cucumber/cucumber-junit -->
        <dependency>
            <groupId>io.cucumber</groupId>
            <artifactId>cucumber-junit</artifactId>
            <version>4.2.0</version>
            <scope>test</scope>
        </dependency>

        <!-- https://mvnrepository.com/artifact/net.masterthought/cucumber-reporting -->
        <dependency>
            <groupId>net.masterthought</groupId>
            <artifactId>cucumber-reporting</artifactId>
            <version>5.6.1</version>
        </dependency>

        <!-- https://mvnrepository.com/artifact/org.apache.poi/poi-ooxml -->
        <dependency>
            <groupId>org.apache.poi</groupId>
            <artifactId>poi-ooxml</artifactId>
            <version>3.8-beta4</version>
        </dependency>

</dependencies>
```

# O V E R A L L  S E L E N I U M  T O P I C S

## Xpath:

---

- Xpath is an XML path which is used to find locators on the webpage using DOM Structure.
- There are two types of Xpath: Absolute which is denoted by single slash and Relative Xpath which is denoted by double slash.
- In that in our project we are using Relative Xpath because we can directly find the element
- anywhere at the webpage but in Absolute Xpath we have to find from the head of the DOM structure.

## Actions:

---

Actions is a predefined class which is used for mouse over actions in a webpage. There are few methods as

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

| Methods | Used For | Syntax |
|---|---|---|
| moveToElement() | used to do mouse over actions | action.moveToElement(element).click().perform(); |
| doubleClick() used for double click | used for Double Click | refname.doubleClick(webelement).perform(); |
| contextClick() used for right click | used for Right Click | actions.contextClick(btnElement).perform(); |
| dragAndDrop() | Used to move a web element from one place to another | refname.dragAndDrop(from, to).perform(); |
| keyUp() | Used for key release | keyDown(txtUsername, Keys.SHIFT) |
| keyDown() | Used for key press | keyUp(txtUsername, Keys.SHIFT) |

## Robot:

---

Robot is a predefined class present in java.awt package which is used for performing keyboard actions in a webpage. There are few methods as

**Robot robot = new Robot();**

| Methods | Used For | Syntax |
|---|---|---|
| keyPress() | used for Key Press | robot.keyPress(KeyEvent.VK_SHIFT); |
| keyRelease() | used for Key Release | robot.keyRelease(KeyEvent.VK_SHIFT); |

# Select:

---

Select is a predefined class which is used to perform dropdown in a webpage. There are few methods as

**Select refname = new Select();**

| Methods | Used For | Syntax |
|---|---|---|
| isMultiple() | To verify whether dropdown is multiselected will returns true in Boolean value | refname.isMultiple(); |
| selectByIndex() | To select an option in dropdown by using index | refname.selectByIndex(1); |
| selectByValue() | To select an option in dropdown by using value | refname.selectByValue(“text”); |
| selectByVisibleText() | To select an option in dropdown by using visible text | refname.selectByVisibleText("Banana"); |
| getAllSelectedOptions() | To get the selected options in dropdown | getAllSelectedOptions() : List<WebElement> |
| getFirstSelectedOptions() | To get the first selected option in dropdown | Refname.getFirstSelectedOptions(String value) |
| deSelectByValue() | To deselect an option in dropdown by using index | Refname.deselectByValue(String value) |
| deSelectByIndex() | To deselect an option in dropdown by using index | refname.deselectByIndex(int index) |
| deSelectByVisibleText() | To deselect an option in dropdown by using index | Refname.deselectByVisibleText(String value) |
| deSelectAll() | To deselect all the selected option in | refname.deselectAll(); |

| | | |
|---|---|---|
| | dropdown | |
| getOptions() | To get all options present in dropdown using list (List <WebElement>) | List <WebElement> elementCount = oSelect.getOptions(); |

# JavaScript Executor:

---

If locator is found in DOM Structure but still it shows noSuchElement Exception due to hidden elements in the webpage,

if sendkeys not working, if click not working and for Scrollup and Scrolldown, we go for JavaScript Executor. It is an Interface. (**Typecasting**)

**JavascriptExecutor js = (JavascriptExecutor) driver;**

**js.executeScript(Script,Arguments);**

There are few methods as

- executeScript()
- ("arguements[0].setAttribute('value')",WebelementRef) sendkeys
- ("arguements[0].getAttribute()") get the particular value
- ("arguements[0].click()",WebelementRef) click
- ("arguements[0].scrollIntoView(true)",WebelementRef) scroll down
- ("arguements[0].scrollIntoView(false)",WebelementRef) scroll up

# TakeScreenShot:

---

It is an Interface. It is used for capturing the webpage. This captured image will defaultly stored in temporary location.

We can store the captured image in our preferred location using CopyFile() Method present in FileUtils Class.

Here we have method called getScreenShotAs().

**TakesScreenshot scrShot =(TakesScreenshot).webdriver;**

Syntax: File SrcFile=scrShot.getScreenshotAs(OutputType.FILE);

## WebTable:

---

All the tables present in webpage is called webtable. First we need to fetch the rows using tagname "tr" and iterate the rows and

then fetch the data using tagname "td" and then iterate the datas and pick a particular data from the webtable using if conditions and

then fetch the data using tagname "th" and then iterate the headers.

There are two types Static - Data's are fixed.

Dynamic - Data's keeps changing

Dynamic Table Code

```
public List<String> getApplicantName(String status){

List<WebElement>
applicantName=driver.findElemtns(By.xpath("//table//td//span[text()='"+status+"']//ancestor::tr//td[2]"
));

List<String>name=new ArrayList<>();
for(int i=0;i<applicantName.size();i++){

String text=applicantName.get(i).getText();

name.add(text);

}
return name;
}
```

## Windows Handling:

---

Consider a webpage has multiple windows example 10 windows, if any actions performed in 2nd

window it will throw noSuchElement Exception

because the control will be still in the parent window (i.e)1st window. Now if you want to switch to the second window, windows Handling is used.

Method Used: driver.switchToWindow();

We can switch window by using String Id, String URL, String Title and There are few methods as

getWindowHandle() to get parent window ID

getWindowHandles() to get all windows ID which returns Set<WebElements> because Set does not allows duplicate.

```
String prtWindow=driver.getWindowHandle();
Set<String>allWindow=driver.getWindowHandles();
int count=0;
for(String window:allWindow){
if(count==3){

driver.switchTo().window(window);
}
count++;
}
```

(or)

```
List<String> li=new ArrayList<>();
li.addAll(allWindow);

driver.switchTo().window(li.get(2));
```

# Alert:

---

Alert is a kind of popup window or popup message. We can't find locators for Alert, so to handle an alert we use a method called

driver.switchToAlert(); If we are not handling alert, we can't do any operations.

Alert has 3 types

**Syntax**

Simple Alert accept();

**driver.switchTo().alert().accept();**

Confirm Alert accept(); dismiss();

**driver.switchTo().alert().dismiss();**

Prompt Alert sendKeys(); accept(); dismiss();

**driver.switchTo().alert().getText();**

**driver.switchTo().alert().sendKeys("Text");**

# Frames:

---

It is webpage embedded inside a webpage.

If any locators are placed inside a particular frame then we need to switch in to that particular frame and need to access the locators.

It is mainly for security purpose. We can switch to particular frame using ID, Name, Index and WebElement Reference.

Methods Used: driver.switchToFrame();

- driver.switchTo().frame(0);
- driver.switchTo().frame("iframe1");

parentFrame(); switch to previous frame

defaultContent(); switch to parent window

```
for(int i=0; i<=size; i++){
            driver.switchTo().frame(i);
            int total=driver.findElements(By.xpath("html/body/a/img")).size();
            System.out.println(total);
               driver.switchTo().defaultContent();}
```

# **Waits**:

---

If locator is found but still noSuchElement Exception is thrown due to webpage loading and wait problems, we go for Waits concept.

Two types Static Wait Will wait for the maximum time given though the locator is found. Example: Thread.Sleep

Dynamic Wait Will not wait for the maximum time, if locator is found it will navigate to next step. There are two types here

Implicit wait Given common for all the locators

Explicit wait Given only for a particular locator. There are two types here

WebDriver wait Given only in terms of seconds

Fluent wait Given in all time formats and handle time out exception.

**Syntax**

**Implicit Wait**:

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

**Explicit Wait**:

WebDriverWait wait=new WebDriverWait(driver, 20);

guru99seleniumlink= wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath( "/html/body/div[1]/section/div[2]/div/div[1]/div/div[1]/div/div/div/div[2]/div[2]/div/div/div/div/div[1] /div/div/a/i")));

**Fluent Wait**:

Wait wait = **new** FluentWait(driver).~~withTimeout~~(5, TimeUnit.***MILLISECONDS***).~~pollingEvery~~(60, TimeUnit.***SECONDS***).ignoring(Exception.**class**);