OOPS Concepts definitions:

==========

1.CLASS:

Class is the collection of objects and methods.

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

OBJECT:

Object is an instance of a class.

Using object,we can able to access the methods in a class.

It is a runtime memory allocation.

Object creation syntax:

Classname objectname=new Classname();

METHODS:

Set of actions to be performed.

Reusable lines of code can be kept inside a method.

We can access the method using object whenever needed.

How to access class properities using object:

object.variable;

object.method();

for eg:-

---

get(), getTitle(), getCurrentUrl(), getText(), getAttribute()

2.ENCAPSULATION:

-->Wrapping up of data and code acting on data together in a single unit.

-->Encapsulation, is to make sure that "sensitive" data is hidden from users.

-->To achieve this, you must: declare class variables/attributes as private

-->Provide public getters and setters methods to access and update the value of a private variable.

I am using encapsulation in real time in POJO class where I create all my locator variables as private. Because private variables cannot be accessed outside the class. If we want to access private variables we have to use getters.

3.POLYMORPHISM:

Performing single action in different ways.

Method Overloading/Static Binding/CompileTime Polymorphism:

--> In same class,there are multiple functions with same name but different parameters then these functions are said to be overloaded.

--> Same method overloaded again and again using different arguments type,count and order.

Program:

```
----------
package org.test;

public class Student {
	public void stuDetails() {
		System.out.println("Student details");
	}
	public void stuDetails(int stuId) {
		System.out.println("Student id: "+stuId);
	}
	public void stuDetails(String stuName, int stuId) {
		System.out.println("Student name: "+stuName);
		System.out.println("Student id: "+stuId);
	}
	public void stuDetails(int StuId, String stuName) {
		System.out.println("Student id: " + StuId);
		System.out.println("Student name: "+stuName);
	}
	public static void main(String[] args) {
		Student s=new Student();
		s.stuDetails();
		s.stuDetails(101);
		s.stuDetails(102, "Java");
		s.stuDetails("Java", 103);
	}
}
```

Output:

```
Student details
Student id: 101
Student id: 102
Student name: Java
Student name: Java
Student id: 103
```

For example:In realtime I m overloading println,sendkeys and frame methods

System.out.println("Renu");

System.out.println(123);

System.out.println(12.2);

webelement.senkeys("Renu"); //Here Sendkeys accepts only one argument

Actions a=new actions(driver);

a.sendkeys(webelement,"Renu"); //using actions class also we can use sendkeys,here i am passing two arguments

driver.switchTo.frame(String id);

driver.switchTo.frame(String name);

driver.switchTo.frame(int index);

driver.switchTo.frame(Webelement); //For same methods i m passing String type,int type and Webelement type

Method overriding/Dynamic Binding/RunTime Polymorphism:

-->Method Overriding is when one of the methods in the super class is redefined in the sub-class.

(or)When I am not satisfied with the business logic of my super class i am overriding that method in my subclass.

In Realtime I m overriding many methods such as retry() method from IRetryAnalyser, transform() method from IAnnotationTransformer and getMessage() method from Exception.

Notes:

-->A constructor cannot be overridden.

-->Final - declared methods cannot be overridden

-->Any method that is static cannot be used to override.

Child class:

```
public class Child extends Parent{
	public void print() {
		System.out.println("Child class method");
	}
	public static void main(String[] args) {
		Child c=new Child();
		c.print();
	}
}
```

Parent class:

```
public class Parent {
	public void print() {
	System.out.println("Parent class method");    //overrided method
	}
}
```

4.INHERITANCE:

Using inheritance we can access one class properties in another class without multiple object creation.so we can save memory.Also we can acheive code reusability.

In real time, I m using inheritance concept in base class where i will maintain all reusable methods.so,by extending base class i can call reusable methods wherever i want.

4a)SINGLE INHERITANCE:

One child class extends one parent class.

4b)MULTILEVEL INHERITANCE:

One child class extends more than one parent class in tree level structure.

4c)MULTIPLE INHERITANCE:

More than one parent class parallely supporting into one child class.We can't acheive multiple inheritance in java due to

1.Syntax error/compilation error(After extends keyword we cannot specify more than one class)

2.Priority problem(When parent classes having same method name with same arguments)

extending into same child class,

4d)HYBRID INHERITANCE:

It is the combination of single and multiple inheritance.

4e)HIERARCHIAL INHERITACE:

More than one child class inheriting the same parent class.

5. DATA ABSTRACTION:

==========

Hiding the implementation of the program.

(or)

Process of hiding certain details and showing only essential information to the user.

Abstraction can be achieved with either abstract classes or interfaces.

The abstract keyword is a non-access modifier, used for classes and methods:

1.Abstract class:

Contains abstract as well as non-abstract methods.

We can't create objects for abstract class(to access the methods in abstract class, it must be inherited in another class).

Abstract method: can only be used in an abstract class, and it does not have a body(business logic). The body is provided by the subclass which extends the abstract class.

Non-abstract method(Concrete method):normal method with business logic.

In real time, I m using 'By' in my project which is an abstract class.

```
public abstract class Sample {
      public abstract void clientId();  //abstract method which has no logic
      public void clientName() {
           System.out.println("Client name is CTS");
                                 //non-abstract method with business logic
      }
}
```

Why And When To Use Abstract Classes and Methods?

To achieve security - hide certain details and only show the important details of an object.

2.Java Interface:

Another way to achieve abstraction in Java, is with interfaces.

-->An interface can have only abstract methods.

-->To access the interface methods, the interface must be "implemented" (kind of "inherited") by another class with the implements keyword (instead of "extends").

-->The body of the interface method is provided in the class which implements the Interface.

```
public interface Test{
void add();   //All the methods inside interface will have 'public abstract' by default,so no need to mention.
}
```

I am using many interfaces in my project such as Webdriver, webelement, Alert, Wait, Javascript executor, Takesscreenshot, List, Set, Map etc

Notes on Interfaces:

-->Like abstract classes, interfaces cannot be used to create objects

-->Interface methods do not have a body - the body is provided by the "implement" class

-->On implementation of an interface, you must override all of its methods

-->Interface methods are by default abstract and public

-->Interface attributes(variables) are by default public, static and final

-->An interface cannot contain a constructor (as it cannot be used to create objects)

Why And When To Use Interfaces?

1) To achieve security - hide certain details and only show the important details of an object (interface).

2) Java does not support "multiple inheritance" (a class can only inherit from one superclass). However, it can be achieved with interfaces, class can implement multiple interfaces.

Advantages of Abstraction:

It reduces the complexity of viewing the things.

Avoids code duplication and increases reusability.

Achieving Multiple inheritance using interface:

1.we will not get syntax error because after "implements" keyword we can specify any number of interfaces.

2.There will be no priority problem (Though all the interfaces have same method, the definition for the particular method

will be present only in the class which implements all the interfaces--->so there will be one method definition)

```
package org.cts.test;

public class Company implements Int1,Int2{

	@Override

	public void intId() {

		System.out.println(123);

	}

	public void intName() {

	System.out.println("Multiple Inheritance achieved through interface");

	}

	public static void main(String[] args) {

		Company c=new Company();

		c.intName();

		c.intId();

	}

}
```

Difference between Abstraction and Encapsulation:

Encapsulation is data hiding(information hiding) while Abstraction is detail hiding(implementation hiding)

While encapsulation groups together data and methods that act upon the data, data abstraction deals with exposing the interface to the user and hiding the details of implementation.

STRING:

Strings are collection of characters enclosed within " " . In Java, String is a Class,strings are treated as objects.

String Literal:

String str = “Java”;

Literal string shares the same memory incase of duplicates.

It is stored in String pool constants(String pool constants present inside heap memory)

When String declared like this, intern() method on String is called. This method references internal pool of string objects. If there already exists a string value “Java”, then str will get reference of that string and no new String object will be created.

String Object(Non-Literal String)

Non-literal string will not share the same memory even if it is a duplicate.

Since we use "new" operator everytime it creates new memory.

Non-literal strings stored in heap memory.

String str = new String(“Java”);

In this,JVM is forced to create a new string reference, even if “Java” is in the reference pool.

Performance Comparison between String literal and String constant:

If we compare performance of string literal and string object, string object will always take more time to execute than string literal because it will construct a new string every time it is executed.

```
Program:
----------
package org.test;
public class Variable {
	public static void main(String[] args) {
		String s = "Java";
		String s1 = "Java";
		System.out.println("Literal String 's': " + s);
		System.out.println("Literal String 's' address: " + System.identityHashCode(s));
		System.out.println("Literal String 's1': " + s1);
		System.out.println("Literal String 's1' address: " + System.identityHashCode(s1));
		System.out.println();
		String s2 = new String("Java");
		String s3 = new String("Java");
		System.out.println("Non-literal String 's2': " + s2);
		System.out.println("Non-literal String 's2' address: " + System.identityHashCode(s2));
		System.out.println("Non-literal String 's3': " + s3);
		System.out.println("Non-literal String 's3' address: " + System.identityHashCode(s3));
		}
}
Output:
Literal String 's': Java
Literal String 's' address: 2018699554
Literal String 's1': Java
Literal String 's1' address: 2018699554
Non-literal String 's2': Java
Non-literal String 's2' address: 1311053135
Non-literal String 's3': Java
Non-literal String 's3' address: 118352462
```

IMMUTABLE STRING:

Immutable string once string created,value can't be changed in reference.

Eg:String literals

MUTABLE STRING:

For mutable string,we can change the value in reference.

Eg:StringBuffer,StringBuilder

StringBuffer->Synchronised(One thread can only access StringBuffer class at a time),Thread safe

StringBuilder---->Asynchronised(Multiple threads can access StringBuilder class at a time),Not thread safe.

Program:

```
package org.test;
public class Variable {
	public static void main(String[] args) {
		String s = "Hello";
		String s1 = "world";
		System.out.println("IMMUTABLE STRING");
		System.out.println("Immutable String 's' address: " + System.identityHashCode(s));
		System.out.println("Before concatenation 's': "+s);
		System.out.println("Immutable String 's1' address: " + System.identityHashCode(s1));
		System.out.println("Before concatenation 's1': "+s1);
		String co = s.concat(s1);
		System.out.println("Immutable String 'co' address: " + System.identityHashCode(co));
		System.out.println("After concatenation 's': "+s);
		System.out.println("After concatenation third reference variable 'co': "+co);

		StringBuffer s2 = new StringBuffer("Hello");
		StringBuffer s3 = new StringBuffer("world");
		System.out.println("MUTABLE STRING");
		System.out.println("Mutable String 's2' address: " + System.identityHashCode(s2));
		System.out.println("Before appending s2: "+s2);
		System.out.println("Mutable String 's3' address: " + System.identityHashCode(s3));
		System.out.println("Before appending s3: "+s3);
		s2.append(s3);
		System.out.println("After appending s2: "+s2);
		System.out.println("Mutable String 's2' address: " + System.identityHashCode(s2));
	}
}
```

```
OUTPUT:
------------
IMMUTABLE STRING
Immutable String 's' address: 2018699554
Before concatenation 's': Hello
Immutable String 's1' address: 1311053135
Before concatenation 's1': world
Immutable String 'co' address: 118352462
After concatenation 's': Hello
After concatenation third reference variable 'co': Helloworld
MUTABLE STRING
Mutable String 's2' address: 1550089733
Before appending s2: Hello
Mutable String 's3' address: 865113938
Before appending s2: world
After appending s2: Hello world
Mutable String 's2' address: 1550089733
```

Note:

System-class

identityHashCode() -method to find the address of the variable.

TYPES OF VARIABLES:

LOCAL VARIABLE:

Local variable scope is only inside the method/block.

We cant use access specifiers for local variable.

We need to initialise local variable.

INSTANCE/GLOBAL VARIABLE:

Global variable declared inside class and outside methods.

We can use access specifiers for global variables.

Need not initialise value for global variable.It will have default value.

STATIC VARIABLE:

Static variable retains value between the objects(Since static variable is shared by all the objects)

2 ways of accessing static variables:

classname.variablename

variablename

Static method:

If u specify a method as static, we need not create object for calling the methods.

2 ways of accessing static methods:

classname.methodname();

methodname();

Program to show the difference between global variable and static variable:

```
public class Variable {
static int a=20;
int b=10;
public void print() {
      System.out.println(a);
      System.out.println(b);
      a++;
      b++;
}
public static void main(String[] args) {
      Variable a=new Variable();
      a.print();
      Variable b=new Variable();
      b.print();
      Variable c=new Variable();
      c.print();
}
}
```

OUTPUT:

```
20
10
21       //static variable retains value between objects
10
```

FINAL VARIABLE:

1.If a variable declared as "final"---->we can't change the value assigned.

2.If a method as declared as "final"----->We can't override that method.

3.If a class as declared as "final"-------->We can't inherit that class.

Access Modifiers:

It is used to set the access level for classes, attributes, methods and constructors.

We divide modifiers into two groups:

Access Modifiers - controls the access level

Non-Access Modifiers - do not control access level, but provides other functionality

Access Modifiers:

private

default or no modifier

protected

public

A)Public(Project level access/Global level access)

In same package as well as different package,we can access the private variables and methods of one class in another class using extends keyword and creating separate object.

B)Protected(Global level access)

In same package,

Using extends keyword and by creating separate object also we can access the properities of another class.

If it is in different package,

using extends keyword only we can access,We cannot create object and access the protected attributes of a class.

C)Default(Package level access)

In same package only we can access(In different package we cannot access)

using extends keyword and by creating object.

D)Private(Class level access)

Outside class,we cannot access private variables.If we want to access private variables outside the class we use getters

and setters.

Non Access Modifiers:

static

final

abstract

synchronised---->When Methods given with synchronised keyword,it can only be accessed by one thread at a time

transient--->At the time of serialization, if we don’t want to save value of a particular variable in a file, then we use transient keyword. When JVM comes across transient keyword, it ignores original value of the variable and save default value of that variable data type.

volatile--->volatile variable in Java is a special variable which is used to signal threads, a compiler that this particular variables value are going to be updated by multiple threads inside Java application. By making a variable volatile using the volatile keyword in Java, application programmer ensures that its value should always be read from main memory and thread should not use cached value of that variable from their own stack.

strictfp-->Java strictfp keyword ensures that you will get the same result on every platform if you perform operations in the floating-point variable.

STORAGE IN JAVA:

What is Stack Memory?

Stack in java is a section of memory which contains methods, local variables, and reference variables. Stack memory is always referenced in Last-In-First-Out order. Local variables are created in the stack.

Stack memory structure is mostly implemented for providing static memory allocation.

What is Heap Memory?

Heap is a section of memory which contains Objects and may also contain reference variables. Instance variables are created in the heap

Program Counter Register: Programs are a set of instructions.Program counter register holds the address of the upcoming instructions to be executed.

CONSTRUCTOR:

Whenever you create object for a class,default costructor will automatically invoked since class name and constructor name are same.

Purpose of constructor:

-->A constructor in Java is a special method that is used to initialize objects.

-->The constructor is called when an object of a class is created.

-->All classes have constructors by default: if you do not create a class constructor yourself, Java creates one

Rules:

-->constructor name must match the class name.

-->It cannot have a return type (like void)

Types:

Default constructor:(without arguments)

```
public class MyClass {
    int x;
    public MyClass() {
     x = 5;
    }
 public static void main(String[] args)
{
 MyClass myObj = new MyClass();
 System.out.println(myObj.x);
}
}
// Outputs 5
```

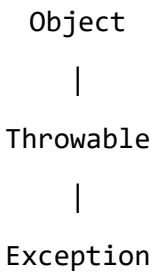
Parameterized constructor:(With arguments)

```
public class MyClass {
 int x;
public MyClass(int y) {
 x = y;
}
 public static void main(String[] args) {
MyClass myObj = new MyClass(5);
System.out.println(myObj.x);
}
}
// Outputs 5
```

Exception:

Exception results in abnormal termination of the program (or)Whenever exception occurs, program terminates at the particular line itself.

We can avoid the abnormal termination of the program by handling the exception.



Exception handling:

try block:

The code which is expected to throw exceptions is placed in the try block.

Catch block:

When an exception occurs, that exception occurred is handled by catch block associated with it.

Every try block should be immediately followed either by a catch block or finally block.

finally block:

The finally block follows a try block or a catch block.

A finally block of code will always be executed whether exception occurs or not.

we can have,

```
try{
}
catch(Exception e){
}
```

```
try{
}
catch(ArithmeticException e){
}
catch(Exception e){
}
```

```
try{
}
catch(Exception e){
}
finally{
}
```

```
try{
}
finally{
}
```

Impossible combination:

-->Throwable class is the super class of all exception.

-->This will handle all the exceptions.so all the child catch blocks will not be used/reached by any code.

-->so while writing the program itself,it will show error.

```
try
{
}
Catch(Throwable e){
}
Catch(Exception e){
}
Catch(ArithmeticException e){
}
```

possible combination:

-->we can specify multiple catch blocks

```
try
{
}
Catch(ArithmeticException e){
}
Catch(IndexOutOfBoundException e){
}
Catch(Throwable e){
}
```

throw:

It is used to explicitly throw an exception.

You cannot throw multiple exceptions.Only one exception can be thrown at a time.

Throw is used within the method.

Checked exception cannot be handled using throw.

Throw is followed by an instance.

throws:

throws keyword is used to declare an exception at method level.

Can handle checked as well as unchecked exception.

Throws is used with the method signature.

You can handle multiple exceptions using throws.

Errors :

These are not exceptions at all, but problems that arise beyond the control of the user or the programmer. Errors are typically ignored in your code because we cannot handle the error.

For example, JVM crash, stack overflow, insufficient memory. They are also ignored at the time of compilation.

Note:

-->A catch block cannot exist without a try statement.

-->It is not compulsory to have finally block whenever a try/catch block is present.

-->The try block cannot be present without either catch block or finally block.

-->No code can be present in between the try, catch, finally blocks.

-->We can have 'n' number of exception throwing statements in try block, once an exception caught control will be moved to catch block, all the remaining lines won’t be executed.

WHAT IS AUTOMATION TESTING?

Test Automation is a process that makes use of automation testing tools to execute pre-scripted tests on applications, then compares the test results to the expected behaviour and reports it to the testers.

Benefits:

1. It executes tasks automatically
2. Faster feedback
3. Increase effectiveness
4. Efficiency
5. Coverage of the software testing.

SELENIUM

- Selenium is a test automation tool used to automate web-based application.
- Selenium is an open-source automation testing tool which is used for automating tests carried out on different web-browsers.

Advantages of Selenium

1. Language and Framework support
2. Opensource
3. Multi browser support
4. Support across various operating system.
5. Less hardware usage

Drawback:

We can't automate desktop application using selenium.

Selenium Components:

1. Selenium IDE
2. Selenium RC (Remote control)
3. Selenium WebDriver
4. Grid

Selenium IDE: -

- Selenium IDE is a Firefox plugin which is used to create and execute test cases
- It records and plays back the interactions which the user had with the web browser
- Using IDE, you can export the programming code in different languages: Java, Ruby, Python and so on

Selenium Grid: -

- Selenium Grid is used for parallel testing or distributed testing. It allows us to execute test scripts parallelly on different machines

Selenium RC: -

- Selenium Remote Control (RC) is used to write test cases in different Programming languages
- In Selenium IDE, we can run the recorded scripts only in Firefox browser, whereas, in Selenium RC, we can run the recorded script in any browser like IE, Chrome, Safari, Opera and so on

Selenium WebDriver: -

- Selenium WebDriver is a tool used to automate testing for web application
- It allows us to execute tests against different browsers like Firefox, Chrome, IE & Safari
- Selenium WebDriver eliminated the use of Selenium Server thus making it work faster than RC

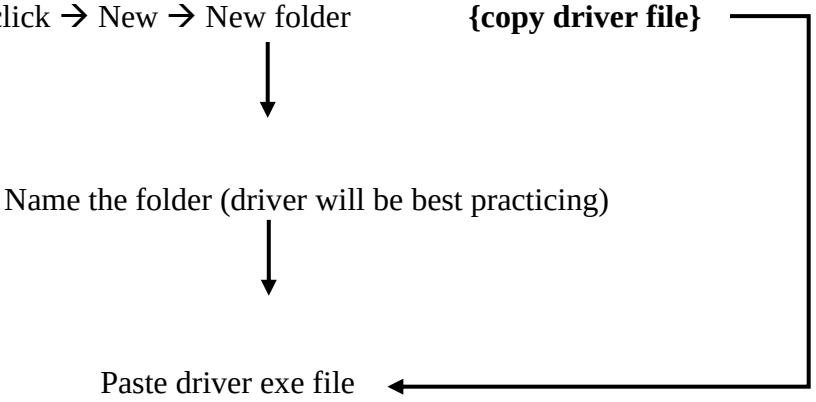
Latest version of selenium

selenium-server-standalone-3.141.59

Browser	**Driver**	**Key**	**Class**
Chrome	chrome driver	webdriver.chrome.driver	ChromeDriver
Firefox	gecko driver	webdriver.gecko.driver	FirefoxDriver
Internet Explorer	ie driver	webdriver.ie.driver	InternetExplorerDriver

To configure Driver:

Create new project → Right click → New → New folder



To configure browser:

System.setProperty(key , path of the driver);

- System is a class.
- ***setProperty()*** is a method in this we have to pass key and value.
- It is used to set the key and path location of driver.

WebDriver

- WebDriver is a interface.
- WebDriver is a web automation framework that allows you to execute your tests against different browser.
- WebDriver also enables you to use a programming language in creating your test script.
- It is mainly used for providing the connection between the browser and local system.
- It acts as a bridge.

Instantiate for webdriver:

WebDriver refName = new DriverClassName();

WebDriver Methods:

Method	**Return Type**	**Description**
driver.get("url");		To launch given URL in the configured browser
driver.gettitle();	String	To get the title of the webpage launched
driver.getCurrentUrl();	String	To get the URL of the current webpage
driver.quit();		To quit the browser
driver.manage().window().maximize()		To maximize the browser tab
driver.switchTo().alert()	Alert	Switch to alert

Sample Program:
==========

```
package org.day.one;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class Sample {

	public static void main(String[] args) {

		//configure your browser
		System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A.
R\\eclipse-workspace\\SCalss1\\Drivers\\chromedriver.exe");

		//Instantiate for webdrivers
		WebDriver d=new ChromeDriver();

		//to maximize browser
		driver.manage().window().maximize();

		//to launch given url
		d.get("https://www.facebook.com");

		//to get the title of the webpage launched
		String title = d.getTitle();
		System.out.println(title);

		//to get the url of the current page
		String url = d.getCurrentUrl();
		System.out.println(url);

		//to quit the browser
		driver.quit();

	}

}
```

Locator:

When we need to perform any action in the browser, we have to find the locator. Locators used to find and match the elements of your page that it needs to interact with.

In ***By*** class all the locator methods are available

By→ Abstract Class Package→ selenium.By

Static Methods of ***By***

- **id**
- **name**
- **className**
- **xpath**
- **tagName**
- **linkText**
- **partiallyLinkText**
- **cssSelector**

WebElement

WebElement → Interface **Package →**selenium.WebElement

Anything that is present on the web page is a WebElement such as text box, button, etc. WebElement represents an HTML element. **Selenium WebDriver** encapsulates a simple form element as an object of the WebElement**.** It basically represents a DOM element and all the HTML documents are made up by these HTML elements.

Methods of WebElement:

txt.sendKeys("sequence")		To pass any values in the textbox
btn.click()		To perform click option for button
txt.getText()	String	To print the text in the WebElement
txt.getAttribute("AttribiuteName)	String	To print the attribute value in the web element
txt.getAttribute("value")	String	To print the values we have passed in text box

1) By.id()

id value is "email". and it's a textbox, then we can find the locator by

```
WebElement txt = driver.findElement(By.id("email"));
```

2) By.name()

name value is "email". and it's a textbox, then we can find the locator by

```
WebElement txt = driver.findElement(By.name("email"));
```

3) By.className()

class value is "`inputtext _55r1 _6luy`". and it's a textbox, then we can find the locator by

```
WebElement txt = driver.findElement(By.className("inputtext _55r1 _6luy"));
```

```java
package org.day.two;


import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class One {

	public static void main(String[] args) {

		System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A.
R\\eclipse-workspace\\SDay2\\driver\\chromedriver.exe");

		WebDriver driver=new ChromeDriver();

		driver.get("https://www.facebook.com/");

		//to find locator of the user name
		WebElement txtUserName = driver.findElement(By.id("email"));
		//to pass values in the text box
		txtUserName.sendKeys("azarara321@gmail.com");

		//to find locator of the password field
		WebElement txtPassword = driver.findElement(By.name("pass"));
		txtPassword.sendKeys("123456789");

		//to find locator of login button
		WebElement btnLogin = driver.findElement(By.className("login"));
		//to click button
		btnLogin.click();
	}
}
```

4) By.xpath()

Xpath is one of the locator available in webpage.

/ → absolute path (It find the WebElement from the top of the downstream.)

// → relative path (It find the WebElement from that particular tag.)

Reason for going to Xpath:

- For validating the locator. (We can check this in webelements by **ctrl+F)**
- When id,classname,name is not present.

General syntax:

//tagName [@attributeName = ' attributeValue ']

If there is more than one attribute value is same for same attribute names in the web element then we have to go for index(matching with more than one loctor)

General syntax:

(//tagName [@attributeName = ' attributeValue ']) [2]

When we try to find locator, if only text is present there then we go for text

text()

//tagName [text( ) = ' text name ']

contains()

//tagName [contains (text( ) , 'partially text')]

contains() *using attributes*

//tagName [contains (@attributeName , 'attributeValue')]

Example Program

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\Zpractice

        WebDriver driver=new ChromeDriver();
        driver.get("http://demo.automationtesting.in/Register.html");

        //to find xpath
        WebElement txtMail = driver.findElement(By.xpath("//input[@type='email']"));
        txtMail.sendKeys("abc@gmail.com");

        //to find path when more locators found
        WebElement txtFst = driver.findElement(By.xpath("(//input[@type='text'])[1]"));
        txtFst.sendKeys("Azar");

        WebElement txtLst = driver.findElement(By.xpath("(//input[@type='text'])[2]"));
        txtLst.sendKeys("AR");

        }

    }
```

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\'

        WebDriver driver=new ChromeDriver();
        driver.get("https://www.facebook.com/");

        WebElement txtmail = driver.findElement(By.id("email"));
        txtmail.sendKeys("azarara321@gmail.com");

        //to print the entered values
        String mail = txtmail.getAttribute("value");
        System.out.println(mail);

    }
}
```

```java
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay

        WebDriver driver=new ChromeDriver();

        driver.get("http://greenstech.in/selenium-course-content.html");

        //contains text
        WebElement txtAdd = driver.findElement(By.xpath("//h6[contains(text(),'Greens')]"));
        String text = txtAdd.getText();
        System.out.println(text);

        //text
        WebElement txtAdd1 = driver.findElement(By.xpath("//p[text()='mail-info']"));
        String text2 = txtAdd1.getText();
        System.out.println(text2);

    }
}
```

==========

```java
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) {

            System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspa

            WebDriver driver=new ChromeDriver();

            driver.get("http://greenstech.in/selenium-course-content.html");

            //contains text
            WebElement txtAdd = driver.findElement(By.xpath("//h6[contains(text(),'Greens')]"));
            String text = txtAdd.getText();
            System.out.println(text);

            //text
            WebElement txtAdd1 = driver.findElement(By.xpath("//p[text()='mail-info']"));
            String text2 = txtAdd1.getText();
            System.out.println(text2);
    }
}
```

DEBUG

- It is the step by step verification.
- We can easily identify the step where the code getting exception.

Steps for debug:

1. Set a break point(double click at the line number.
2. R+click
3. Debug as
4. Java application
5. Switch

F6=> stepover

F8=>close debug

Types of Debug

- Eclipse debugger.
- Firefox JavaScript debugger.
- Dynamic debugging technique
- On line debugging tool.

For page loading time issue

Thread.sleep(milliseconds);

Thread => class

sleep()=>static method

It throws Interrupted Exception

1000 milli second = 1 second

ACTIONS

Actions → Class

Package → selenium.interactions

MouseOverAction

- When we place a mouse on some option it will display a list of subOption.
- For mouseOverAction we can use Actions class.

Declare Actions

Actions a = new Actions(WebDriver ref name);

Methods in Action

a.moveToElement(WebElement)		To move mouse point or curser to webelement
a.dragAndDrop(source,target)		For drag and drop option
a.sendKeys(source,sequence)		To pass any values in the textbox
a.sendKeys(char sequence)		To pass any values in the textbox
a.doubleClick(WebElement)		To perform double click
a.contextClick(WebElement)		To perform right click

The menu list disappears within the fractions of seconds before Selenium identify the next submenu item and perform click action on it.

So, it is better to use .perform() method.

Whenever Actions methods takes place we use or end that method with ***perform()*** to perform that method.

If we use one method to perform we use ***perform();***

If we use more than one method in one line of code we use ***buid().perform();***

Example Program

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-works
        WebDriver driver=new ChromeDriver();

        //declare Actions Class
        Actions a=new Actions(driver);
        driver.get("https://www.shopclues.com/wholesale.html");

        //to move mouse point
        WebElement sport = driver.findElement(By.xpath("//a[text()='Sports & More']"));
        a.moveToElement(sport).perform();

        //if time issues takes place
        Thread.sleep(3000);

        WebElement weight = driver.findElement(By.xpath("//a[text()='Weight Gainers']"));
        weight.click();
    }

}
```

==========

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspa
        WebDriver driver=new ChromeDriver();

        driver.manage().window().maximize();
        driver.get("http://demo.guru99.com/test/drag_drop.html");

        Actions ac=new Actions(driver);

        //to perform drag and drop operation
        WebElement sour = driver.findElement(By.xpath("//a[text()=' BANK ']"));
        WebElement dest = driver.findElement(By.xpath("(//li[@class='placeholder'])[1]"));
        ac.dragAndDrop(sour, dest).perform();

        Thread.sleep(3000);

        WebElement sour2 = driver.findElement(By.xpath("//a[text()=' 5000 ']"));
        WebElement dest2 = driver.findElement(By.id("amt7"));
        ac.dragAndDrop(sour2, dest2).perform();
    }
}
```

ROBOT

- **Robot → class**
- **Package → java.awt**
- **Exception →AwtException(Abstract window toolkit)**

Robot class is a class which is used to perform the keyboard action in java.

Declare Robot

Robot r=new Robot();

Mehods of Robot:

r.keyPress()		To press any key
r.keyRelease()		To release key(whenever keyPress() takes place keyRelease() should also mentioned)
KeyEvent => Class(takes place inside the robot method where it consists of all keyboard keys).		

Example Program

```
package org.com;
import java.awt.AWTException;
import java.awt.Robot;
import java.awt.event.KeyEvent;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;
public class hai {
    public static void main(String[] args) throws AWTException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspac
        WebDriver driver=new ChromeDriver();
        driver.get("http://www.greenstechnologys.com/");
        Actions a=new Actions(driver);
        //Robot class declaration
        Robot r= new Robot();
        WebElement paragraph = driver.findElement(By.xpath("(//p[@style-size:18px;'])[2]"));
        //double click in actions
        a.doubleClick(paragraph).perform();
        //right click in actions
        a.contextClick(paragraph).perform();
        //for using down key
        r.keyPress(KeyEvent.VK_DOWN);
        r.keyRelease(KeyEvent.VK_DOWN);
        //for using enter key
        r.keyPress(KeyEvent.VK_ENTER);
        r.keyRelease(KeyEvent.VK_ENTER);
    }
}
```

ALERT

Alert → interface

Alert is a small message box displayed on the screen to give some information to the user. Alert and webpage are different Alert has no locators. When alert appeared first we need to switch into the alert to handle the alert, then only user can perform the next operation in the webpage.

To switch into the alert

Alert a=WebDriver.switchTO().alert();

Types of Alert

1. Simple Alert → Contains only ok button
2. Confirm Alert → Contains both ok and cancel button
3. Prompt Alert → Contains text box with ok and cancel button

Methods in Alert

r.accept()	**Accept the alert**
a.dismiss()	**Dismiss the alert**
a.sendKey()	**To insert the values**
a.getText	**To print the text in the alert**

Simple Alert

```
package org.com;

import org.openqa.selenium.Alert;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\

        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();

        driver.get("http://demo.automationtesting.in/Alerts.html");

        //Switching into Simple alert
        Alert a = driver.switchTo().alert();

        //to accept in the alert
        a.accept();
    }

}
```

Confirm Alert

```
package org.com;

import org.openqa.selenium.Alert;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. F

        WebDriver driver=new ChromeDriver();
        driver.get("http://demo.automationtesting.in/Alerts.html");

        //Switching to confirm alert
        Alert a = driver.switchTo().alert();

        // accept the alert
        a.accept();
    }
}
```

Confirm Alert

```
package org.com;

import org.openqa.selenium.Alert;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\

        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();

        driver.get("http://demo.automationtesting.in/Alerts.html");

        //Switching to Prompt  alert
        Alert a = driver.switchTo().alert();

        //passing the values in alert
        a.sendKeys("Azar");

        //accept the alert
        a.accept();
    }
}
```

JAVASCRIPT EXECUTOR

Java Script → Interface

Package → selenium.JavaScriptExecutor

- JavaScriptExecutor is an Interface that helps to execute JavaScript through Selenium Webdriver.
- In Selenium Webdriver, locators like XPath, CSS, etc. are used to identify and perform operations on a web page.
- In case, these locators do not work you can use JavaScriptExecutor. You can use JavaScriptExecutor to perform a desired operation on a web element.

To implement JavaScript

1. Type casting
   JavaScriptExecutor js = (JavaScriptExecutor) WebdriverRef.Name
2. Use methods in JavaScript

Methods of JavaScript

js.executeScript ("JavaSript code" ,WebElement reference);

JavaScript Codes

arguments[0].setAttribute('value','input txt')		To pass any input text in text box
return arguments[0].getAttribute('value')	String	Retrieve the values of user entered text
arguments[0].click()		For button click
arguments[0].scrollIntoView(false)		Scroll down
arguments[0].scrollIntoView(true)		Scroll up

When ever we want retrieve the user entered values the method returns Object. With that object reference name we can do upcasting to get string values to be printed.

String s1=(String)object;

Example

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\
        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        JavascriptExecutor js=(JavascriptExecutor)driver;
        //pass values in the text box using Java Script
        WebElement txtEmail = driver.findElement(By.id("email"));
        js.executeScript("arguments[0].setAttribute('value','azarara321@gmail.com')", txtEmail);
        //retrieve the user entered text in the webelement
        Object object = js.executeScript("return arguments[0].getAttribute('value')", txtEmail);
        //upcasting
        String s1=(String)object;
        System.out.println(s1);
        WebElement txtPass = driver.findElement(By.id("pass"));
        js.executeScript("arguments[0].setAttribute('value','12345')", txtPass);
        //button click using Java Script
        WebElement btnLogin = driver.findElement(By.name("login"));
        js.executeScript("arguments[0].click()", btnLogin);
        }
    }
```

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay7\\driv

        WebDriver driver=new ChromeDriver();
        JavascriptExecutor js=(JavascriptExecutor)driver;
        driver.get("http://toolsqa.com/");

        WebElement txtScroll = driver.findElement(By.xpath("//span[text()='Selenium Training Benefit']"));
        //scroll up
        js.executeScript("arguments[0].scrollIntoView(false)", txtScroll);
        //scroll down
        js.executeScript("arguments[0].scrollIntoView(true)", txtScroll);
    }
}
```

TAKES SCREENSHOT

TakesScreenshot → Interface

TakesScreenshot ts=(TakesScreenshot)WebDriverRef.Name;

Methods

File src = ts.getScreenshotAs(OutputType.FILE);

File → **Return type for *getScreenshotAs( );***

Steps:

1. Typecast
2. Store screenshot in default
3. Create a screenshot

The output format of the screenshot will be Base64, Bytes, Class, FILE.

In order to store the screenshot in our project folder

1. After taking the scerrnshot, create a file class
2. Create a folder in the package and give the path of the folder in the File class
3. In the path at last add the name of the screen shot.
4. Use ***FileUtils.copyFile(source, destination);*** to copy ***src*** file and past it in the desired project folder. ***Source*** → ref name of ***getScreenshotAs( ).***
5. copyFile() is a static method in the FileUtils class.

Example

```
package org.com;

import java.io.File;
import java.io.IOException;
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay7\\d

        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("http://www.greenstechnologys.com/");

        //typecasting
        TakesScreenshot ts=(TakesScreenshot)driver;

        //take screenshot
        File src = ts.getScreenshotAs(OutputType.FILE);

        //copy from src and save it in destination folder
        File dest= new File("C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay7\\Screenshot\\Greesn1.png");
        FileUtils.copyFile(src, dest);
    }

}
```

VISIBLITY OF WEBELEMENT

To check visibility of the WebElements

isEnabled() - method to check whether the web element is enabled or not.

isDisplayed() - method to check whether the web element is displayed(present) or not

isSelected() -method to check whether the web element is selectable or not

It is widely used in radio button, dropdown, checkbox

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class Task_02 {

    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\
        WebDriver driver =new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        WebElement btnLogin = driver.findElement(By.name("login"));
        //to check whether WebElement is displayed
        boolean displayed = btnLogin.isDisplayed();
        System.out.println(displayed);
        //to check whether WebElement is enabled
        boolean enabled = btnLogin.isEnabled();
        System.out.println(enabled);
        WebElement btnCreate = driver.findElement(By.xpath("//a[text()='Create New Account']"));
        btnCreate.click();
        Thread.sleep(3000);
        WebElement rdoGender = driver.findElement(By.name("sex"));
        rdoGender.click();
        //to check whether WebElement is selected
        boolean selected2 = rdoGender.isSelected();
        System.out.println(selected2);
    }
}
```

FRAMES

html embedded inside another html

When any locator is placed inside the frame we cannot directly access the locator.

First we need to switch into the frame, then only we can access frame.

To switch into frame

First we have to check frame is available in DOM or not.

R + click → view frame source

Or

Inspect→cntrl + F →//iframe or //frameset etc.

Methods to switch into frame (method-overloading)

driver.switchTo().frame(string id);

driver.switchTo().frame(string name);

driver.switchTo().frame(web element);

driver.switchTo().frame(index);

Methods to switch out of frame

To switch from current frame to immediate parent frame (frame inside frame concept)

driver.switchTo().parentframe();

To switch the control from any frame to main.

driver.switchTo().defaultContent();

Example

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay

        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();

        driver.get("https://netbanking.hdfcbank.com/netbanking/?_ga=2.176378149.1819882415.153388321

        //switching into frame using string id
        driver.switchTo().frame("login_page");
        Thread.sleep(2000);

        WebElement login = driver.findElement(By.xpath("//img[@src='/gif/continue_new1.gif?v=1']"));
        login.click();

        //to switch the contril from frame to main
        driver.switchTo().defaultContent();
    }
}
```

WINDOWS HANDLING

Whenever we execute any program it can access current window webelement only.

When we have multiple windows to switch control between windows we go for windows handling.

To switch into other window

driver.switchTo().window(String URL)

driver.switchTo().window(String title)

driver.switchTo().window(Window ID)

To find window ID

Parent id:

driver.getWindowHandle() → String

Child id:

driver.getWindowHandles() → Set<String>

```
package org.com;

import java.util.Set;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SD

        WebDriver driver= new ChromeDriver();
        driver.manage().window().maximize();

        driver.get("https://www.snapdeal.com/");

        WebElement txtSearch = driver.findElement(By.id("inputValEnter"));
        txtSearch.sendKeys("Hand sanitizer");
        WebElement btnSearch = driver.findElement(By.xpath("//span[text()='Search']"));
        btnSearch.click();
        WebElement btnPro = driver.findElement(By.xpath("(//img[@class='product-image '])[1]"));
        btnPro.click();

          //to get parent window id
          String parentWind = driver.getWindowHandle();

          //to get all window id including parent
          Set<String> allWind = driver.getWindowHandles();

          //to switch into child window by iteration
          for (String cd : allWind) {

              if(!(parentWind.equals(cd))) {

                  //switch to child window
                  driver.switchTo().window(cd);
              }
          }

          WebElement btnAdd = driver.findElement(By.id("add-cart-button-id"));
          btnAdd.click();

          //to switch to parent window
          driver.switchTo().defaultContent();
    }
}
```

WEBTABLE

To access the elements in the webtable we need to go for it.

Every table must be in the pattern of

```
<table id="customers">
  <tbody>
    <tr>
      <th>Company</th>
      <th>Contact</th>
      <th>Country</th>
    </tr>
    <tr>
      <td>Alfreds Futterkiste</td>
      <td>Maria Anders</td>
      <td>Germany</td>
    </tr>
    <tr>…</tr>
    <tr>…</tr>
    <tr>…</tr>
    <tr>…</tr>
    <tr>…</tr>
  </tbody>
</table>
```

table - tag name

tr - table row

th - table heading

td - table data

To find the web table

```
// Find the table
WebElement table = driver.findElement(By.xpath("//table[@id='customers']"));

// (or)

WebElement table = driver.findElement(By.xpath("(//table)[4]"));

// (or)


List<WebElement> tables = driver.findElements(By.xpath("//table"));
WebElement table = tables.get(3);

// (or)

List<WebElement> tables = driver.findElements(By.tagName("table"));
WebElement table = tables.get(3);
```

```java
//to print first table in the webpage
package org.com;

import java.util.List;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai{
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclips
        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.w3schools.com/html/html_tables.asp");

        //to get the table id useing xpath
        WebElement main = driver.findElement(By.xpath("//table[@id='customers']"));

        //to get the rows elements using tagname
        List<WebElement> tr = main.findElements(By.tagName("tr"));

        //iterating each rows
        for(int i=0;i<tr.size();i++) {
            WebElement row = tr.get(i);

            //to get heading in that table
            List<WebElement> th = row.findElements(By.tagName("th"));
            for(int j=0;j<th.size();j++) {
                WebElement head = th.get(j);
                String text = head.getText();
                System.out.println(text);
            }

            //to get data in that table
            List<WebElement> td = row.findElements(By.tagName("td"));
            for(int j=0;j<td.size();j++) {
                WebElement data = td.get(j);
                String text1 = data.getText();
                System.out.println(text1);
            }
        }
        driver.quit();
    }
}
```

To print the first row of the table

```java
public class hai{
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclips
        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.w3schools.com/html/html_tables.asp");
        //to get the table id useing xpath
        WebElement main = driver.findElement(By.xpath("//table[@id='customers']"));
        //to get the rows elements using tagname
        List<WebElement> tr = main.findElements(By.tagName("tr"));
            //to get 4th row of the table
            WebElement row = tr.get(3);
            //to get data in that table
            List<WebElement> td = row.findElements(By.tagName("td"));
            for(int j=0;j<td.size();j++) {
                WebElement data = td.get(j);
                String text1 = data.getText();
                System.out.println(text1);
            }
        driver.quit();
    }
}
```

File Edit Format View Help

WebTable(Without tr td tags----------WebTable with common atributes)

==========

```
<div class=row>
<div class=cell>Name</div>
<div class=cell>Email id</div>
<div class=cell>Mob</div>
</div>

<div class=row>
<div class=cell>Sara</div>
<div class=cell>Sara@gmail.com</div>
<div class=cell>9876543210</div>
</div>

<div class=row>
<div class=cell>Amy</div>
<div class=cell>Amy@yahoo.com</div>
<div class=cell>9846342123</div>
</div>

List<WebElement> rows=driver.findElements(By.xpath("//div[@class='row']"));
for(WebElement row: rows){

List<WebElement> cells=row.findElements(By.xpath("//div[@class='cell']"));
for(WebElement cell : cells){

String data=cell.getText();
System.out.println(data);
}

}
```

Output:

======

```
Name
Emailid
Mob
Sara
Sara@gmail.com
9876543210
Amy
Amy@gmail.com
9846342123
```

DROPDOWN

Whenever dropdown takes place we need to go for Select class.

Select s=new Select(WebElement)

Dropdown Syntax

```
Drop Down syntax:
---------------------

<select>
<option value="IND">India</option>
<option value="US">United States</option>
</select>

Select:C
--------
selectByIndex(10)
selectByValue("IND")
selectByVisibleText("India")
```

Methods:

```
Select:class
-----------
1.selectByValue()-m
2.selectByVisibleText()-m
3.selectByIndex()-m
4.getOptions()-m
5.getAllSelectedOptions()-m
6.getFirstSelectedOption()-m
7.isMultiple()-m
8.deSelectByIndex()-m
9.deSelectByValue()-m
10.deSelectByVisibleText()-m
11.deSelectAll()-m
```

Example:

```
import org.openqa.selenium.support.ui.Select;

public class hai {

    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\S
        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        WebElement btnCreate = driver.findElement(By.xpath("//a[text()='Create New Account']"));
        btnCreate.click();
        WebElement year = driver.findElement(By.id("year"));
        //Select class to select the webelement
        Select s=new Select(year);
        //to select the required webelement using index
            s.selectByIndex(2);
        //to select the required webelement using index
            s.selectByValue("1996");
        //to select the required webelement using index
            s.selectByVisibleText("1998");
        //to select the required webelement using index
            s.deselectByIndex(2);
        //to select the required webelement using index
            s.deselectByValue("1996");
        //to select the required webelement using index
            s.deselectByVisibleText("1998");
        driver.quit();
    }
}
```

CSS VALUE

```
CSS(Cascading Style sheets) VALUE:
==================================
<p style=”color:green;font-size:24px;“>Greens Technologies</p>

-->“style” attribute is way of declaring CSS properties of any HTML element.
-->Each attribute has a name and a value, separated by a colon (:).
-->Each property declaration is  separated by a semi-colon (;).

Disadvantage of getAttribute("attributename")
---------------------------------------------------------
We cannot get the value of each attribute separately using getAttribute()

WebElement f1 = driver.findElement(By.xpath("//p[text()='Greens Technologies']"));
String text=f1.getAttribute("style");
System.out.println(text);

OutPut:
---------
color:green;font-size:24px

Instead of getAttribute(),we go for getCssValue("name")
-------------------------------------------------------
System.out.println(f1.getCssValue("font-weight"));
System.out.println(f1.getCssValue("color"));
System.out.println(f1.getCssValue("font-size"));
System.out.println(f1.getCssValue("background-color"));
System.out.println(f1.getCssValue("text-align"));

output:
------
400
rgba(28, 30, 33, 1)
12px
rgba(255, 255, 255, 1)
center
```

If the tag name changes dynamically we can use *.

Eg:

//button[@name=’login’] →here the button name changes dynamically

//*[@name=’login’] → use *

Highlighting text

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class Test_01 {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Azar. A. R\\eclipse-workspace\\Zpractice\\
        WebDriver driver= new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        WebElement txtHighlight = driver.findElement(By.xpath("//h2[contains(text(),'Facebook helps')]"));
        JavascriptExecutor js=(JavascriptExecutor) driver;

        //Highlighting the text in webpage
        js.executeScript("arguments[0].setAttribute('style','background:yellow')",txtHighlight);
    }

}
```

```
package org.com;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class Task_02 {

    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar.
        WebDriver driver =new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        WebElement btnLogin = driver.findElement(By.name("login"));
        String color = btnLogin.getCssValue("background-color");
        String fontSize = btnLogin.getCssValue("font-size");
        String width = btnLogin.getCssValue("width");
        String fam = btnLogin.getCssValue("font-family");
        System.out.println(color);
        System.out.println(fontSize);
        System.out.println(width);
        System.out.println(fam);

    }

}
```

NAVIGATION COMMANDS

driver.navigate().to("url")	To navigate to given url
driver.navigate().forward()	To move to the next page
driver.navigate().backward()	To move to the current page
driver.navigate().refresh()	To refresh the particular page

get() Will wait till the webpage loaded completely and it will not maintain cookies(history)

navigate().to() Will not wait till the page load completely. But it will maintain browser history so that we can move to the previous page and next page.

```
package org.com;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class Task_02 {

    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A.
        WebDriver driver =new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");

        //navigate to next url
        driver.navigate().to("https://www.redbus.in/");
        //to go to previous page (moves to facebook again)
        driver.navigate().back();
        //to go to next page (moves to redbus again)
        driver.navigate().forward();
        //to refresh the page (refreshes the page)
        driver.navigate().refresh();
    }
}
```

WAITS (synchronisation)

Types:

1. unconditional wait
2. conditional wait

1. Unconditional Wait

Thread.sleep() → For the given time script will pause its execution. Even the webelement is found earlier the program will not resume until the given time completes.

Disadvantages

1. Application will get slow
2. Performance will be reduced.

2. Conditional Wait

For a given condition we can make our script to wait is known as conditional wait.

Types

1. Implicit Wait
2. Explicit Wait

2.1 Implicit Wait

Whenever we need to find webelement in webpage, if the webelement is not present, before throwing the exception it will wait for the given time. When the webelement appeared the program will resume and it wont wait for the time to complete

If it could not find the webelement it will throw TimeOutException.

It is applicable for all the locators.

Default polling time is 250 ms(milli seconds)[every 250ms it will go and check the webelement found or not)

```java
import java.util.concurrent.TimeUnit;

public class Task_02 {

    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R
        WebDriver driver =new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");

        //implicit wait
        driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

        WebElement txtmail = driver.findElement(By.id("email"));
        txtmail.sendKeys("azar");
    }
}
```

2.2 Explicit Wait

It is applicable for particular locator/ condition.

For the given condition to be satisfied or for finding the webelement till that we can make our program to wait.

Types

1. WebDriverWait
2. FluentWait

- ***Wait*** (Wait is the interface)
- ***FluentWait implements Wait***
- ***WebDriverWait extends FluentWait***

All the methods in fluent wait will also be in WebDriverWait

2.2.1 WebDriverWait

Whenever the time interval we give it will take only in SECONDS (so we cannot overload).

It cannot handle TimeOutException.

Default polling time is 500 ms

WebDriverWait w = new WebDriverWait(driver, 10);

```java
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

public class Test02 {

    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Azar. A. R\\ecl
        WebDriver driver= new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        //instantiate WebDriverWait
        WebDriverWait w = new WebDriverWait(driver, 10);
        w.until(ExpectedConditions.alertIsPresent());

        Alert alert = driver.switchTo().alert();
        alert.accept();

        WebDriverWait w1 = new WebDriverWait(driver, 20);
        w1.until(ExpectedConditions.elementToBeClickable(By.name("login")));

        WebElement login = driver.findElement(By.name("login"));
        login.click();
    }
}
```

2.2.2 FluentWait

We can give the time interval in terms of MILLISECONDS, MACROSECONDS etc.

It can handle TimeOutException.(because here we have additional method)

We can set the polling time.

```java
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.FluentWait;

public class Test02 {

    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Azar. A. R\\eclipse-workspace\\
        WebDriver driver= new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        //instantiate FluentWait
        FluentWait<WebDriver> w = new FluentWait<>(driver).withTimeout(Duration.ofSeconds(20)).
                            pollingEvery(Duration.ofSeconds(1)).ignoring(Throwable.class);
        w.until(ExpectedConditions.alertIsPresent());

        Alert alert = driver.switchTo().alert();
        alert.accept();

        w.until(ExpectedConditions.elementToBeClickable(By.name("login")));

        WebElement login = driver.findElement(By.name("login"));
        login.click();
    }
}
```

Methods

WebDriver d=new ChromeDriver();		
WebDriver = d		
d.get("url")		To get into / launch webpage
d.getTitle()	String	To print the title of the webpage
d.getCurrentUrl	String	To print the current page url
d.manage().window().maximize()		To maximize the window screen
d.findElement()	WebElement	Find the webelement in the webpage
d.findElements()	List<WebElement>	Find all the xpath with similar xpath name
d.quit()		To quit the browser
ALERT		
d.switchTo().alert()	Alert	To switch into alert when alert takes place
FRAMES		
d.switchTo().frame()		to switch into frame
d.switchTo().parantFrame()		to switch from child frame to parent frame
d.switchTo().defaultContent()		to switch from frame to main content

WebElement e=driver.findElement(By.id("email"))		
WebElement=e		
By.locator()		to get the locator
e.sendKeys		to pass the values in the text box
e.click		to click button in web page
e.getText()	String	print existing text in the webpage
e.getAttribute("AttributeName")	String	to print the atribute value
e.getAttribute("value")	String	to print the user enterd value in the webpage
e.findElement()		Find the webelement in the webpage
e.findElements()		Find all the xpath with similar xpath name

Locator	
By.Locator	
By.id()	id value
By.name()	name value
By.className()	class value
By.Xpath()	**attributes**
	//tagName[@attributeName='attributeValue']
	text
	//tagName[text()='textName']
	partial text
	//tagName[contains(text(),'partial text']
	partial attribute
	//tagName[contains(@atttributeName,'attributeValue')]

VISIBLITY OF WEBELEMENT		
e.isSelected()	Boolean	to check whether the webelement is selected or not
e.isEnabled()	Boolean	to check whether the webelement is enabled or not
e.isDisplayed()	Boolean	to check whether the webelement is displayed or not
CSS VALES		
e.getCssValue("details tag");		

Actions a = new Actions(d)		
a.moveToElement(e)		To move curser/ mouse point
a.dragAndDrop(source e, target e)		For drag and drop option
a.doubleClick(e)		double click option
a.contextClick(e)		right click option
a.sendKeys(source e, target words)		pass values in text field
perform()		mentioned at last of actions method

Robot r =new Robot()		
r.keyPress()		Key press operation
r.keyRelease()		Key release operation
	KeyEvent.VK_DOWN	eg for downkey operation inside robot method
	KeyEvent.VK_UP	eg for upkey operation inside robot method
	KeyEvent.VK_SHIFT	eg for shiftkey operation inside robot method

Alert ar=d.switchTo().alert();		
ar.accept()		accept the alert
ar.dismiss()		to dismiss the alert
ar.sendKeys()		to insert the value in alert
ar.getText()	String	to print text in the alert

JavaScriptExecutor js = (JavaScriptExecutor)d		
js.executeScript (“JavaSript code” ,WebElement reference);		to excecute JavaScript
JavaScript Codes		
arguments[0].setAttribute('value','input txt')		to pass any input text in the text box
return arguments[0].getAttribute('value')	String	to retrieve the user enterd values
arguments[0].click()		to click any button
arguments[0].scrollIntoView(false)		for scroll down operation
arguments[0].scrollIntoView(true)		for scroll up operation
arguments[0].setAttribute('style','background:color')		to heighlight the webelement in the wepage

TakesScreenshot ts=(TakesScreenshot) d;		
ts.getScreenshotAs(OutputType.FILE)	File	to take screen short and return file directiry stored
FileUtils.copyFiles(source,destination)		to copy from the default storage and paste it in the destination

d.switchTo().frame()	
d.switchTo().frame(String id)	To switch into frame
d.switchTo().frame(String name)	To switch into frame
d.switchTo().frame(webelement)	To switch into frame
d.switchTo().parantFrame(String id)	to switch out of child frame
d.switchTo().defaultContent()	to switch to main page of DOM

d.switchTo().window()		
d.switchTo().window(string url)		to switch into other window using url
d.switchTo().window(string title)		to switch into other window using page title
d.switchTo().window(string id)		to switch into other window using window id
to find id		
d.getWindowHandle()	String	to get parent id
d.getWindowHandles()	Set<String>	to get all the windows id
d.switchTo().defaultContent()		to switch to main parent window

Waits
Unconditional wait
Thread.sleep(milliseconds)
Conditional Wait
Implicit wait
d.manage().timeouts().implicitlyWait(10,TimeUnit.SECONDS)
d.manage().timeouts().implicitlyWait(10,TimeUnit.MINUTES)
d.manage().timeouts().implicitlyWait(10,TimeUnit.MILLISECONDS)
d.manage().timeouts().implicitlyWait(10,TimeUnit.MICROSECONDS)
d.manage().timeouts().implicitlyWait(10,TimeUnit.HOURS)
d.manage().timeouts().implicitlyWait(10,TimeUnit.DAYS)
Explicit Wait
WebDriverWait
WebDriverWait w = new WebDriverWait(d, 10);
w.until(ExpectedConditions.elementsToBeClickable(By.Name("login");
FluentWait<WebDriver> w = new FluentWait<>(driver).withTimeout(Duration.ofSeconds(20)). pollingEvery(Duration.ofSeconds(1)).ignoring(Throwable.class);
w.until(ExpectedConditions.alertIsPresent());

7Navigation	
d.navigate().to("url")	to navigate to given url
d.navigate().forward()	to move to the next page
d.navigate().backward()	to move to previous page
d.navigate().refresh()	to refresh the current page

MAVEN

Maven/Ant/Gradle:Build tool
==========

Maven is a project management tool or software build tool or dependency management tool.
It is used for project build or dependencies.

Maven is addresses for two aspects.

1)It describes how software is build.
2)It describes the software dependencies.

In Maven execution starts from pom.xml file, it reads the POM.xml file and starts execution.

Uses of Maven:

It is used to build the software application ,manage your dependencies ,run your tests and create reports.

Maven Structure:
---
src/main/java
src/main/resources
src/test/java
src/test/resources
JRE System library
Maven Dependencies
src
target
pom.xml

pom.xml:(Project object model):
---
pom.xml,where we will give our software dependencies.
In maven project,Execution always starts from pom.xml

Maven Lifecycle/Goal:
---
clean---------install----------run

Whenever you change a version.It will clean all the previous cached files.It will download new version's file and you can start testing.

Repositories in Maven:

Local repository(in your local system)------.m2 folder
Central repository
Remote repository

How to update(force saving) your maven project:

Right click your project---click maven---click update project-------enable "Force update of snapshot and releases" checkbox----click ok.

How to find .m2 repository location:(Local repository)
---
window----Preferences-----Maven------user settings----Local repository location will be there.

Zoom Meeting

*Maven - Notepad

File Edit Format View Help

Build Tool:Version:3.6.3

Maven/Ant/Gradle

Maven:

1.Creates the folder structure
2.start the execution from pom.xml
3.Download the latest jar file

steps:

1.Download the Apache maven zip file.

2.Config the JAVA_HOME and MAVEN_HOME in the environment variable:

User variables:

JAVA_HOME(java path without bin----------without including bin folder take the path)
MAVEN_HOME(maven path without bin----------without including bin folder take the path)

System variables:
---
path(give java and maven path with including bin folder)

User variables:

JAVA_HOME(java path without bin----------without including bin folder take the path)
MAVEN_HOME(maven path without bin----------without including bin folder take the path)

System variables:
---
path(give java and maven path with including bin folder)

3.Verify Maven and java configurred correctely:
---
cmd prompt--->

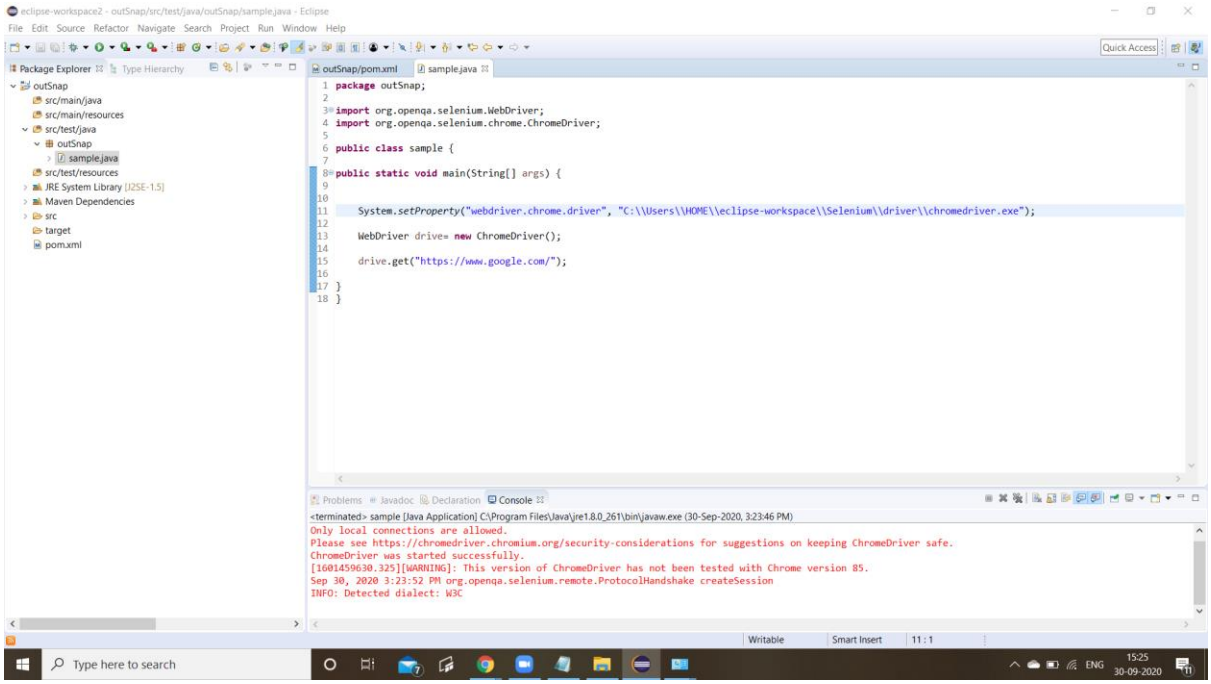
java:
---
java -version

maven:
---
mvn -version

4.Add the maven plugin in eclipse
5.Create maven project in eclipse

Command Prompt

```
Microsoft Windows [Version 10.0.18362.1082]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Prabha>java -version
java version "1.8.0_221"
Java(TM) SE Runtime Environment (build 1.8.0_221-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.221-b11, mixed mode)

C:\Users\Prabha>mvn -version
Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
Maven home: C:\Users\Prabha\Downloads\apache-maven-3.6.3-bin (1)\apache-maven-3.6.3\bin\..
Java version: 1.8.0_221, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk1.8.0_221\jre
Default locale: en_IN, platform encoding: Cp1252
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"

C:\Users\Prabha>
```


eclipse-workspace2 - outSnap/src/test/java/outSnap/sample.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Type Hierarchy
outSnap
src/main/java
src/main/resources
src/test/java
outSnap
sample.java
src/test/resources
JRE System Library [J2SE-1.5]
Maven Dependencies
src
target
pom.xml
outSnap/pom.xml
sample.java
1 package outSnap;
2
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.chrome.ChromeDriver;
5
6 public class sample {
7
8 public static void main(String[] args) {
9
10
11 System.setProperty("webdriver.chrome.driver", "C:\\Users\\HOME\\eclipse-workspace\\Selenium\\driver\\chromedriver.exe");
12
13 WebDriver drive= new ChromeDriver();
14
15 drive.get("https://www.google.com/");
16
17 }
18 }
Problems
Javadoc
Declaration
Console
<terminated> sample [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (30-Sep-2020, 3:23:46 PM)
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
[1601459630.325][WARNING]: This version of ChromeDriver has not been tested with Chrome version 85.
Sep 30, 2020 3:23:52 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
Writable
Smart Insert
11 : 1
Type here to search
ENG
15:25
30-09-2020

DATA DRIVEN

Datadriven framework:
---
passing testdata to a testcase through any of the given sources
eg:excel,xml,csv(comma separated file)

Excel: extentions/ formats:

.xls:
---
row-2^16
column-2^8

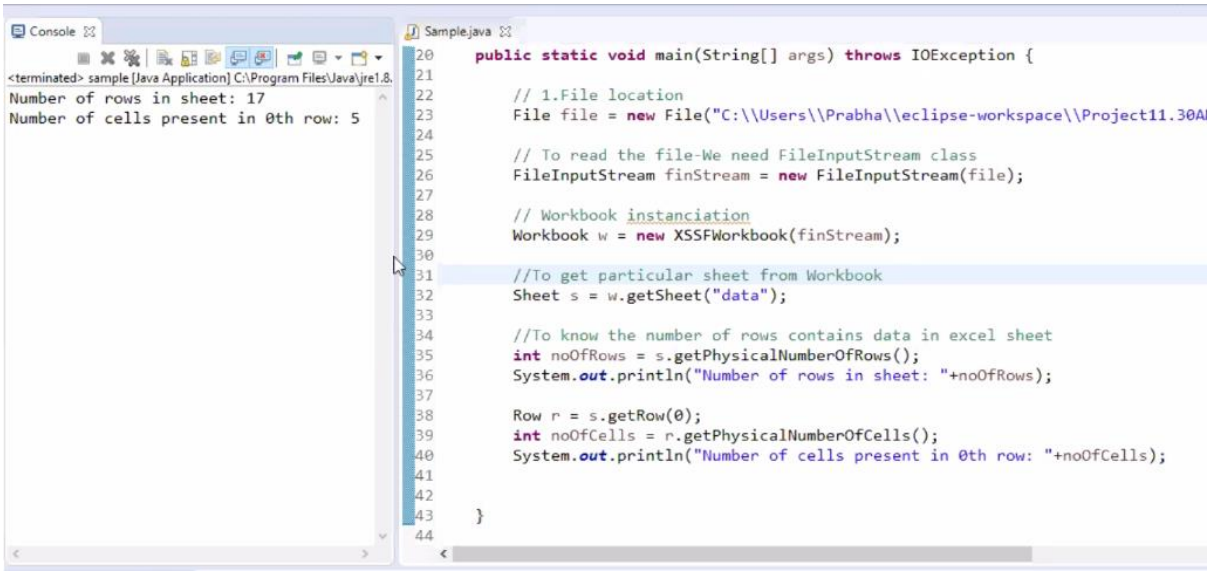
.xlsx:
---
row-2^20
column-2^14
MACROS------supported which is used to automate some tasks in excel

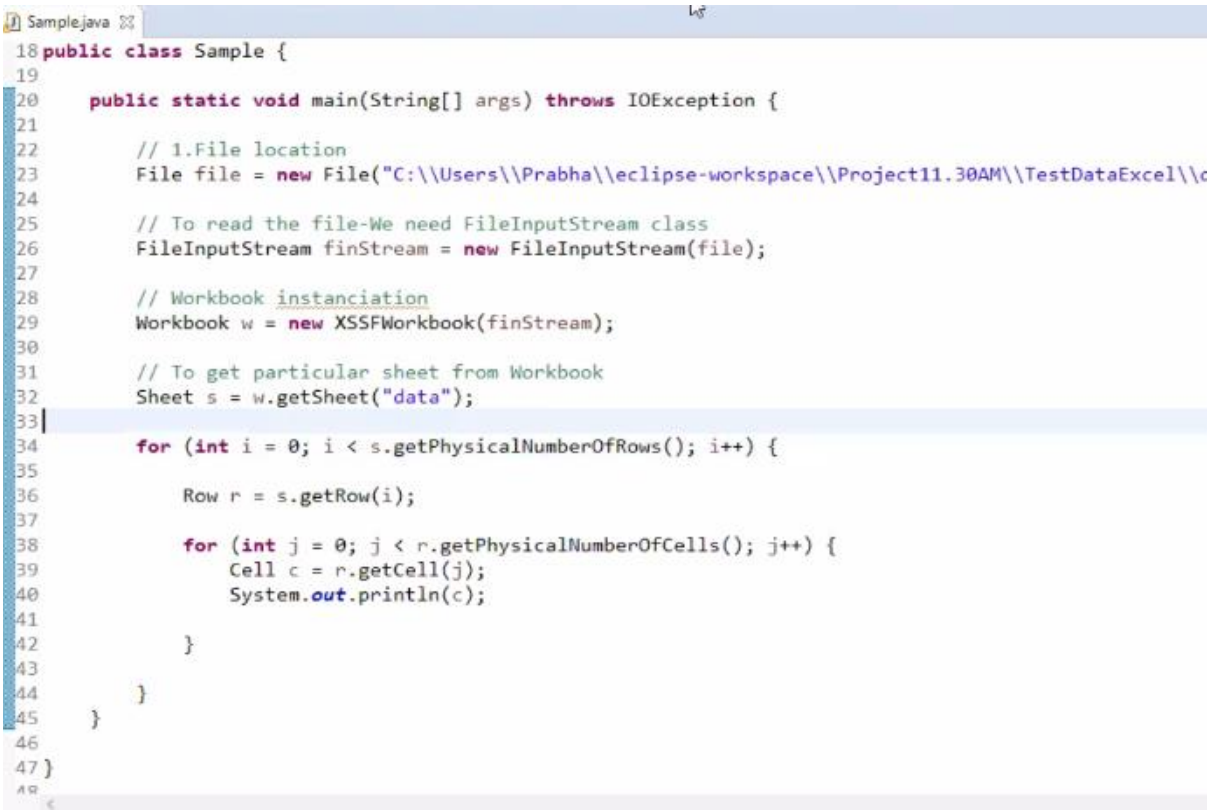
Maven dependencies:

JXL-can read .xls file only
POI- can read both .xls and .xlsx file

```java
public class Sample {

    public static void main(String[] args) throws IOException {

        // 1.File location
        File file = new File("C:\\Users\\Prabha\\eclipse-workspace\\Project11.30AM\\TestDataExcel\\data.xlsx");

        // To read the file-We need FileInputStream class
        FileInputStream finStream = new FileInputStream(file);

        // Workbook instanciation
        Workbook w = new XSSFWorkbook(finStream);

        //To get particular sheet from Workbook
        Sheet s = w.getSheet("data");

        //To get particular row from given sheet
        Row r = s.getRow(1);

        //To get particular cell from given row
        Cell c = r.getCell(0);

        System.out.println(c);

    }
```

```java
public class Sample {

    public static void main(String[] args) throws IOException {

        // 1.File location
        File file = new File("C:\\Users\\Prabha\\eclipse-workspace\\Project11.30AM\\TestDataExcel\\data.xlsx");

        // To read the file-We need FileInputStream class
        FileInputStream finStream = new FileInputStream(file);

        // Workbook instanciation
        Workbook w = new XSSFWorkbook(finStream);

        //To get particular sheet from Workbook
        Sheet s = w.getSheet("data");

        //To get particular row from given sheet
        Row r = s.getRow(1);

        //To get particular cell from given row
        Cell c = r.getCell(0);

        System.out.println(c);

    }
```





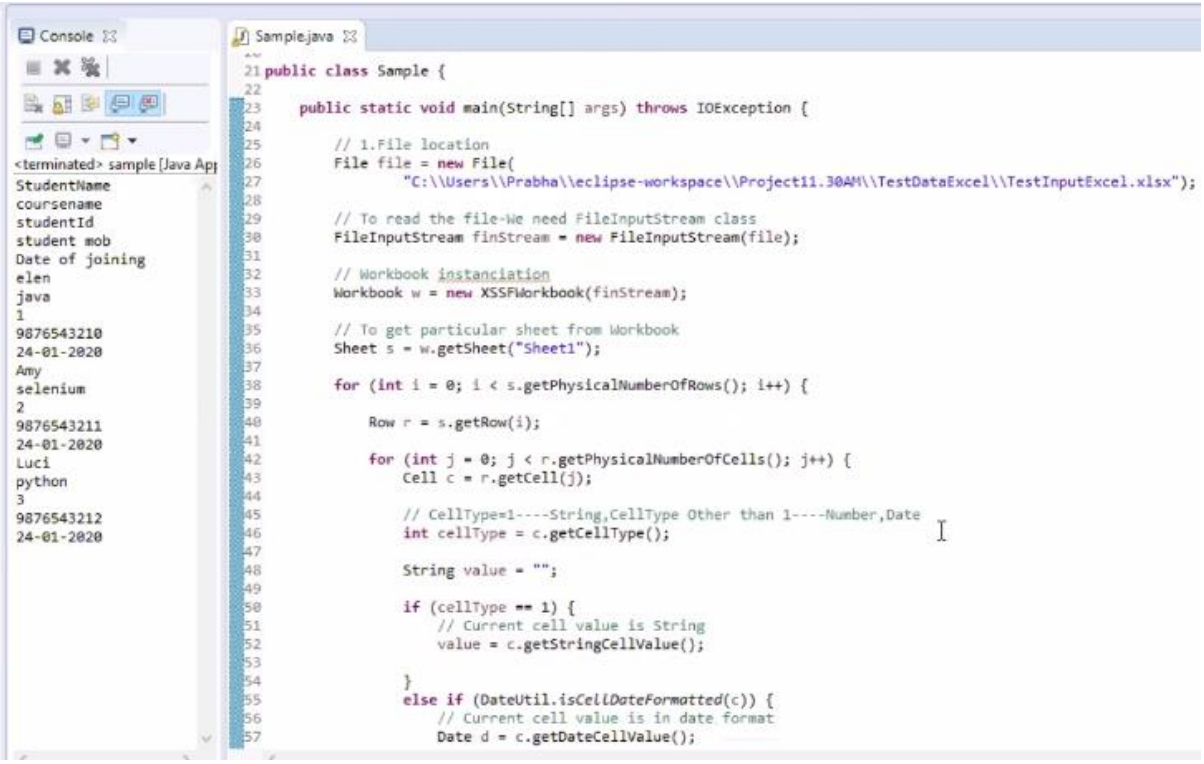
Maven dependency we are going to use in our project:
poi-ooxml 3.8beta4

To read .xls file:

HSSFWorkbook----------Class

To read .xlsx file:

XSSFWorkbook----------Class



To read a file:

FileInputStream

To write into a file:

FileOutputStream

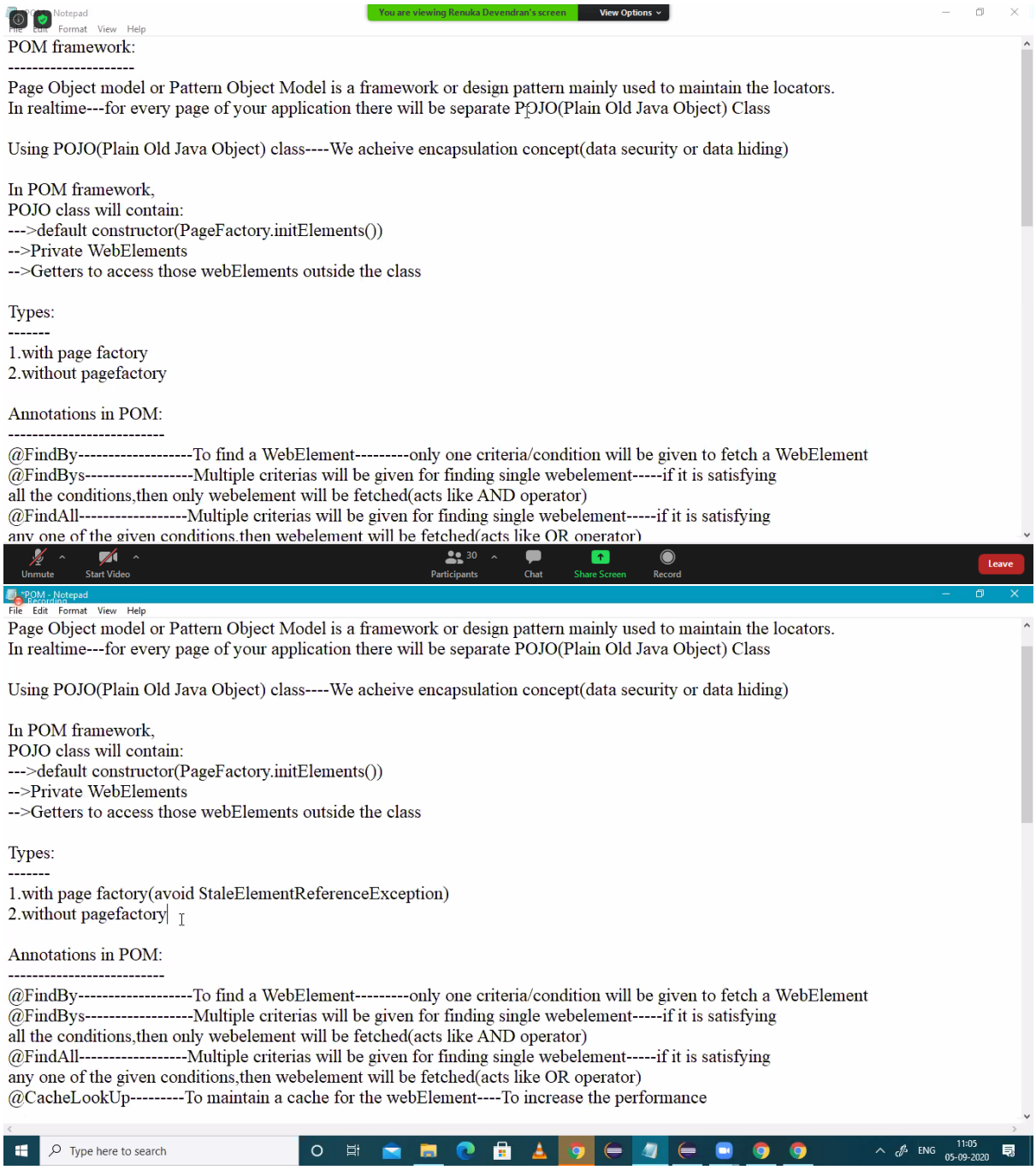
Create excel manually:
---
create a folder under your project---right click the folder--->click new---->File----->give filename.xlsx----->give finish

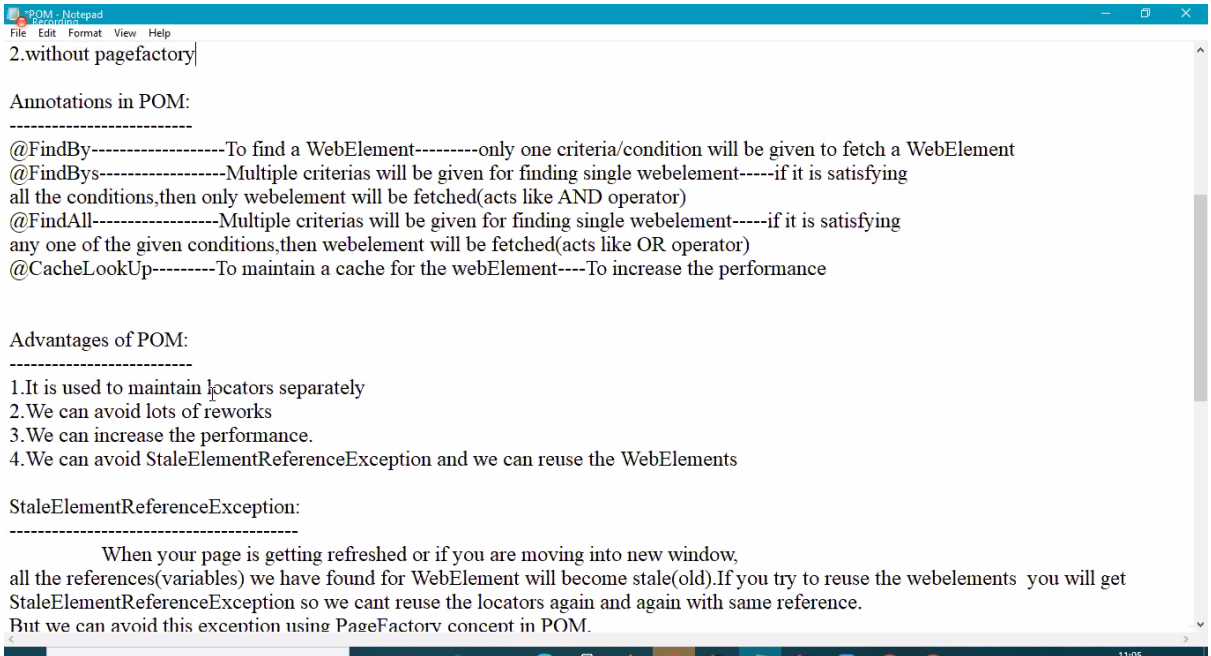
getPhysicalNumberOfRows()----------to get number rows present in workbook
getPhysicalNumberOfCells()----------to get number cells present in particular row

```java
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;

import org.apache.poi.ss.usermodel.Cell;
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;

public class Sample {

    // File write

    public static void main(String[] args) throws IOException {

        // File path=======Folderpath\filename.fileextension
        File file = new File("C:\\Users\\Prabha\\eclipse-workspace\\Project11.30AM\\TestDataExcel\\InputData.xlsx");

        Workbook w = new XSSFWorkbook();

        Sheet s = w.createSheet("Input");

        Row r = s.createRow(0);

        Cell c = r.createCell(0);

        c.setCellValue("Java");

        FileOutputStream fout = new FileOutputStream(file);

        w.write(fout);

        System.out.println("Done successfully");

    }

}
```

```java
    private static void excelWrite(int rowNo, int cellNo, String data) throws IOException {

        // File path=======Folderpath\filename.fileextension
        File file = new File("C:\\Users\\Prabha\\eclipse-workspace\\Project11.30AM\\TestDataExcel\\InputData.xlsx");

        FileInputStream fin = new FileInputStream(file);

        Workbook w = new XSSFWorkbook(fin);

        Sheet s = w.getSheet("Input");

        Row r = s.getRow(rowNo);

        Cell c = r.createCell(cellNo);

        c.setCellValue(data);

        FileOutputStream fout = new FileOutputStream(file);

        w.write(fout);
    }
```

POM FRAMEWORK





*POM - Notepad
File Edit Format View Help

2.without pagefactory

Annotations in POM:

@FindBy----------To find a WebElement---------only one criteria/condition will be given to fetch a WebElement
@FindBys----------Multiple criterias will be given for finding single webelement-----if it is satisfying
all the conditions,then only webelement will be fetched(acts like AND operator)
@FindAll----------Multiple criterias will be given for finding single webelement-----if it is satisfying
any one of the given conditions,then webelement will be fetched(acts like OR operator)
@CacheLookUp---------To maintain a cache for the webElement----To increase the performance

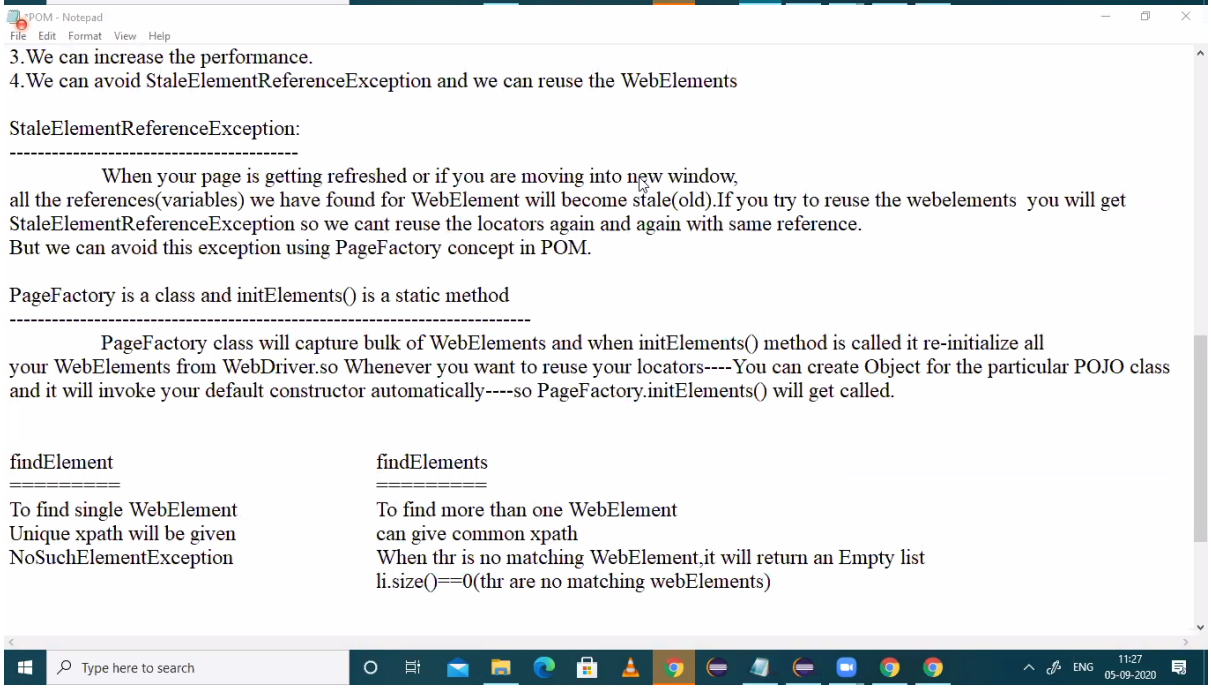
Advantages of POM:

1.It is used to maintain locators separately
2.We can avoid lots of reworks
3.We can increase the performance.
4.We can avoid StaleElementReferenceException and we can reuse the WebElements

StaleElementReferenceException:

When your page is getting refreshed or if you are moving into new window,
all the references(variables) we have found for WebElement will become stale(old).If you try to reuse the webelements you will get
StaleElementReferenceException so we cant reuse the locators again and again with same reference.
But we can avoid this exception using PageFactory concept in POM.

Type here to search ENG 11:05 05-09-2020



*POM - Notepad
File Edit Format View Help

3.We can increase the performance.
4.We can avoid StaleElementReferenceException and we can reuse the WebElements

StaleElementReferenceException:

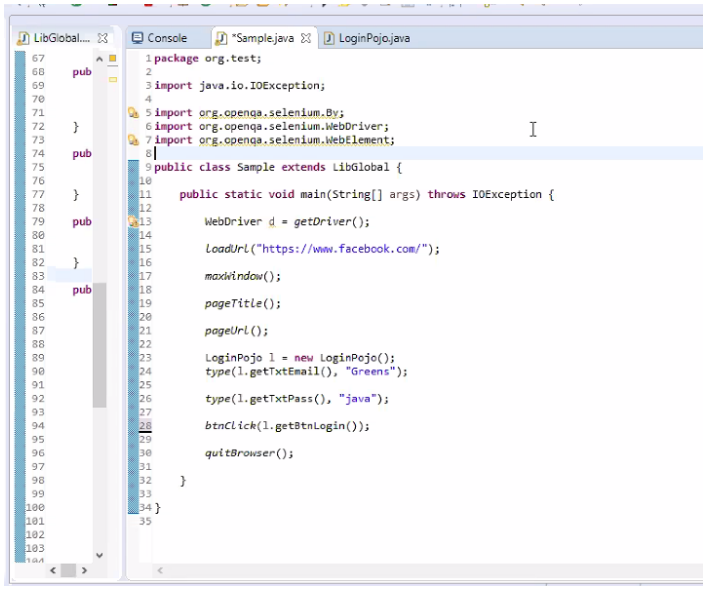
When your page is getting refreshed or if you are moving into new window,
all the references(variables) we have found for WebElement will become stale(old).If you try to reuse the webelements you will get
StaleElementReferenceException so we cant reuse the locators again and again with same reference.
But we can avoid this exception using PageFactory concept in POM.

PageFactory is a class and initElements() is a static method

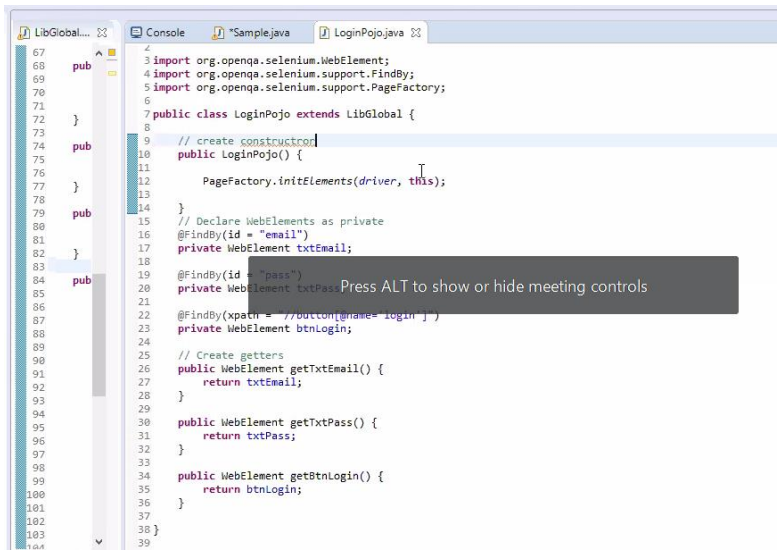
PageFactory class will capture bulk of WebElements and when initElements() method is called it re-initialize all
your WebElements from WebDriver.so Whenever you want to reuse your locators----You can create Object for the particular POJO class
and it will invoke your default constructor automatically----so PageFactory.initElements() will get called.

findElement	findElements
==========	==========
To find single WebElement	To find more than one WebElement
Unique xpath will be given	can give common xpath
NoSuchElementException	When thr is no matching WebElement,it will return an Empty list
	li.size()==0(thr are no matching webElements)

Type here to search ENG 11:27 05-09-2020

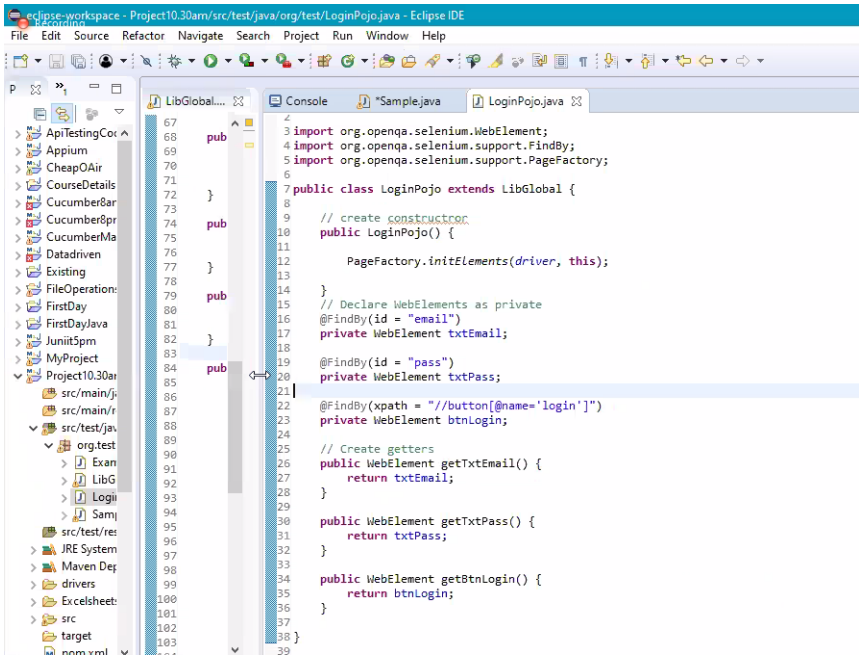


```java
package org.test;

import java.io.IOException;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

public class Sample extends LibGlobal {

    public static void main(String[] args) throws IOException {

        WebDriver d = getDriver();

        loadUrl("https://www.facebook.com/");

        maxWindow();

        pageTitle();

        pageUrl();

        LoginPojo l = new LoginPojo();
        type(l.getTxtEmail(), "Greens");

        type(l.getTxtPass(), "java");

        btnClick(l.getBtnLogin());

        quitBrowser();

    }

}
```



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

public class LoginPojo extends LibGlobal {

    // create constructror
    public LoginPojo() {

        PageFactory.initElements(driver, this);

    }
    // Declare WebElements as private
    @FindBy(id = "email")
    private WebElement txtEmail;

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

    @FindBy(xpath = "//button[@name='login']")
    private WebElement btnLogin;

    // Create getters
    public WebElement getTxtEmail() {
        return txtEmail;
    }

    public WebElement getTxtPass() {
        return txtPass;
    }

    public WebElement getBtnLogin() {
        return btnLogin;
    }

}
```

Press ALT to show or hide meeting controls



POM - Notepad

File Edit Format View Help

any one of the given conditions,then webelement will be fetched(acts like OR operator)
@CacheLookUp---------To maintain a cache for the webElement----To increase the performance

Advantages of POM:

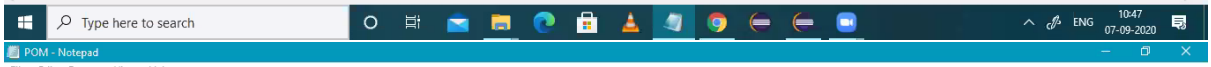
1.It is used to maintain locators separately
2.We can avoid lots of reworks
3.We can increase the performance.
4.We can avoid StaleElementReferenceException and we can reuse the WebElements

StaleElementReferenceException:

When your page is getting refreshed or if you are moving into new window,
all the references(variables) we have found for WebElement will become stale(old).If you try to reuse the webelements you will get StaleElementReferenceException so we cant reuse the locators again and again with same reference.
But we can avoid this exception using PageFactory concept in POM.

PageFactory is a class and initElements() is a static method

PageFactory class will capture bulk of WebElements and when initElements() method is called it re-initialize all your WebElements from WebDriver.so Whenever you want to reuse your locators----You can create Object for the particular POJO class and it will invoke your default constructor automatically----so PageFactory.initElements() will get called.



POM - Notepad

File Edit Format View Help

POM framework:

Page Object model or Pattern Object Model is a framework or design pattern mainly used to maintain the locators.
In realtime---for every page of your application there will be separate POJO(Plain Old Java Object) Class

Using POJO(Plain Old Java Object) class----We acheive encapsulation concept(data security or data hiding)

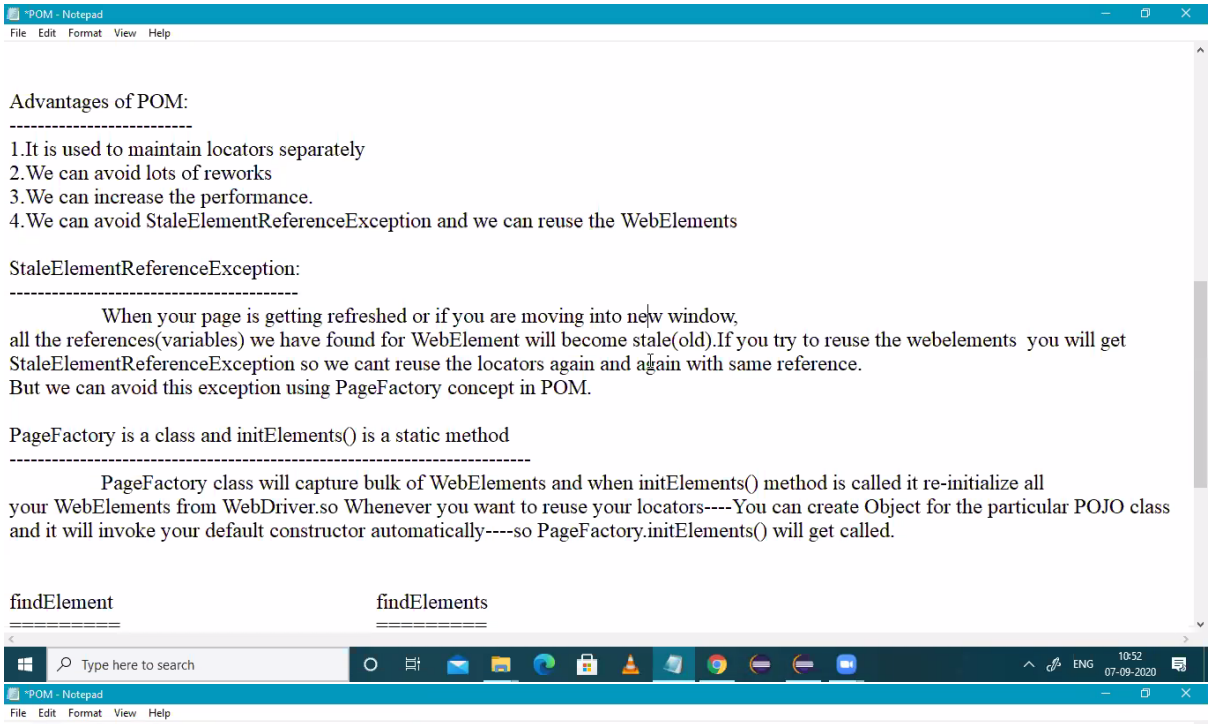
In POM framework,
POJO class will contain:
--->default constructor(PageFactory.initElements())
-->Private WebElements
-->Getters to access those webElements outside the class

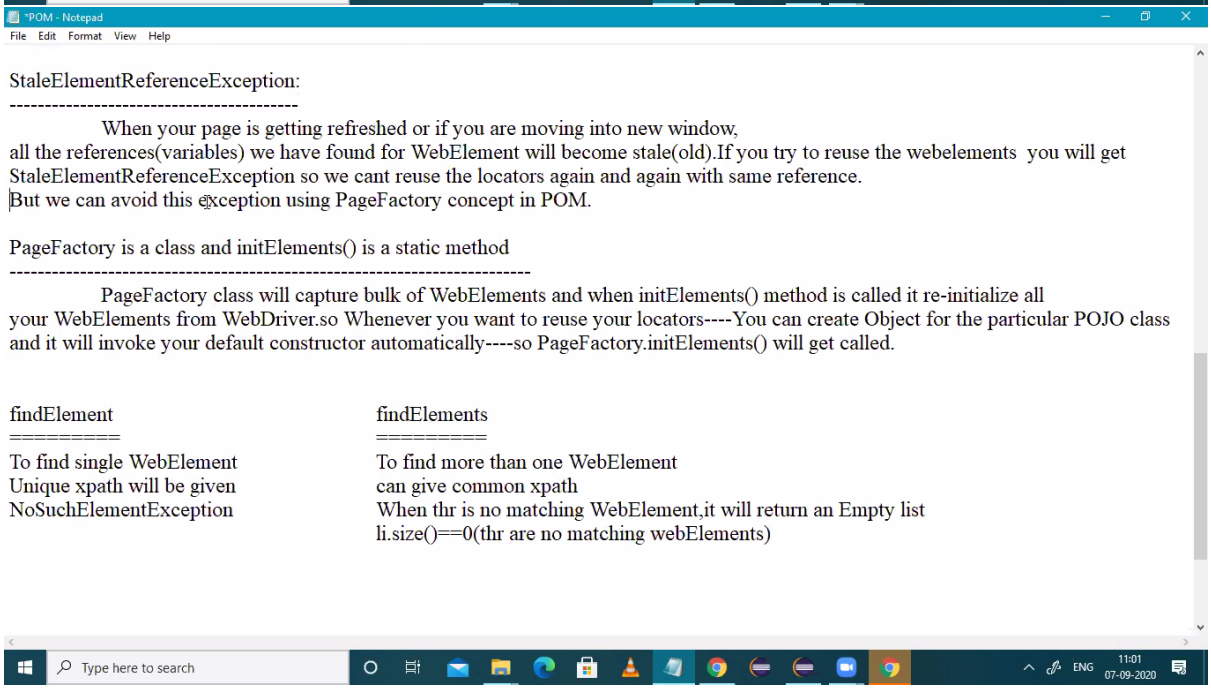
Types:

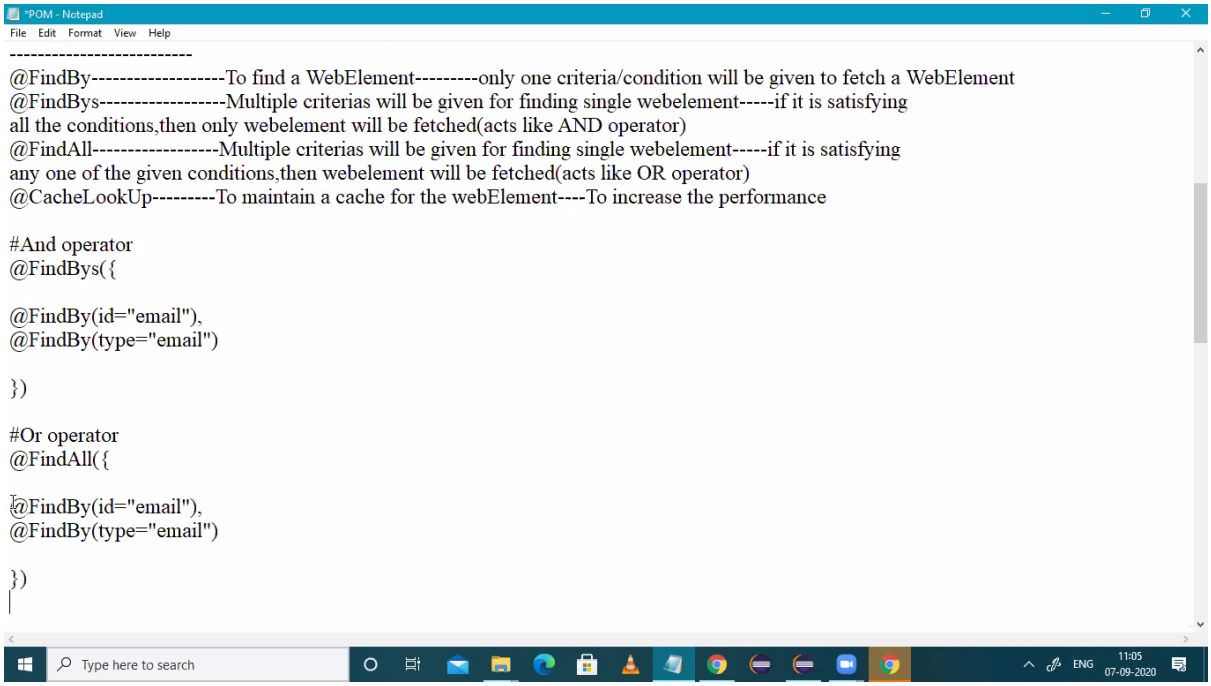
1.with page factory(avoid StaleElementReferenceException)
2.without pagefactory

Annotations in POM:

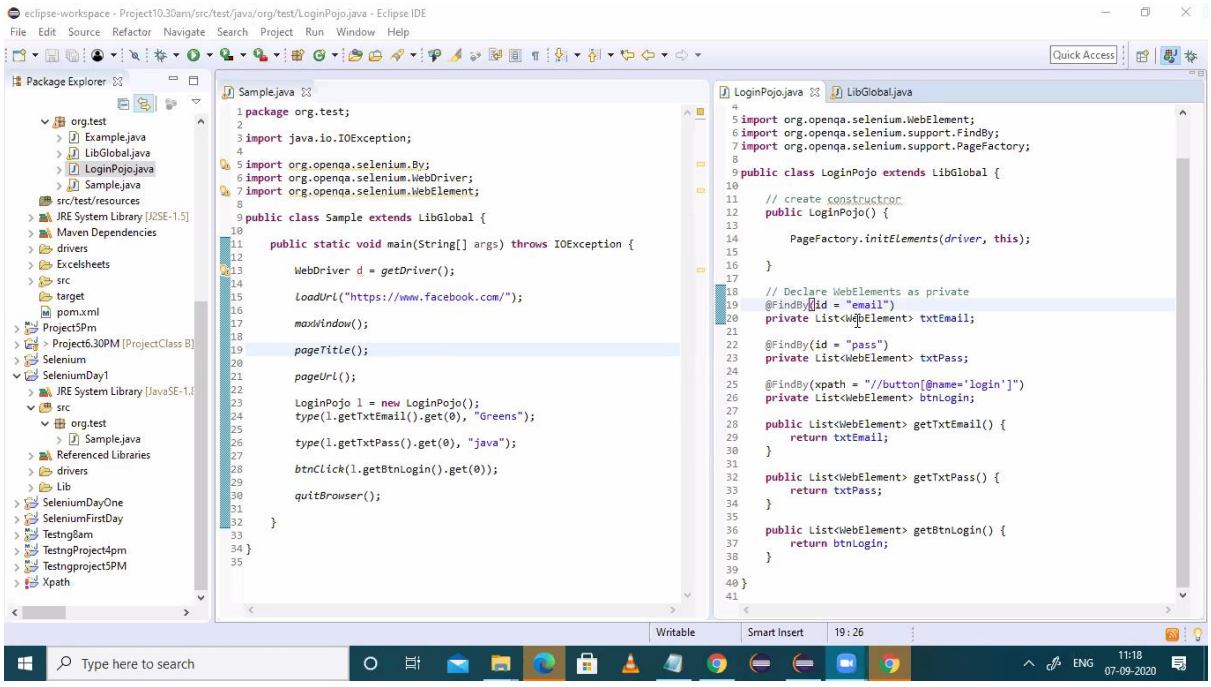
@FindBy----------To find a WebElement---------only one criteria/condition will be given to fetch a WebElement
@FindBys----------Multiple criterias will be given for finding single webelement-----if it is satisfying all the conditions,then only webelement will be fetched(acts like AND operator)
@FindAll----------Multiple criterias will be given for finding single webelement-----if it is satisfying any one of the given conditions,then webelement will be fetched(acts like OR operator)

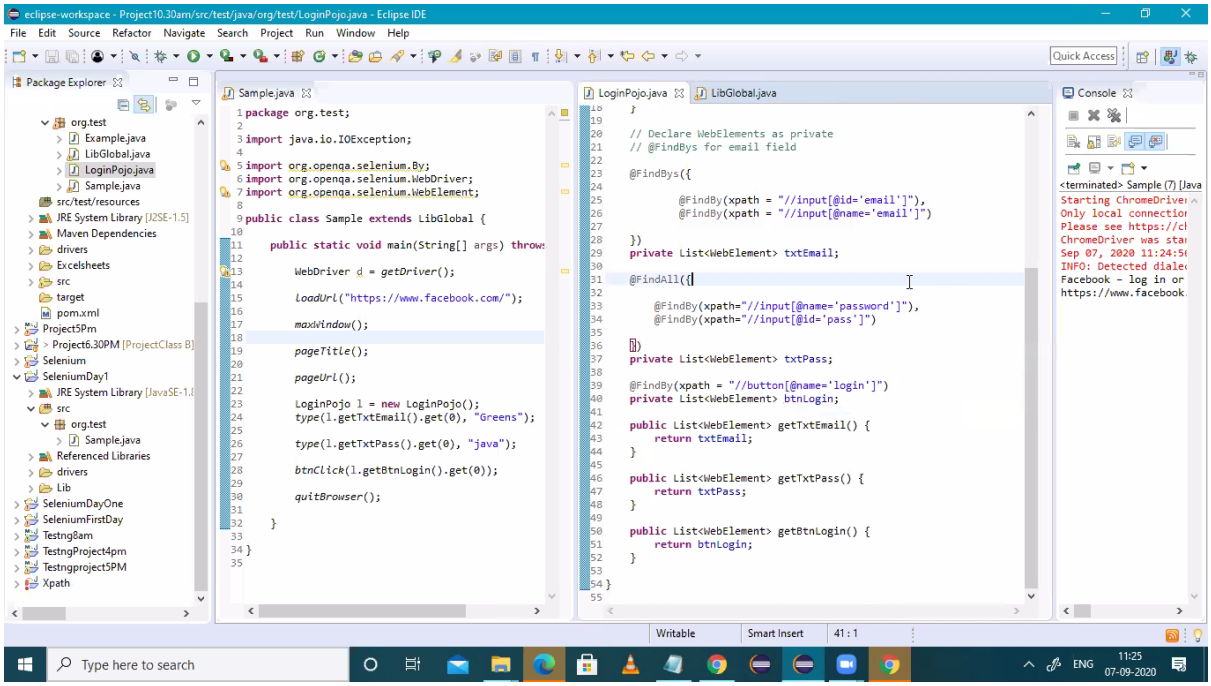

*POM - Notepad
File Edit Format View Help
Advantages of POM:
1.It is used to maintain locators separately
2.We can avoid lots of reworks
3.We can increase the performance.
4.We can avoid StaleElementReferenceException and we can reuse the WebElements
StaleElementReferenceException:
When your page is getting refreshed or if you are moving into new window,
all the references(variables) we have found for WebElement will become stale(old).If you try to reuse the webelements you will get
StaleElementReferenceException so we cant reuse the locators again and again with same reference.
But we can avoid this exception using PageFactory concept in POM.
PageFactory is a class and initElements() is a static method
PageFactory class will capture bulk of WebElements and when initElements() method is called it re-initialize all
your WebElements from WebDriver.so Whenever you want to reuse your locators----You can create Object for the particular POJO class
and it will invoke your default constructor automatically----so PageFactory.initElements() will get called.
findElement
findElements
Type here to search
ENG
10:52
07-09-2020


*POM - Notepad
File Edit Format View Help
StaleElementReferenceException:
When your page is getting refreshed or if you are moving into new window,
all the references(variables) we have found for WebElement will become stale(old).If you try to reuse the webelements you will get
StaleElementReferenceException so we cant reuse the locators again and again with same reference.
But we can avoid this exception using PageFactory concept in POM.
PageFactory is a class and initElements() is a static method
PageFactory class will capture bulk of WebElements and when initElements() method is called it re-initialize all
your WebElements from WebDriver.so Whenever you want to reuse your locators----You can create Object for the particular POJO class
and it will invoke your default constructor automatically----so PageFactory.initElements() will get called.
findElement
To find single WebElement
Unique xpath will be given
NoSuchElementException
findElements
To find more than one WebElement
can give common xpath
When thr is no matching WebElement,it will return an Empty list
li.size()==0(thr are no matching webElements)
Type here to search
ENG
11:01
07-09-2020


*POM - Notepad
File Edit Format View Help

@FindBy----------To find a WebElement---------only one criteria/condition will be given to fetch a WebElement
@FindBys----------Multiple criterias will be given for finding single webelement-----if it is satisfying
all the conditions,then only webelement will be fetched(acts like AND operator)
@FindAll----------Multiple criterias will be given for finding single webelement-----if it is satisfying
any one of the given conditions,then webelement will be fetched(acts like OR operator)
@CacheLookUp---------To maintain a cache for the webElement----To increase the performance
#And operator
@FindBys({
@FindBy(id="email"),
@FindBy(type="email")
})
#Or operator
@FindAll({
@FindBy(id="email"),
@FindBy(type="email")
})
Type here to search
ENG
11:05
07-09-2020


eclipse-workspace - Project10.30am/src/test/java/org/test/LoginPojo.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
org.test
Example.java
LibGlobal.java
LoginPojo.java
Sample.java
src/test/resources
JRE System Library [J2SE-1.5]
Maven Dependencies
drivers
Excelsheets
src
target
pom.xml
Project5Pm
Project6.30PM [ProjectClass B]
Selenium
SeleniumDay1
JRE System Library [JavaSE-1.8]
src
org.test
Sample.java
Referenced Libraries
drivers
Lib
SeleniumDayOne
SeleniumFirstDay
Testng8am
TestngProject4pm
Testngproject5PM
Xpath
Sample.java
1 package org.test;
2
3 import java.io.IOException;
4
5 import org.openqa.selenium.By;
6 import org.openqa.selenium.WebDriver;
7 import org.openqa.selenium.WebElement;
8
9 public class Sample extends LibGlobal {
10
11 public static void main(String[] args) throws IOException {
12
13 WebDriver d = getDriver();
14
15 loadUrl("https://www.facebook.com/");
16
17 maxWindow();
18
19 pageTitle();
20
21 pageUrl();
22
23 LoginPojo l = new LoginPojo();
24 type(l.getTxtEmail().get(0), "Greens");
25
26 type(l.getTxtPass().get(0), "java");
27
28 btnClick(l.getBtnLogin().get(0));
29
30 quitBrowser();
31
32 }
33
34 }
35
LoginPojo.java
LibGlobal.java
4
5 import org.openqa.selenium.WebElement;
6 import org.openqa.selenium.support.FindBy;
7 import org.openqa.selenium.support.PageFactory;
8
9 public class LoginPojo extends LibGlobal {
10
11 // create constructror
12 public LoginPojo() {
13
14 PageFactory.initElements(driver, this);
15
16 }
17
18 // Declare WebElements as private
19 @FindBy(id = "email")
20 private List<WebElement> txtEmail;
21
22 @FindBy(id = "pass")
23 private List<WebElement> txtPass;
24
25 @FindBy(xpath = "//button[@name='login']")
26 private List<WebElement> btnLogin;
27
28 public List<WebElement> getTxtEmail() {
29 return txtEmail;
30 }
31
32 public List<WebElement> getTxtPass() {
33 return txtPass;
34 }
35
36 public List<WebElement> getBtnLogin() {
37 return btnLogin;
38 }
39
40 }
41
Writable
Smart Insert
19 : 26
Type here to search
ENG
11:18
07-09-2020


eclipse-workspace - Project10.30am/src/test/java/org/test/LoginPojo.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
org.test
Example.java
LibGlobal.java
LoginPojo.java
Sample.java
src/test/resources
JRE System Library [J2SE-1.5]
Maven Dependencies
drivers
Excelsheets
src
target
pom.xml
Project5Pm
Project6.30PM [ProjectClass B]
Selenium
SeleniumDay1
JRE System Library [JavaSE-1.8]
src
org.test
Sample.java
Referenced Libraries
drivers
Lib
SeleniumDayOne
SeleniumFirstDay
Testng8am
TestngProject4pm
Testngproject5PM
Xpath
Sample.java
1 package org.test;
2
3 import java.io.IOException;
4
5 import org.openqa.selenium.By;
6 import org.openqa.selenium.WebDriver;
7 import org.openqa.selenium.WebElement;
8
9 public class Sample extends LibGlobal {
10
11 public static void main(String[] args) throws
12
13 WebDriver d = getDriver();
14
15 loadUrl("https://www.facebook.com/");
16
17 maxWindow();
18
19 pageTitle();
20
21 pageUrl();
22
23 LoginPojo l = new LoginPojo();
24 type(l.getTxtEmail().get(0), "Greens");
25
26 type(l.getTxtPass().get(0), "java");
27
28 btnClick(l.getBtnLogin().get(0));
29
30 quitBrowser();
31
32 }
33
34 }
35
LoginPojo.java
LibGlobal.java
19
20 // Declare WebElements as private
21 // @FindBys for email field
22
23 @FindBys({
24
25 @FindBy(xpath = "//input[@id='email']"),
26 @FindBy(xpath = "//input[@name='email']")
27
28 })
29 private List<WebElement> txtEmail;
30
31 @FindAll({
32
33 @FindBy(xpath="//input[@name='password']"),
34 @FindBy(xpath="//input[@id='pass']")
35
36 })
37 private List<WebElement> txtPass;
38
39 @FindBy(xpath = "//button[@name='login']")
40 private List<WebElement> btnLogin;
41
42 public List<WebElement> getTxtEmail() {
43 return txtEmail;
44 }
45
46 public List<WebElement> getTxtPass() {
47 return txtPass;
48 }
49
50 public List<WebElement> getBtnLogin() {
51 return btnLogin;
52 }
53
54 }
55
Console
<terminated> Sample (7) [Java
Starting ChromeDriver
Only local connection
Please see https://ch
ChromeDriver was star
Sep 07, 2020 11:24:56
INFO: Detected dialec
Facebook - log in or
https://www.facebook.
Writable
Smart Insert
41 : 1
Type here to search
ENG
11:25
07-09-2020

JUNIT

Junit - Notepad

File Edit Format View Help

Junit framework:
---
mainly used by developers fot Unit testing

Jar/Dependency required:
---
junit 4.12
Hamcrest-all 1.3(supporting jar)--------To avoid NoClassDefinitionFound

Annotations:

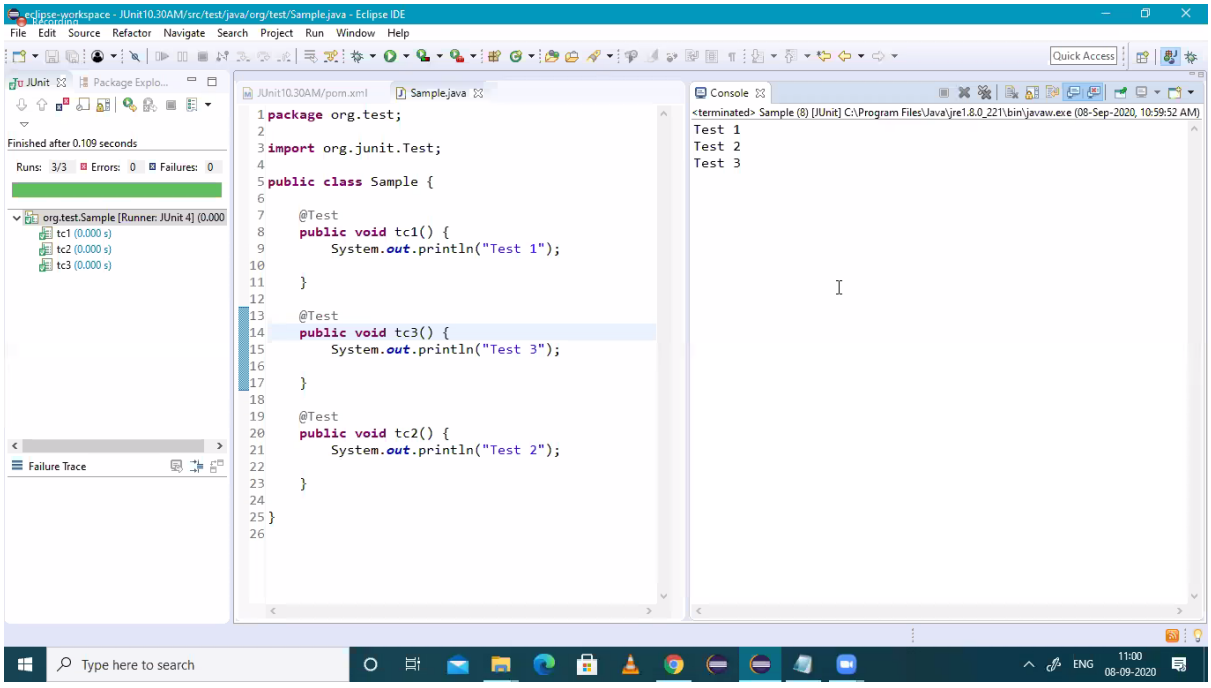
@BeforeClass-------executes before the execution of all the @Test
(we can mention the browser launch steps)

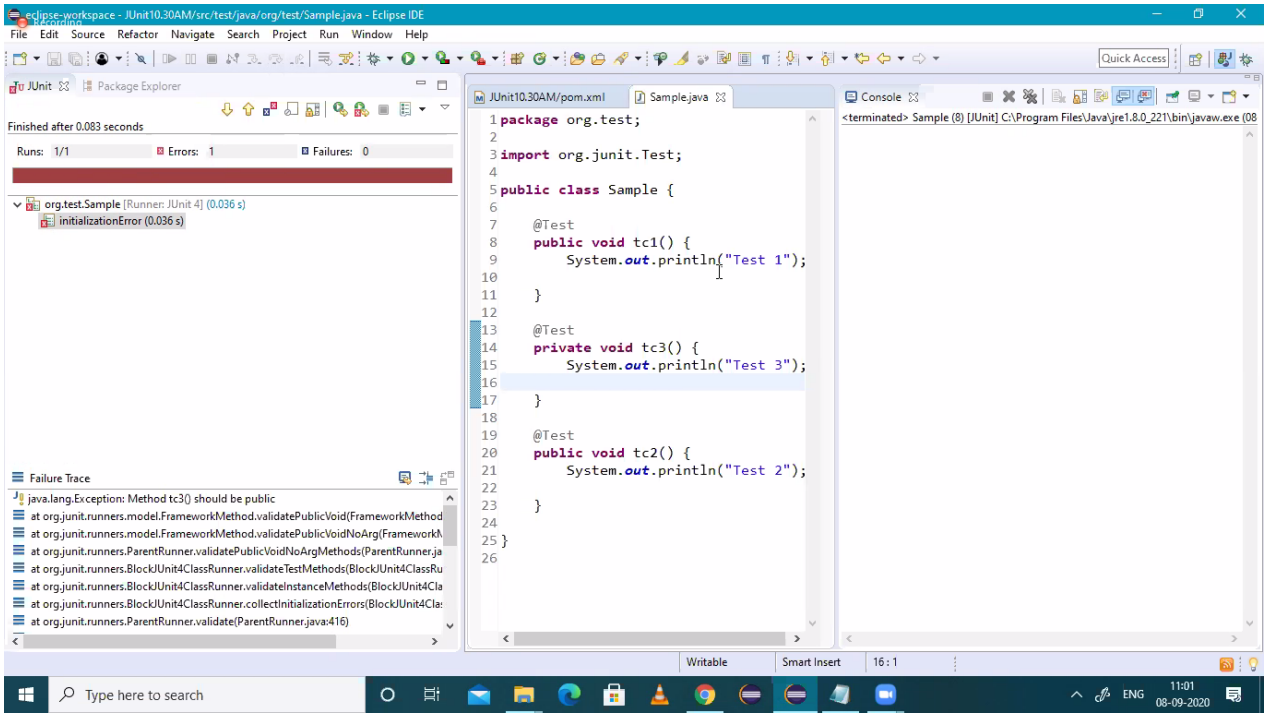
@Before----------executes before the execution of each @Test
(We can calculate the start time of execution of @Test)

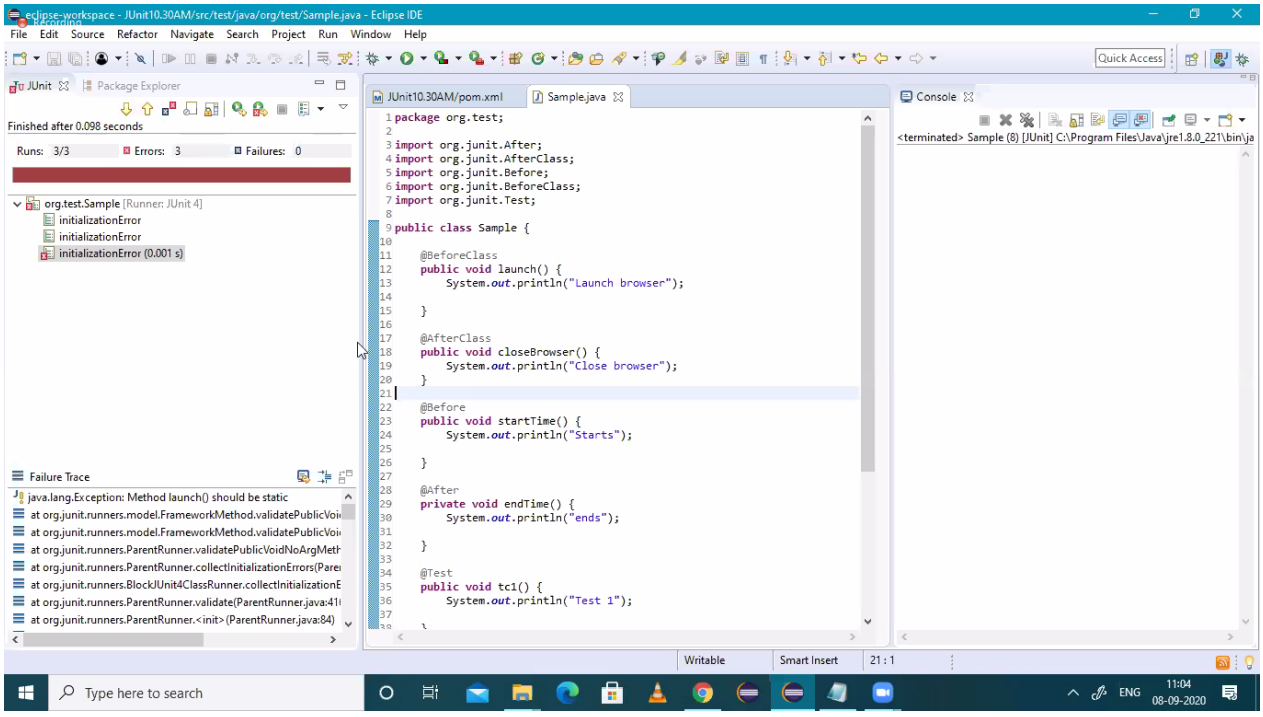
@Test----------used to write the business logic

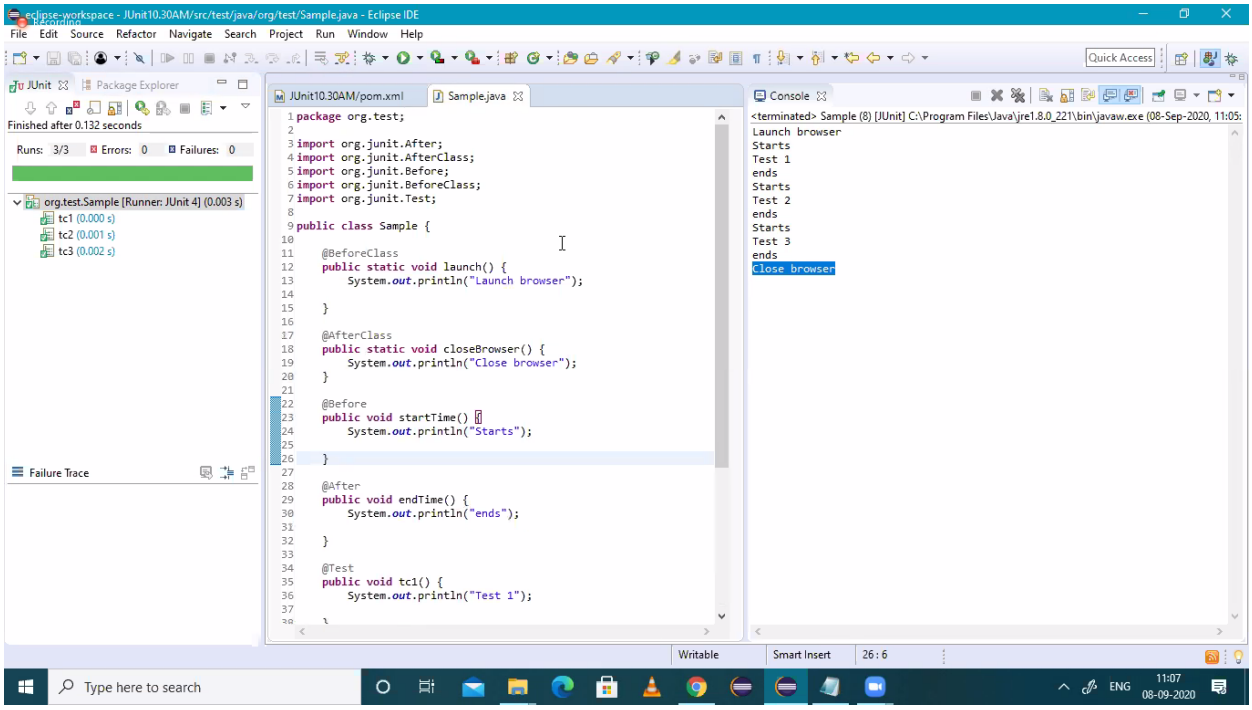
@After----------executes after the execution of each @Test
(We can calculate the end time of execution of @Test)

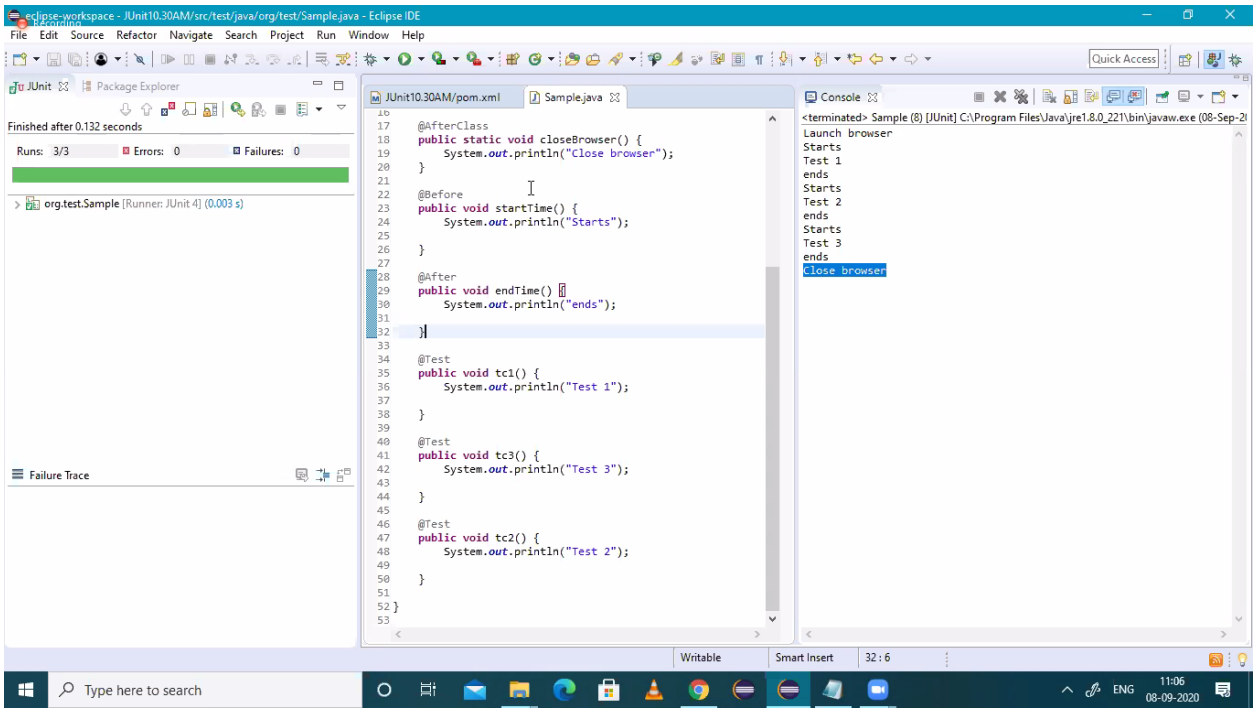
@AfterClass-------executes after the execution of all the @Test
(We can mention the steps to close the browser)

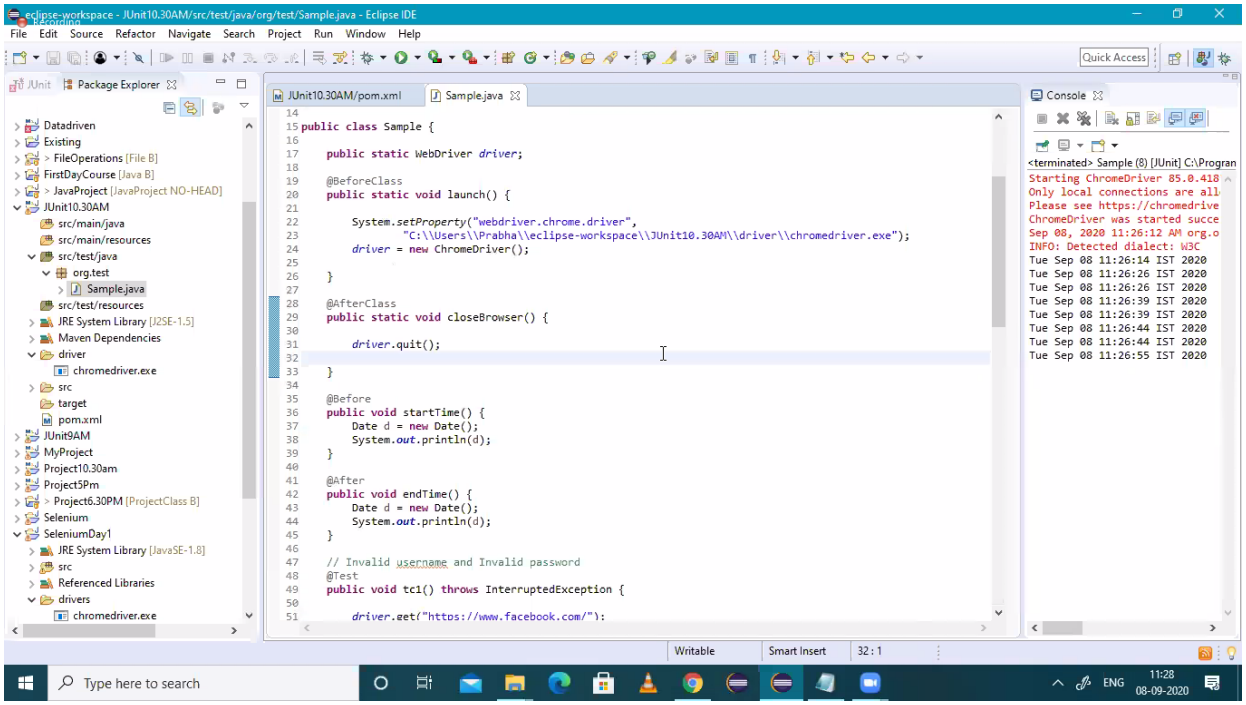


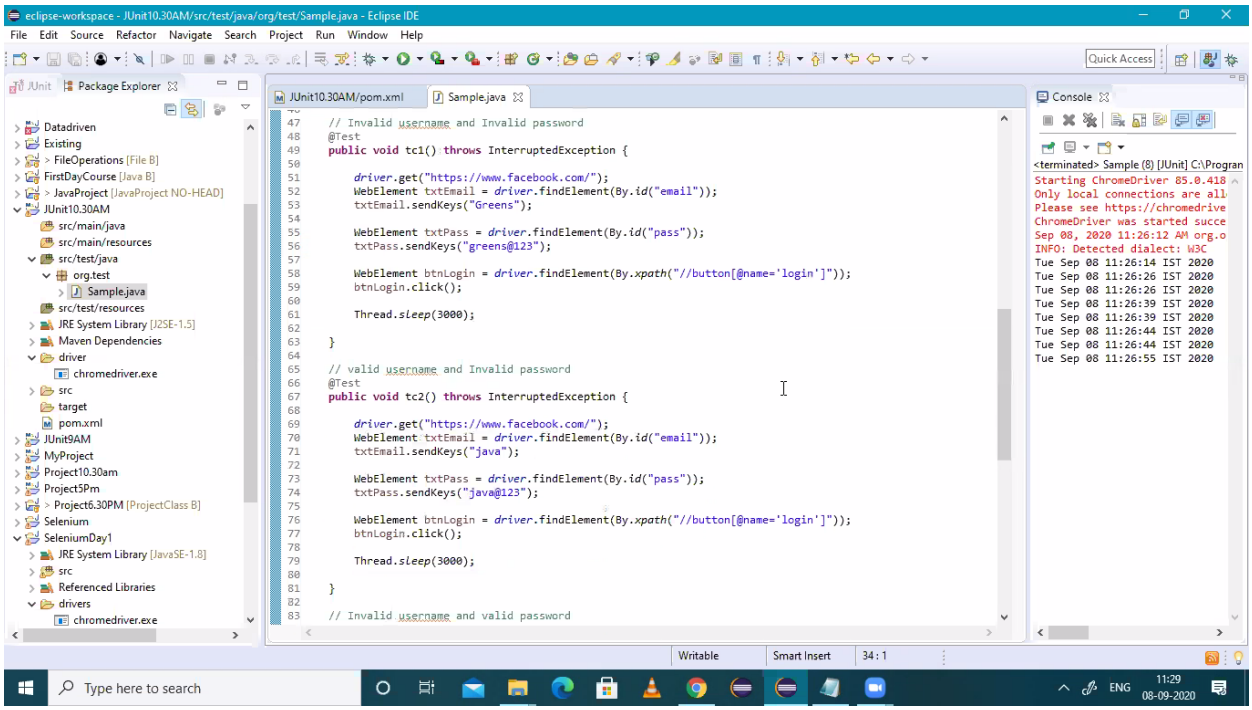

eclipse-workspace - JUnit10.30AM/src/test/java/org/test/Sample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
JUnit
Package Explorer
Finished after 0.083 seconds
Runs: 1/1
Errors: 1
Failures: 0
org.test.Sample [Runner: JUnit 4] (0.036 s)
initializationError (0.036 s)
Failure Trace
java.lang.Exception: Method tc3() should be public
JUnit10.30AM/pom.xml
Sample.java
1 package org.test;
3 import org.junit.Test;
5 public class Sample {
7 @Test
8 public void tc1() {
9 System.out.println("Test 1");
11 }
13 @Test
14 private void tc3() {
15 System.out.println("Test 3");
17 }
19 @Test
20 public void tc2() {
21 System.out.println("Test 2");
23 }
25 }
Console
<terminated> Sample (8) [JUnit] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (08
Writable
Smart Insert
16 : 1
Type here to search
ENG
11:01
08-09-2020

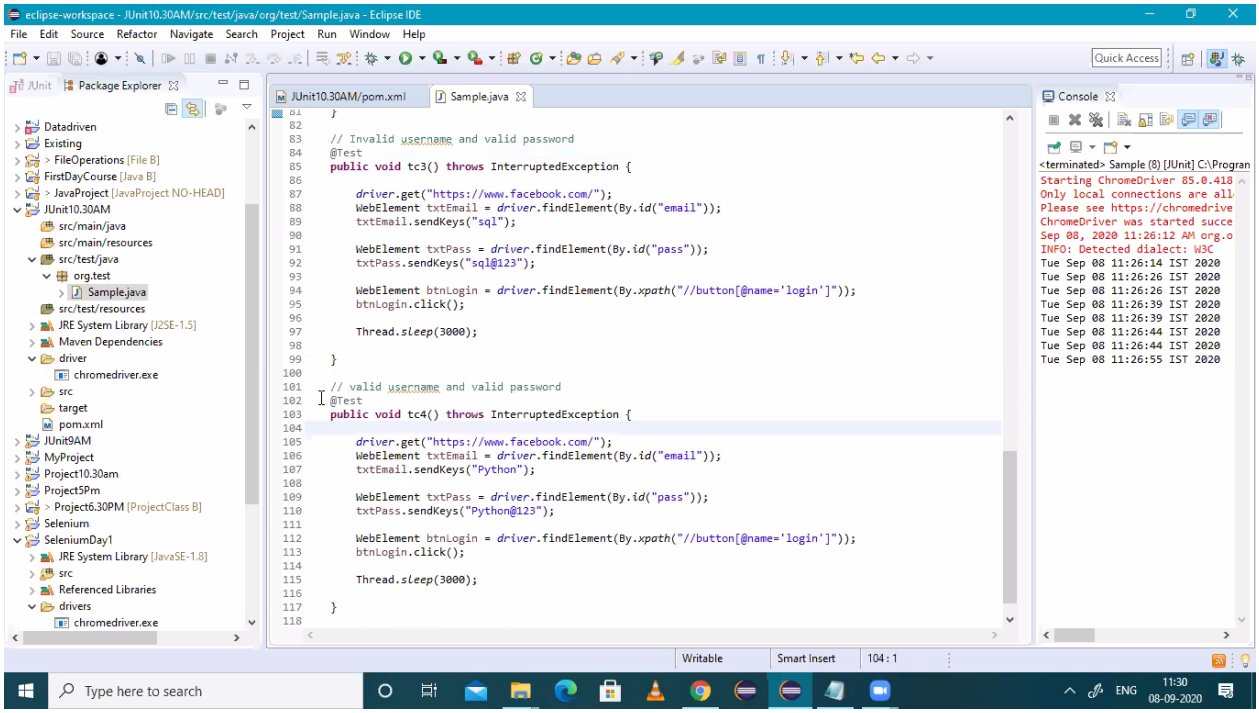

eclipse-workspace - JUnit10.30AM/src/test/java/org/test/Sample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
JUnit
Package Explorer
Finished after 0.098 seconds
Runs: 3/3
Errors: 3
Failures: 0
org.test.Sample [Runner: JUnit 4]
initializationError
initializationError
initializationError (0.001 s)
Failure Trace
java.lang.Exception: Method launch() should be static
JUnit10.30AM/pom.xml
Sample.java
1 package org.test;
3 import org.junit.After;
4 import org.junit.AfterClass;
5 import org.junit.Before;
6 import org.junit.BeforeClass;
7 import org.junit.Test;
9 public class Sample {
11 @BeforeClass
12 public void launch() {
13 System.out.println("Launch browser");
15 }
17 @AfterClass
18 public void closeBrowser() {
19 System.out.println("Close browser");
20 }
22 @Before
23 public void startTime() {
24 System.out.println("Starts");
26 }
28 @After
29 private void endTime() {
30 System.out.println("ends");
32 }
34 @Test
35 public void tc1() {
36 System.out.println("Test 1");
Console
<terminated> Sample (8) [JUnit] C:\Program Files\Java\jre1.8.0_221\bin\ja
Writable
Smart Insert
21 : 1
Type here to search
ENG
11:04
08-09-2020

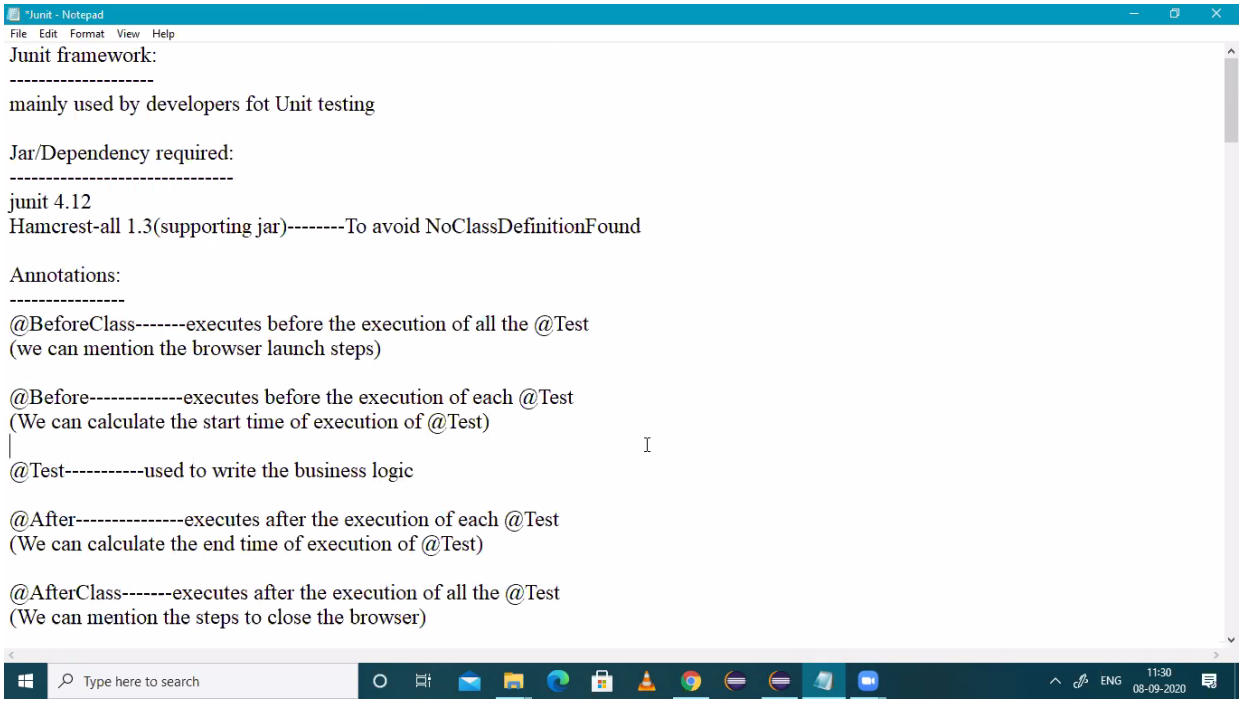

eclipse-workspace - JUnit10.30AM/src/test/java/org/test/Sample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
JUnit
Package Explorer
Finished after 0.132 seconds
Runs: 3/3
Errors: 0
Failures: 0
org.test.Sample [Runner: JUnit 4] (0.003 s)
tc1 (0.000 s)
tc2 (0.001 s)
tc3 (0.002 s)
Failure Trace
JUnit10.30AM/pom.xml
Sample.java
1 package org.test;
2
3 import org.junit.After;
4 import org.junit.AfterClass;
5 import org.junit.Before;
6 import org.junit.BeforeClass;
7 import org.junit.Test;
8
9 public class Sample {
10
11 @BeforeClass
12 public static void launch() {
13 System.out.println("Launch browser");
14
15 }
16
17 @AfterClass
18 public static void closeBrowser() {
19 System.out.println("Close browser");
20 }
21
22 @Before
23 public void startTime() {
24 System.out.println("Starts");
25
26 }
27
28 @After
29 public void endTime() {
30 System.out.println("ends");
31
32 }
33
34 @Test
35 public void tc1() {
36 System.out.println("Test 1");
37
Console
<terminated> Sample (8) [JUnit] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (08-Sep-2020, 11:05:
Launch browser
Starts
Test 1
ends
Starts
Test 2
ends
Starts
Test 3
ends
Close browser
Writable
Smart Insert
26 : 6
Type here to search
ENG
11:07
08-09-2020


eclipse-workspace - JUnit10.30AM/src/test/java/org/test/Sample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
JUnit
Package Explorer
Finished after 0.132 seconds
Runs: 3/3
Errors: 0
Failures: 0
org.test.Sample [Runner: JUnit 4] (0.003 s)
Failure Trace
JUnit10.30AM/pom.xml
Sample.java
17 @AfterClass
18 public static void closeBrowser() {
19 System.out.println("Close browser");
20 }
21
22 @Before
23 public void startTime() {
24 System.out.println("Starts");
25
26 }
27
28 @After
29 public void endTime() {
30 System.out.println("ends");
31
32 }
33
34 @Test
35 public void tc1() {
36 System.out.println("Test 1");
37
38 }
39
40 @Test
41 public void tc3() {
42 System.out.println("Test 3");
43
44 }
45
46 @Test
47 public void tc2() {
48 System.out.println("Test 2");
49
50 }
51
52 }
53
Console
<terminated> Sample (8) [JUnit] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (08-Sep-2
Launch browser
Starts
Test 1
ends
Starts
Test 2
ends
Starts
Test 3
ends
Close browser
Writable
Smart Insert
32 : 6
Type here to search
ENG
11:06
08-09-2020

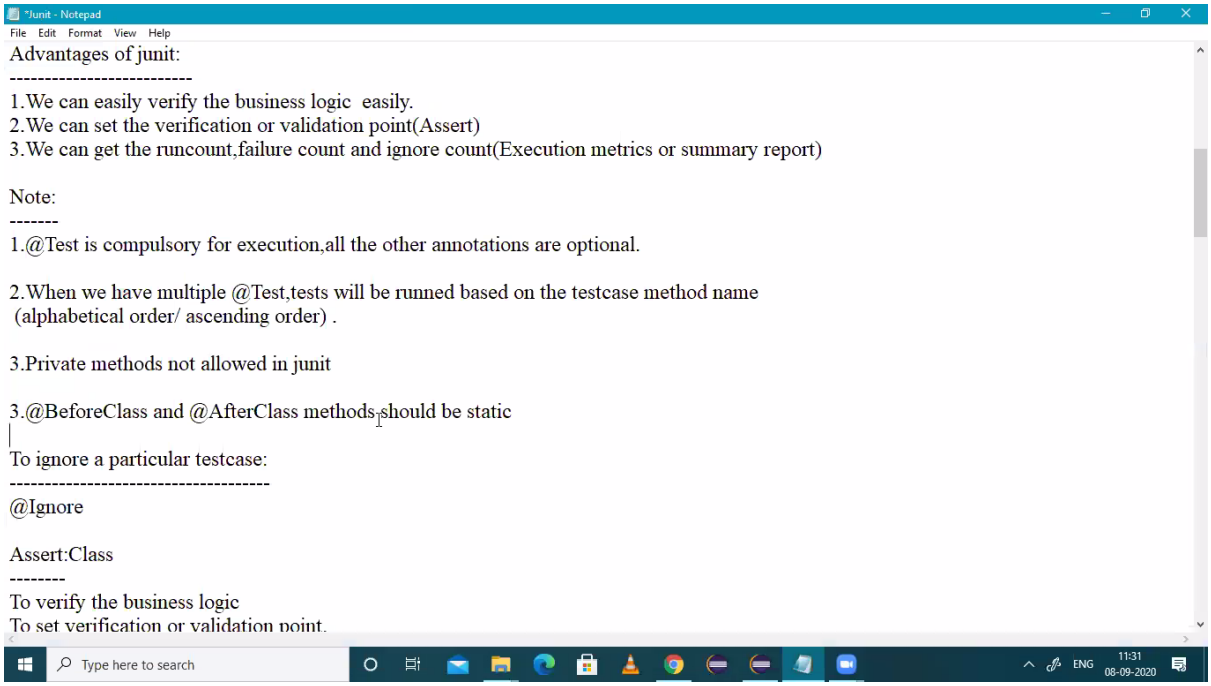

eclipse-workspace - JUnit10.30AM/src/test/java/org/test/Sample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
JUnit Package Explorer
Datadriven
Existing
FileOperations [File B]
FirstDayCourse [Java B]
JavaProject [JavaProject NO-HEAD]
JUnit10.30AM
src/main/java
src/main/resources
src/test/java
org.test
Sample.java
src/test/resources
JRE System Library [J2SE-1.5]
Maven Dependencies
driver
chromedriver.exe
src
target
pom.xml
JUnit9AM
MyProject
Project10.30am
Project5Pm
Project6.30PM [ProjectClass B]
Selenium
SeleniumDay1
JRE System Library [JavaSE-1.8]
src
Referenced Libraries
drivers
chromedriver.exe
JUnit10.30AM/pom.xml
Sample.java
public class Sample {
public static WebDriver driver;
@BeforeClass
public static void launch() {
System.setProperty("webdriver.chrome.driver",
"C:\\Users\\Prabha\\eclipse-workspace\\JUnit10.30AM\\driver\\chromedriver.exe");
driver = new ChromeDriver();
}
@AfterClass
public static void closeBrowser() {
driver.quit();
}
@Before
public void startTime() {
Date d = new Date();
System.out.println(d);
}
@After
public void endTime() {
Date d = new Date();
System.out.println(d);
}
// Invalid username and Invalid password
@Test
public void tc1() throws InterruptedException {
driver.get("https://www.facebook.com/");
Console
<terminated> Sample (8) [JUnit] C:\Program
Starting ChromeDriver 85.0.418
Only local connections are all
Please see https://chromedrive
ChromeDriver was started succe
Sep 08, 2020 11:26:12 AM org.o
INFO: Detected dialect: W3C
Tue Sep 08 11:26:14 IST 2020
Tue Sep 08 11:26:26 IST 2020
Tue Sep 08 11:26:26 IST 2020
Tue Sep 08 11:26:39 IST 2020
Tue Sep 08 11:26:39 IST 2020
Tue Sep 08 11:26:44 IST 2020
Tue Sep 08 11:26:44 IST 2020
Tue Sep 08 11:26:55 IST 2020
Writable
Smart Insert
32 : 1
Type here to search
ENG
11:28
08-09-2020


eclipse-workspace - JUnit10.30AM/src/test/java/org/test/Sample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
JUnit Package Explorer
JUnit10.30AM/pom.xml
Sample.java
// Invalid username and Invalid password
@Test
public void tc1() throws InterruptedException {
driver.get("https://www.facebook.com/");
WebElement txtEmail = driver.findElement(By.id("email"));
txtEmail.sendKeys("Greens");
WebElement txtPass = driver.findElement(By.id("pass"));
txtPass.sendKeys("greens@123");
WebElement btnLogin = driver.findElement(By.xpath("//button[@name='login']"));
btnLogin.click();
Thread.sleep(3000);
}
// valid username and Invalid password
@Test
public void tc2() throws InterruptedException {
driver.get("https://www.facebook.com/");
WebElement txtEmail = driver.findElement(By.id("email"));
txtEmail.sendKeys("java");
WebElement txtPass = driver.findElement(By.id("pass"));
txtPass.sendKeys("java@123");
WebElement btnLogin = driver.findElement(By.xpath("//button[@name='login']"));
btnLogin.click();
Thread.sleep(3000);
}
// Invalid username and valid password
Console
<terminated> Sample (8) [JUnit] C:\Program
Starting ChromeDriver 85.0.418
Only local connections are all
Please see https://chromedrive
ChromeDriver was started succe
Sep 08, 2020 11:26:12 AM org.o
INFO: Detected dialect: W3C
Tue Sep 08 11:26:14 IST 2020
Tue Sep 08 11:26:26 IST 2020
Tue Sep 08 11:26:26 IST 2020
Tue Sep 08 11:26:39 IST 2020
Tue Sep 08 11:26:39 IST 2020
Tue Sep 08 11:26:44 IST 2020
Tue Sep 08 11:26:44 IST 2020
Tue Sep 08 11:26:55 IST 2020
Writable
Smart Insert
34 : 1
Type here to search
ENG
11:29
08-09-2020


eclipse-workspace - JUnit10.30AM/src/test/java/org/test/Sample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
JUnit
Package Explorer
Datadriven
Existing
FileOperations [File B]
FirstDayCourse [Java B]
JavaProject [JavaProject NO-HEAD]
JUnit10.30AM
src/main/java
src/main/resources
src/test/java
org.test
Sample.java
src/test/resources
JRE System Library [J2SE-1.5]
Maven Dependencies
driver
chromedriver.exe
src
target
pom.xml
JUnit9AM
MyProject
Project10.30am
Project5Pm
Project6.30PM [ProjectClass B]
Selenium
SeleniumDay1
JRE System Library [JavaSE-1.8]
src
Referenced Libraries
drivers
chromedriver.exe
JUnit10.30AM/pom.xml
Sample.java
}
// Invalid username and valid password
@Test
public void tc3() throws InterruptedException {
driver.get("https://www.facebook.com/");
WebElement txtEmail = driver.findElement(By.id("email"));
txtEmail.sendKeys("sql");
WebElement txtPass = driver.findElement(By.id("pass"));
txtPass.sendKeys("sql@123");
WebElement btnLogin = driver.findElement(By.xpath("//button[@name='login']"));
btnLogin.click();
Thread.sleep(3000);
}
// valid username and valid password
@Test
public void tc4() throws InterruptedException {
driver.get("https://www.facebook.com/");
WebElement txtEmail = driver.findElement(By.id("email"));
txtEmail.sendKeys("Python");
WebElement txtPass = driver.findElement(By.id("pass"));
txtPass.sendKeys("Python@123");
WebElement btnLogin = driver.findElement(By.xpath("//button[@name='login']"));
btnLogin.click();
Thread.sleep(3000);
}
Console
<terminated> Sample (8) [JUnit] C:\Program
Starting ChromeDriver 85.0.418
Only local connections are all
Please see https://chromedrive
ChromeDriver was started succe
Sep 08, 2020 11:26:12 AM org.o
INFO: Detected dialect: W3C
Tue Sep 08 11:26:14 IST 2020
Tue Sep 08 11:26:26 IST 2020
Tue Sep 08 11:26:26 IST 2020
Tue Sep 08 11:26:39 IST 2020
Tue Sep 08 11:26:39 IST 2020
Tue Sep 08 11:26:44 IST 2020
Tue Sep 08 11:26:44 IST 2020
Tue Sep 08 11:26:55 IST 2020
Writable
Smart Insert
104 : 1
Type here to search
ENG
11:30
08-09-2020


*Junit - Notepad
File Edit Format View Help
Junit framework:
---
mainly used by developers fot Unit testing
Jar/Dependency required:
---
junit 4.12
Hamcrest-all 1.3(supporting jar)--------To avoid NoClassDefinitionFound
Annotations:

@BeforeClass-------executes before the execution of all the @Test
(we can mention the browser launch steps)
@Before----------executes before the execution of each @Test
(We can calculate the start time of execution of @Test)
@Test----------used to write the business logic
@After----------executes after the execution of each @Test
(We can calculate the end time of execution of @Test)
@AfterClass-------executes after the execution of all the @Test
(We can mention the steps to close the browser)
Type here to search
ENG
11:30
08-09-2020


*Junit - Notepad
File Edit Format View Help
Advantages of junit:
1.We can easily verify the business logic easily.
2.We can set the verification or validation point(Assert)
3.We can get the runcount,failure count and ignore count(Execution metrics or summary report)
Note:
1.@Test is compulsory for execution,all the other annotations are optional.
2.When we have multiple @Test,tests will be runned based on the testcase method name
(alphabetical order/ ascending order) .
3.Private methods not allowed in junit
3.@BeforeClass and @AfterClass methods should be static
To ignore a particular testcase:
@Ignore
Assert:Class
To verify the business logic
To set verification or validation point.
Type here to search
ENG
11:31
08-09-2020

TESTNG

TestNG (Test, Next, Generation)

TestNG is a framework

Jar dependencies required:

testng -> 6.14.3 (or) 6.7

jcommander -> 1.27 (supporting Jar)

There are lots of similarities between Junit and TestNG framework But, testNG has lots of better and advanced features compared to Junit.

Advantage of testNG over Junit

-> It provides the default html reports.

-> We change the order of execution of test cases using "priority".

-> We can ignore particular test case from execution using (enabled = false).

-> We can run a particular testcase multiple times using invocation count

-> We can pass data from testNG.xml to test case using parameters tag.

-> we can pass bulk of data to a testcase using Data provider

-> We can rerun the failed testcases both manually and automatically.

-> We can do parallel execution to save the time consumption.

-> We can do cross browser testing to check the compatibility and stability of the application

-> We can group the testcases

-> Both HardAssert and Soft Assert is possible.

Add testNG dependency:

-> Download testNG 6.14.3

-> Add it to eclipse workspace

-> install testNG plugin in eclipse market place

Note:- No Class DefenitionFound error will come so we need to add supporting jar called "Jcommander 1.27".

Annotations

@BeforeClass
@AfterClass
@BeforeMethod
@AfterMethod
@Test
@BeforeSuite
@AfterSuite
@BeforeTest
@AfterTest
@BeforeGroup
@AfterGroup

Note:-

1) private methods allowed in testNG

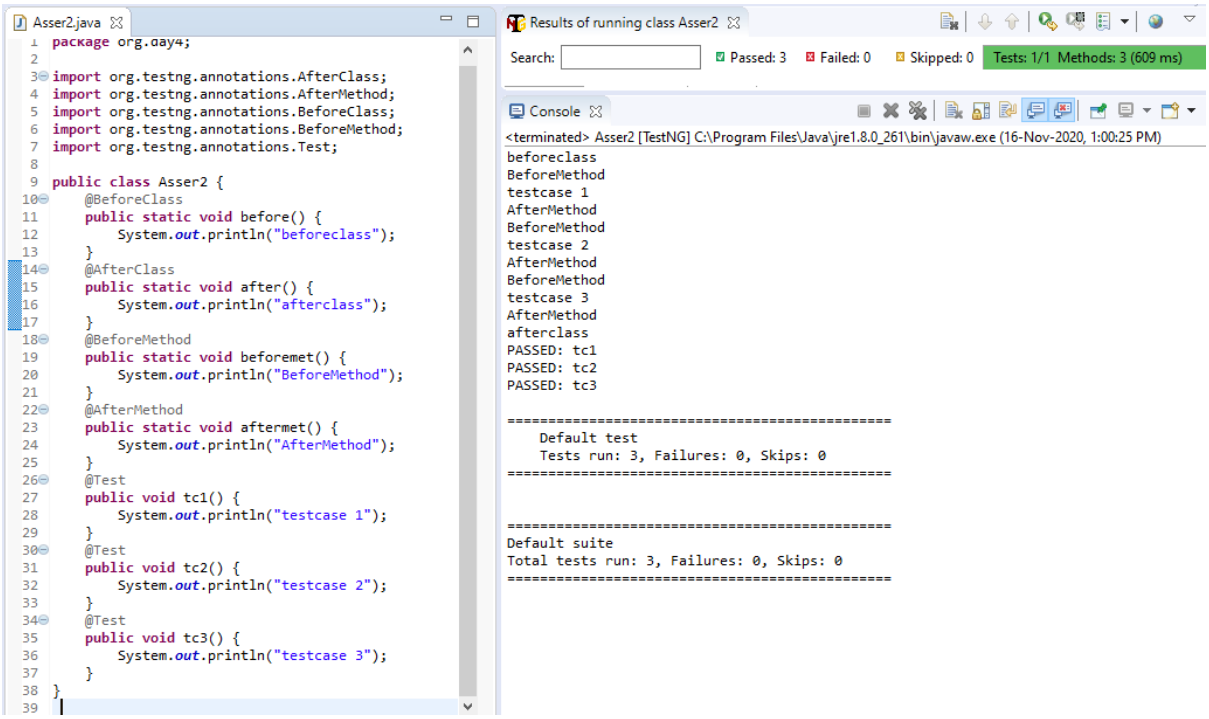
2) @BeforeClass and @AfterClass need not be static.

Eg:

```
@BeforeClass
public void line1(){
    Sysout("line-1");}
@AfterClass
public void lineLast(){
    Sysout("line-2");}
@AfterMethod
public void line3(){
    Sysout("line-3");}
@BeforeMethod
public void line4(){
    sysout("line-4");}
@Test
public void line5(){
    Sysout("line-5");}
@Test
public void line6(){
    Sysout("line-6");}
```

O/P =>

line-1 -> line-5 -> line-4 -> line-3 -> line-6 -> line-2



Asser2.java

```
1  package org.day4;
2
3  import org.testng.annotations.AfterClass;
4  import org.testng.annotations.AfterMethod;
5  import org.testng.annotations.BeforeClass;
6  import org.testng.annotations.BeforeMethod;
7  import org.testng.annotations.Test;
8
9  public class Asser2 {
10     @BeforeClass
11     public static void before() {
12         System.out.println("beforeclass");
13     }
14     @AfterClass
15     public static void after() {
16         System.out.println("afterclass");
17     }
18     @BeforeMethod
19     public static void beforemet() {
20         System.out.println("BeforeMethod");
21     }
22     @AfterMethod
23     public static void aftermet() {
24         System.out.println("AfterMethod");
25     }
26     @Test
27     public void tc1() {
28         System.out.println("testcase 1");
29     }
30     @Test
31     public void tc2() {
32         System.out.println("testcase 2");
33     }
34     @Test
35     public void tc3() {
36         System.out.println("testcase 3");
37     }
38 }
39
```

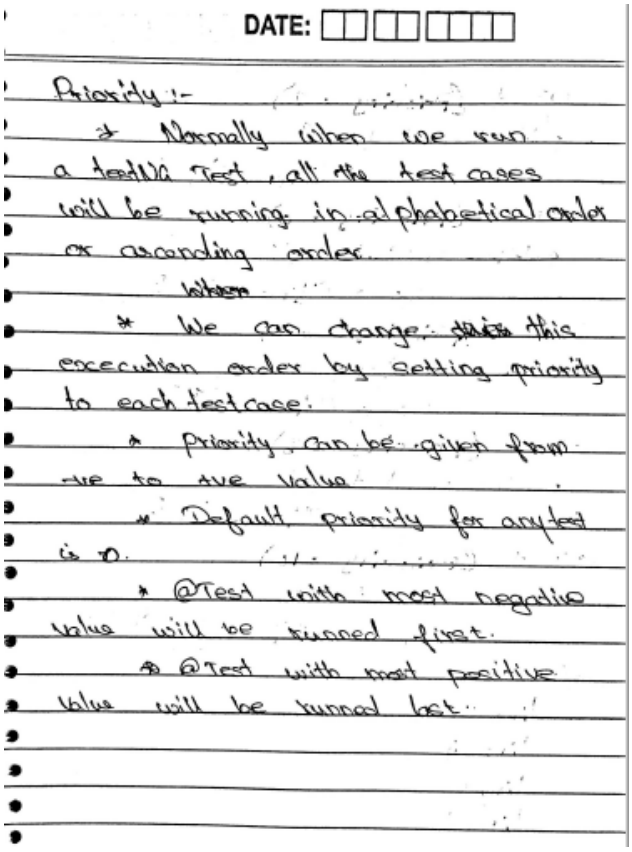
Results of running class Asser2

Search: | Passed: 3 | Failed: 0 | Skipped: 0 | Tests: 1/1 Methods: 3 (609 ms)

Console

```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (16-Nov-2020, 1:00:25 PM)
beforeclass
BeforeMethod
testcase 1
AfterMethod
BeforeMethod
testcase 2
AfterMethod
BeforeMethod
testcase 3
AfterMethod
afterclass
PASSED: tc1
PASSED: tc2
PASSED: tc3

===============================================
    Default test
    Tests run: 3, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 3, Failures: 0, Skips: 0
===============================================
```

PRIORITY



DATE:

Priority :-

* Normally when we run a testNG Test, all the test cases will be running in alphabetical order or ascending order.

* We can change this execution order by setting priority to each testcase.

* Priority can be given from -ve to +ve value.

* Default priority for any test is 0.

* @Test with most negative value will be runned first.

* @Test with most positive value will be runned last.

*Asser2.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser2 {
6      @Test(priority=10)
7      public void tc1() {
8          System.out.println("testcase 1");
9      }
10     @Test(priority=-5)
11     public void tc2() {
12         System.out.println("testcase 2");
13     }
14     @Test(priority=-10)
15     public void tc3() {
16         System.out.println("testcase 3");
17     }
18     @Test(priority=0)
19     public void tc4() {
20         System.out.println("testcase 4");
21     }
22     @Test
23     public void tc5() {
24         System.out.println("testcase 5");
25     }
26     @Test(priority=1)
27     public void tc6() {
28         System.out.println("testcase 6");
29     }
30 }
31
```

Results of running class Asser2

Search: Passed: 6 Failed: 0

Console

```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\
testcase 3
testcase 2
testcase 4
testcase 5
testcase 6
testcase 1
PASSED: tc3
PASSED: tc2
PASSED: tc4
PASSED: tc5
PASSED: tc6
PASSED: tc1

===============================================
    Default test
    Tests run: 6, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 6, Failures: 0, Skips: 0
===============================================
```

IGNORE PARTICULAR TESTCASE

Enabled

DATE:

If we want to ignore the execution (skip) of the particular testcase we can give (enabled = false).

Asser2.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser2 {
6
7      @Test
8      public void tc1() {
9          System.out.println("testcase 1");
10     }
11     @Test
12     public void tc2() {
13         System.out.println("testcase 2");
14     }
15     @Test(enabled=false)
16     public void tc3() {
17         System.out.println("testcase 3");
18     }
19     @Test
20     public void tc4() {
21         System.out.println("testcase 4");
22     }
23 }
24
```

Results of running class Asser2

Search: Passed: 3 Failed: 0 Ski

Console

```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\ja
testcase 1
testcase 2
testcase 4
PASSED: tc1
PASSED: tc2
PASSED: tc4

===============================================
    Default test
    Tests run: 3, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 3, Failures: 0, Skips: 0
===============================================
```

INVOCATION COUNT

Invocation Count DATE:

We can run a particular test case multiple times using invocation count.

When we run a particular test case again and again we can check the stability of the testscript.

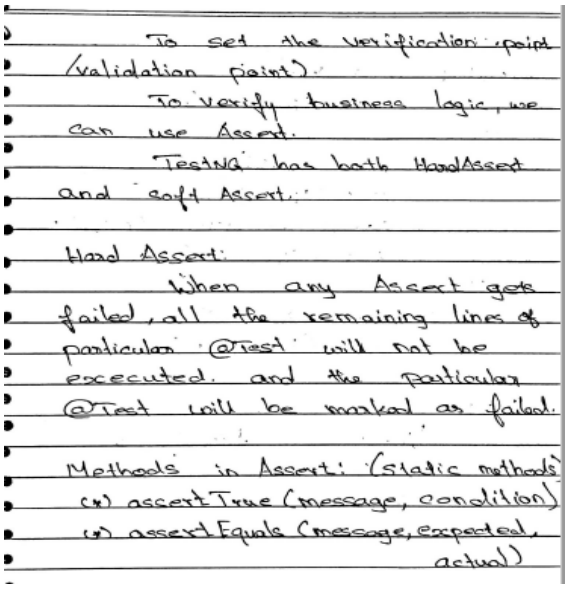
Asser2.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser2 {
6
7      @Test
8      public void tc1() {
9          System.out.println("testcase 1");
10     }
11     @Test
12     public void tc2() {
13         System.out.println("testcase 2");
14     }
15     @Test(invocationCount=3)
16     public void tc3() {
17         System.out.println("testcase 3");
18     }
19     @Test
20     public void tc4() {
21         System.out.println("testcase 4");
22     }
23 }
24
```

Console

```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_26
testcase 1
testcase 2
testcase 3
testcase 3
testcase 3
testcase 4
PASSED: tc1
PASSED: tc2
PASSED: tc3
PASSED: tc3
PASSED: tc3
PASSED: tc4

===============================================
    Default test
    Tests run: 6, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 6, Failures: 0, Skips: 0
===============================================
```

HARD ASSERT



To set the verification point (validation point).

To verify business logic, we can use Assert.

TestNG has both HardAsset and soft Assert.

Hard Assert:

When any Assert gets failed, all the remaining lines of particular @Test will not be executed. and the particular @Test will be marked as failed.

Methods in Assert: (static methods)

(*) assertTrue(message, condition)

(*) assertEquals(message, expected, actual)

Asser2.java

```
1  package org.day4;
2
3  import org.testng.Assert;
4  import org.testng.annotations.Test;
5
6  public class Asser2 {
7      @Test
8      public void tc1() {
9          System.out.println("1");
10     }
11     @Test
12     public void tc2() {
13         System.out.println("2");
14         //Hard Assert
15         Assert.assertTrue(false);
16         System.out.println("3");
17         System.out.println("4");
18         System.out.println("5");
19     }
20     @Test
21     public void tc3() {
22         System.out.println("6");
23     }
24 }
25
```

Results of running class Asser2

Search: | Passed: 2 | Failed: 1 | Skipped: 0 | Tests: 1/1 Methods: 3 (1661 ms)

All Tests | Failed Tests | Summary

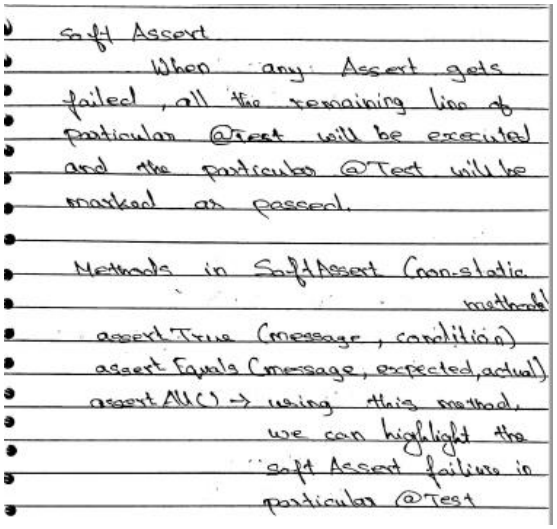
- Default suite (2/1/0/0) (0.125 s)
  - Default test (0.125 s)
    - org.day4.Asser2
      - tc1 (0.063 s)
      - tc2 (0.015 s)
      - tc3 (0.047 s)

Failure Exception

Console

```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (15-Nov-2020, 3:23:57 PM)
1
2
6
PASSED: tc1
PASSED: tc3
FAILED: tc2
java.lang.AssertionError: expected [true] but found [false]
        at org.testng.Assert.fail(Assert.java:96)
        at org.testng.Assert.failNotEquals(Assert.java:776)
        at org.testng.Assert.assertTrue(Assert.java:44)
        at org.testng.Assert.assertTrue(Assert.java:54)
        at org.day4.Asser2.tc2(Asser2.java:15)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
```

SOFT ASSERT



*Asser2.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4  import org.testng.asserts.SoftAssert;
5
6  public class Asser2 {
7      @Test
8      public void tc1() {
9          System.out.println("1");
10     }
11     @Test
12     public void tc2() {
13         System.out.println("2");
14         //SOFT ASSERT (without assertAll())
15         SoftAssert as=new SoftAssert();
16         as.assertTrue(false);
17         System.out.println("3");
18         System.out.println("4");
19         System.out.println("5");
20     }
21     @Test
22     public void tc3() {
23         System.out.println("6");
24     }
25 }
26
```

Results of running class Asser2

Search: | Passed: 3 | Failed: 0 | Skipped: 0 | Tests: 1/1

All Tests | Failed Tests | Summary

- Default suite (3/0/0/0) (0.079 s)
  - Default test (0.079 s)
    - org.day4.Asser2
      - tc1 (0.047 s)
      - tc2 (0.032 s)
      - tc3 (0 s)

Console

<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (15-Nov-

```
===============================================
    Default test
    Tests run: 3, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 3, Failures: 0, Skips: 0
===============================================
```

*Asser2.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4  import org.testng.asserts.SoftAssert;
5
6  public class Asser2 {
7      @Test
8      public void tc1() {
9          System.out.println("1");
10     }
11     @Test
12     public void tc2() {
13         System.out.println("2");
14         //SOFT ASSERT (with assertAll())
15         SoftAssert as=new SoftAssert();
16         as.assertTrue(false);
17         System.out.println("3");
18         System.out.println("4");
19         as.assertAll();
20         System.out.println("5");
21     }
22     @Test
23     public void tc3() {
24         System.out.println("6");
25     }
26 }
27
```

Results of running class Asser2

Search: | Passed: 2 | Failed: 1 | Skipped: 0 | Tests: 1/1 Me

All Tests | Failed Tests | Summary

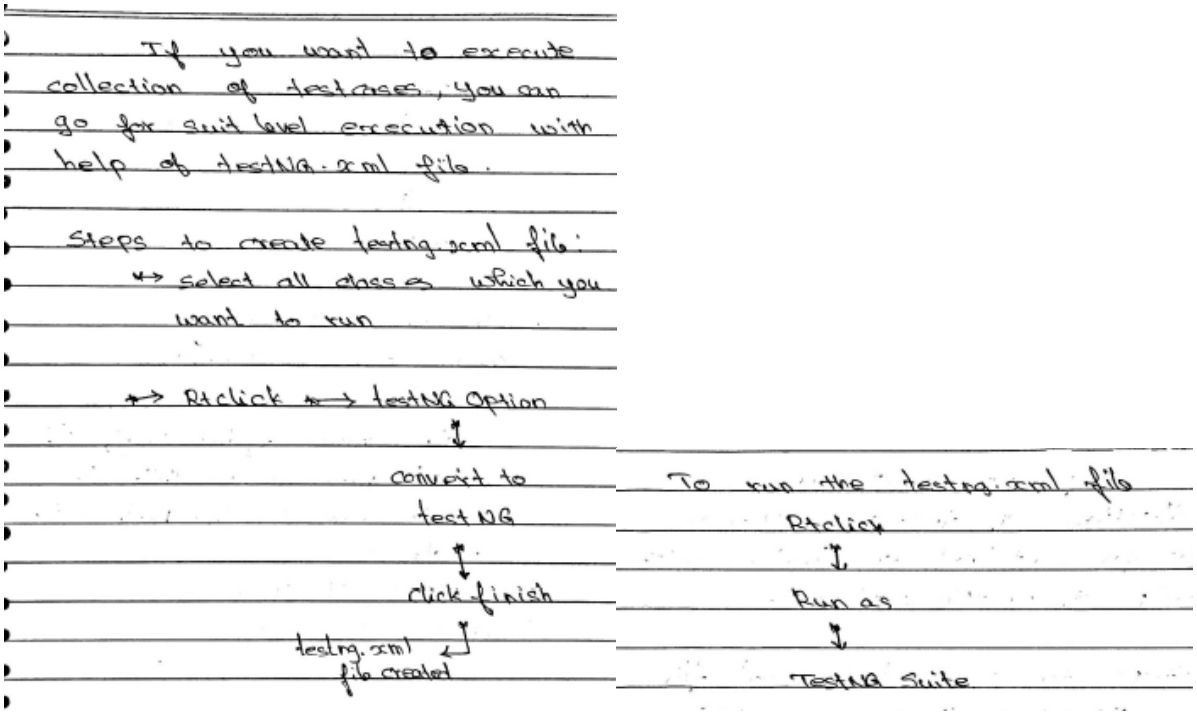
- Default suite (2/1/0/0) (0.078 s)
  - Default test (0.078 s)
    - org.day4.Asser2
      - tc1 (0.047 s)
      - tc2 (0.031 s)
      - tc3 (0 s)

Console

<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (15-Nov-2020

```
===============================================
    Default test
    Tests run: 3, Failures: 1, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 3, Failures: 1, Skips: 0
===============================================
```

SUITE LEVEL EXECUTION



*Asser2.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser2 {
6      @Test
7      public void tc1() {
8          System.out.println("1");
9      }
10     @Test
11     public void tc3() {
12         System.out.println("3");
13     }
14     @Test
15     public void tc5() {
16         System.out.println("5");
17     }
18     @Test
19     public void tc7() {
20         System.out.println("7");
21     }
22 }
23
```

Asser4.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser4 {
6      @Test
7      public void tc1() {
8          System.out.println("11");
9      }
10     @Test
11     public void tc2() {
12         System.out.println("22");
13     }
14     @Test
15     public void tc3() {
16         System.out.println("33");
17     }
18     @Test
19     public void tc4() {
20         System.out.println("44");
21     }
22 }
23
```

*Asser3.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test
4
5  public class Asser3 {
6      @Test
7      public void tc2() {
8          System.out.println("2");
9      }
10     @Test
11     public void tc4() {
12         System.out.println("4");
13     }
14     @Test
15     public void tc6() {
16         System.out.println("6");
17     }
18     @Test
19     public void tc8() {
20         System.out.println("8");
21     }
22 }
23
24
```

testng.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testı
<suite name="Suite">
  <test name="Test">
    <classes>
      <class name="org.day4.Asser2"/>
      <class name="org.day4.Asser3"/>
      <class name="org.day4.Asser4"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Results of running suite

Search:

Passed: 12 Failed: 0 Skipped: 0

Tests: 1/1 Methods: 12 (1681 ms)

All Tests | Failed Tests | Summary

- Suite (12/0/0/0) (0.113 s)
  - Test (0.113 s)
    - org.day4.Asser2
      - tc1 (0.033 s)
      - tc3 (0.006 s)
      - tc5 (0.004 s)
      - tc7 (0.008 s)
    - org.day4.Asser3
      - tc2 (0.018 s)
      - tc4 (0.02 s)
      - tc6 (0.004 s)
      - tc8 (0.003 s)
    - org.day4.Asser4
      - tc1 (0.004 s)
      - tc2 (0.005 s)
      - tc3 (0.002 s)
      - tc4 (0.006 s)

Failure Exception

Console

<terminated> F_04_TestNG_testng.xml [TestNG] C:\Program Files\Java\jre1.8

```
1
3
5
7
2
4
6
8
11
22
33
44

===============================================
Suite
Total tests run: 12, Failures: 0, Skips: 0
===============================================
```

*testng.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testı
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="org.day4.Asser2"/>
      <class name="org.day4.Asser3"/>
      <class name="org.day4.Asser4"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Results of running suite

Search:

Passed: 12 Failed: 0 Skipped: 0

Tests: 1/1 Methods: 12 (787 ms)

All Tests | Failed Tests | Summary

- Suite (12/0/0/0) (0.105 s)
  - Test (0.105 s)
    - org.day4.Asser2
      - tc1 (0.036 s)
      - tc3 (0.006 s)
      - tc5 (0.006 s)
      - tc7 (0.005 s)
    - org.day4.Asser3
      - tc2 (0.004 s)
      - tc4 (0.018 s)
      - tc6 (0.006 s)
      - tc8 (0.002 s)
    - org.day4.Asser4
      - tc1 (0.012 s)
      - tc2 (0.004 s)
      - tc3 (0.003 s)
      - tc4 (0.003 s)

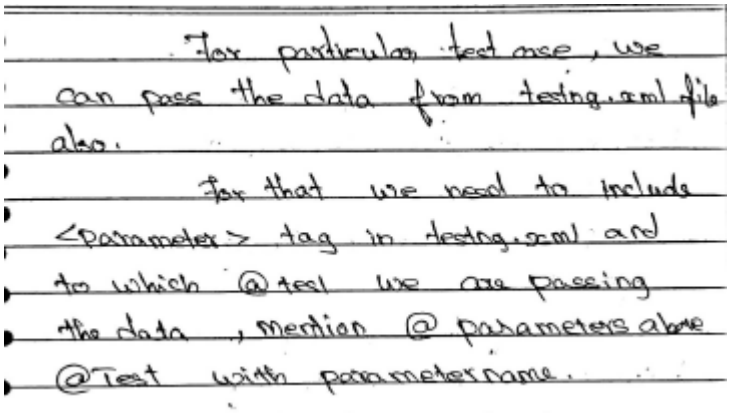
Failure Exception

Console

<terminated> F_04_TestNG_testng.xml [TestNG] C:\Program Files\Java\jre

```
1
3
5
7
2
4
6
8
11
22
33
44

===============================================
Suite
Total tests run: 12, Failures: 0, Skips: 0
===============================================
```

PARAMETERS



Asser.java

```
11 public class Asser {
12     public static WebDriver driver;
13     @Parameters({"email","pass"})
14     @Test
15     public void tc1(String email,String pass) {
16         WebDriverManager.chromedriver().setup();
17         driver = new ChromeDriver();
18         driver.manage().window().maximize();
19         driver.get("https://www.facebook.com/");
20         WebElement txtEmail = driver.findElement(By.id("email"));
21         txtEmail.sendKeys(email);
22         WebElement txtPass = driver.findElement(By.id("pass"));
23         txtPass.sendKeys(pass);
24     }
25 }
```

*testng1.xml

```
2 <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
3 <suite name="Suite">
4     <test name="Test">
5         <parameter name="email" value="abc@gmail.com">
6         </parameter>
7         <parameter name="pass" value="123"></parameter>
8         <classes>
9             <class name="org.day4.Asser" />
10        </classes>
11    </test> <!-- Test -->
12 </suite> <!-- Suite -->
13
```

If the parameter name passed in the xml file and @parameters has mismatch, then it will throw exception.

But we can handle that using @optional:

So if there is any mismatch between parameter name, value given in @optional will be taken as parameter value.

Asser.java

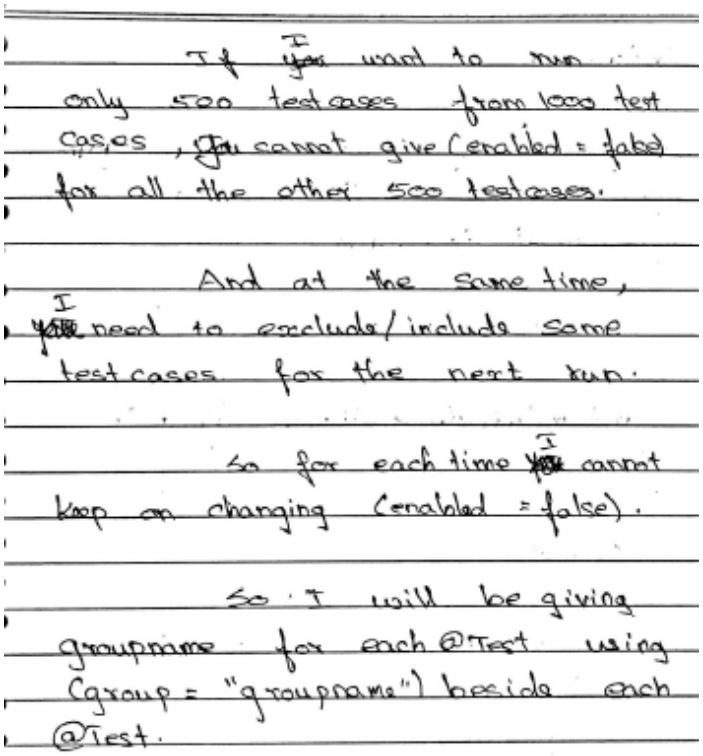
```
12 public class Asser {
13     public static WebDriver driver;
14     @Parameters({"mai","pass"})
15     @Test
16     public void tc1(@Optional("xyzyzv@hag.com")String email,String pass) {
17         WebDriverManager.chromedriver().setup();
18         driver = new ChromeDriver();
19         driver.manage().window().maximize();
20         driver.get("https://www.facebook.com/");
21         WebElement txtEmail = driver.findElement(By.id("email"));
22         txtEmail.sendKeys(email);
23         WebElement txtPass = driver.findElement(By.id("pass"));
24         txtPass.sendKeys(pass);
25     }
26 }
```

testng1.xml

```
2 <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
3 <suite name="Suite">
4     <test name="Test">
5         <parameter name="email" value="abc@gmail.com">
6         </parameter>
7         <parameter name="pass" value="123"></parameter>
8         <classes>
9             <class name="org.day4.Asser" />
10        </classes>
11    </test> <!-- Test -->
12 </suite> <!-- Suite -->
13
```

Design Source

GROUPING:



Asser2.java

```java
package org.day4;

import org.testng.annotations.Test;

public class Asser2 {
    @Test(groups="reg")
    public void tc1() {
        System.out.println("1");
    }
    @Test(groups="smoke")
    public void tc3() {
        System.out.println("3");
    }
    @Test(groups="sanity")
    public void tc5() {
        System.out.println("5");
    }
    @Test(groups="smoke")
    public void tc7() {
        System.out.println("7");
    }
}
```

Asser3.java

```java
package org.day4;

import org.testng.annotations.Test;

public class Asser3 {
    @Test(groups="reg")
    public void tc2() {
        System.out.println("2");
    }
    @Test(groups="smoke")
    public void tc4() {
        System.out.println("4");
    }
    @Test(groups="sanity")
    public void tc6() {
        System.out.println("6");
    }
    @Test(groups="reg")
    public void tc8() {
        System.out.println("8");
    }
}
```

Asser4.java

```java
package org.day4;

import org.testng.annotations.Test;

public class Asser4 {
    @Test(groups= {"smoke","reg"})
    public void tc1() {
        System.out.println("11");
    }
    @Test(groups="reg")
    public void tc2() {
        System.out.println("22");
    }
    @Test(groups="smoke")
    public void tc3() {
        System.out.println("33");
    }
    @Test(groups="sanity")
    public void tc4() {
        System.out.println("44");
    }
}
```

Include:

include

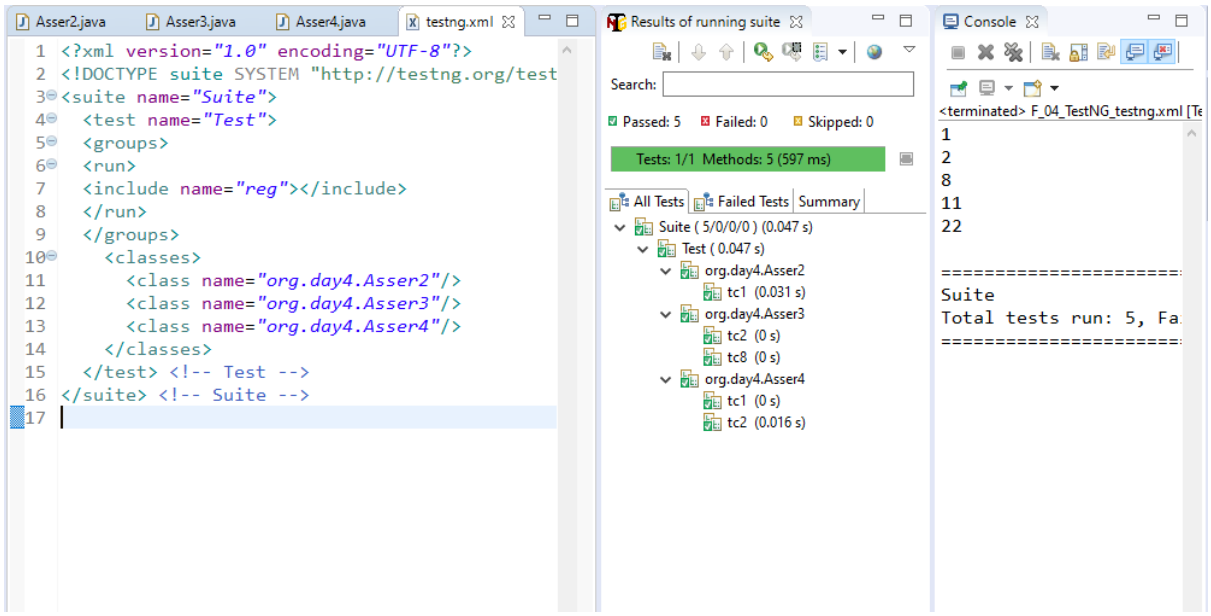
If you want to run 300 test cases from 1000 test cases, give group name to all 300 test cases you want to run.

eg:

@Test(groups = "smoke")

and include the particular group in testng.xml

```
<groups>
<run>
<include name = "smoke" />
</run>
</groups>
```



Exclude:

exclude

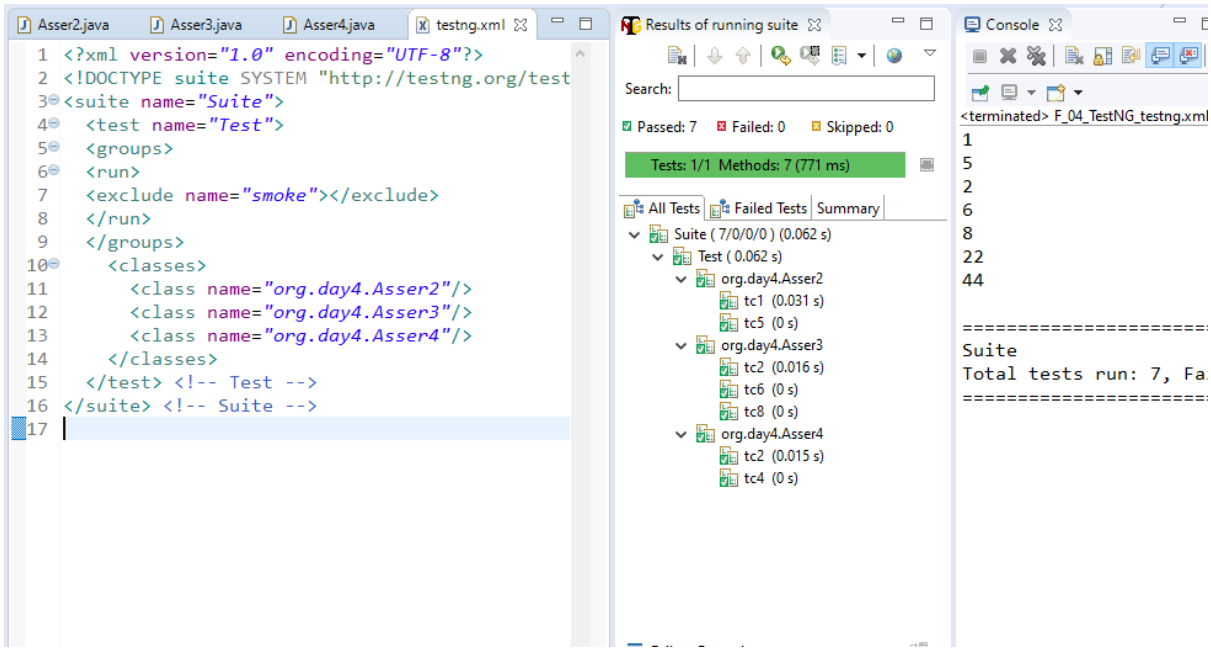
If you want to ignore 300 testcases from 1000 testcases, give group name to all 300 test cases you want to ignore.

eg:

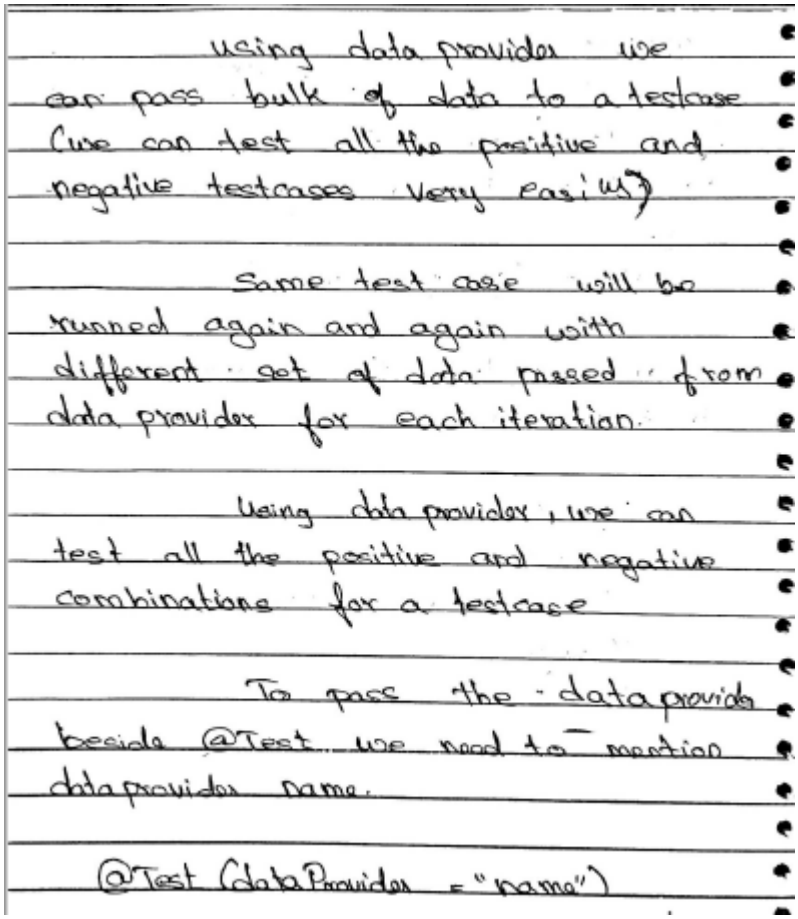
@Test (groups = "smoke")

and exclude the particular group in testng.xml

```
<groups>
<run>
<exclude name = "smoke" />
</run>
</groups>
```



DATAPROVIDER:



*Asser.java

```
10
11 public class Asser {
12     public static WebDriver driver;
13
14     @Test
15     public void tc1() {
16         WebDriverManager.chromedriver().setup();
17         driver = new ChromeDriver();
18         driver.manage().window().maximize();
19     }
20
21     @Test(dataProvider="sampleData")
22     public void tc2(String email,String pass) {
23         driver.get("https://www.facebook.com/");
24         WebElement txtEmail = driver.findElement(By.id("email"));
25         txtEmail.sendKeys(email);
26         WebElement txtPass = driver.findElement(By.id("pass"));
27         txtPass.sendKeys(pass);
28     }
29
30     @DataProvider(name="sampleData")
31     public Object[][] data() {
32         return new Object[][] {
33             {"abc","abcdef"},
34             {"ijk","ijklmn"},
35             {"xyz","xyzabc"}
36         };
37     }
38 }
39
```

Console

<terminated> Asser [TestNG] C:\Program Files\Java\jre1.

```
SLF4J: Failed to load class "org.
SLF4J: Defaulting to no-operation
SLF4J: See http://www.slf4j.org/c
Starting ChromeDriver 86.0.4240.2
Only local connections are allowe
Please see https://chromedriver.c
ChromeDriver was started successf
Nov 15, 2020 3:10:31 PM org.openq
INFO: Detected dialect: W3C
PASSED: tc1
PASSED: tc2("abc", "abcdef")
PASSED: tc2("ijk", "ijklmn")
PASSED: tc2("xyz", "xyzabc")

===================================
    Default test
    Tests run: 4, Failures: 0, Sk
===================================


===================================
Default suite
Total tests run: 4, Failures: 0,
===================================
```

Data provider from other class

testng.xml | Asser.java | Asser0.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser0 {
6
7      @Test(dataProvider="sampleData",dataProviderClass=Asser.class)
8      public void tc2(String email,String pass) {
9          System.out.println(email);
10         System.out.println(pass);
11         System.out.println("_____");
12     }
13 }
14
```

Results of running class Asser0

Search | Passed: 3 | Failed: 0 | Skipped: 0

Console

<terminated> Asser0 [TestNG] C:\Program Files\Java\j

```
abc
abcdef
_____
ijk
ijklmn
_____
xyz
xyzabc
_____
PASSED: tc2("abc", "abcdef")
PASSED: tc2("ijk", "ijklmn")
PASSED: tc2("xyz", "xyzabc")

===============================
    Default test
    Tests run: 3, Failures: 0, S
===============================
```

PARALLEL EXECUTION:

To reduce the execution time we use parallel execution.

We can acheive parallel execution in 3 ways

(*) using methods

(*) using classes

(*) using tests.

testng.xml | Asser2.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser2 {
6      @Test
7      public void tc1() {
8          System.out.println("1");
9      }
10     @Test
11     public void tc2() {
12         System.out.println("2");
13     }
14     @Test
15     public void tc3() {
16         System.out.println("3");
17     }
18 }
19
```

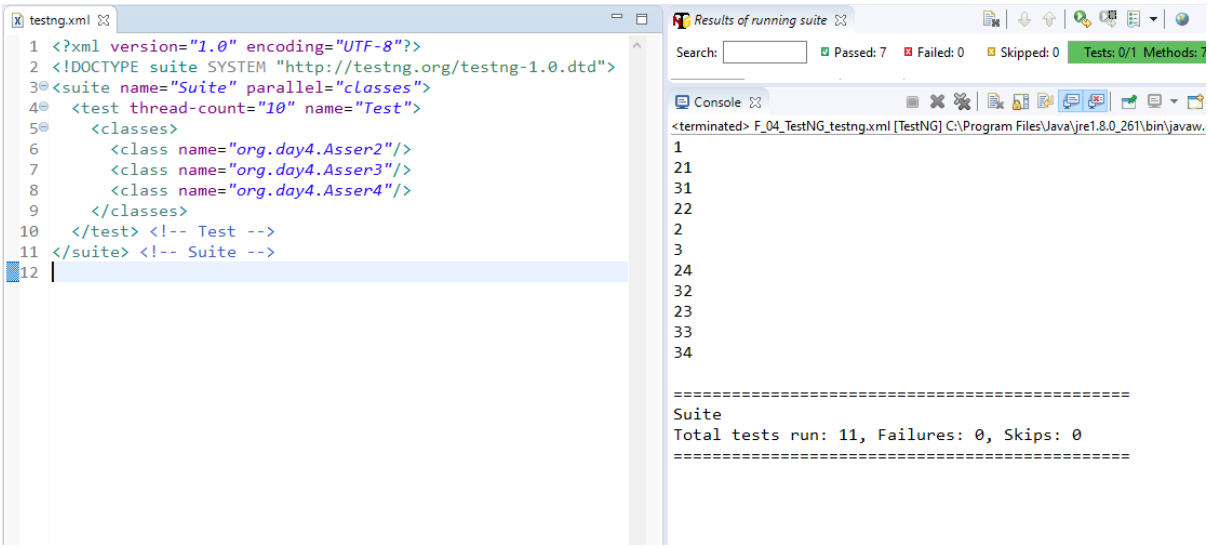
Asser3.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser3 {
6      @Test
7      public void tc21() {
8          System.out.println("21");
9      }
10     @Test
11     public void tc22() {
12         System.out.println("22");
13     }
14     @Test
15     public void tc623() {
16         System.out.println("23");
17     }
18     @Test
19     public void tc24() {
20         System.out.println("24");
21     }
22 }
23
```

Asser4.java

```
1  package org.day4;
2
3  import org.testng.annotations.Test;
4
5  public class Asser4 {
6      @Test
7      public void tc31() {
8          System.out.println("31");
9      }
10     @Test
11     public void tc32() {
12         System.out.println("32");
13     }
14     @Test
15     public void tc33() {
16         System.out.println("33");
17     }
18     @Test
19     public void tc34() {
20         System.out.println("34");
21     }
22 }
23
```

Using classes



testng.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="classes">
  <test thread-count="10" name="Test">
    <classes>
      <class name="org.day4.Asser2"/>
      <class name="org.day4.Asser3"/>
      <class name="org.day4.Asser4"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Results of running suite

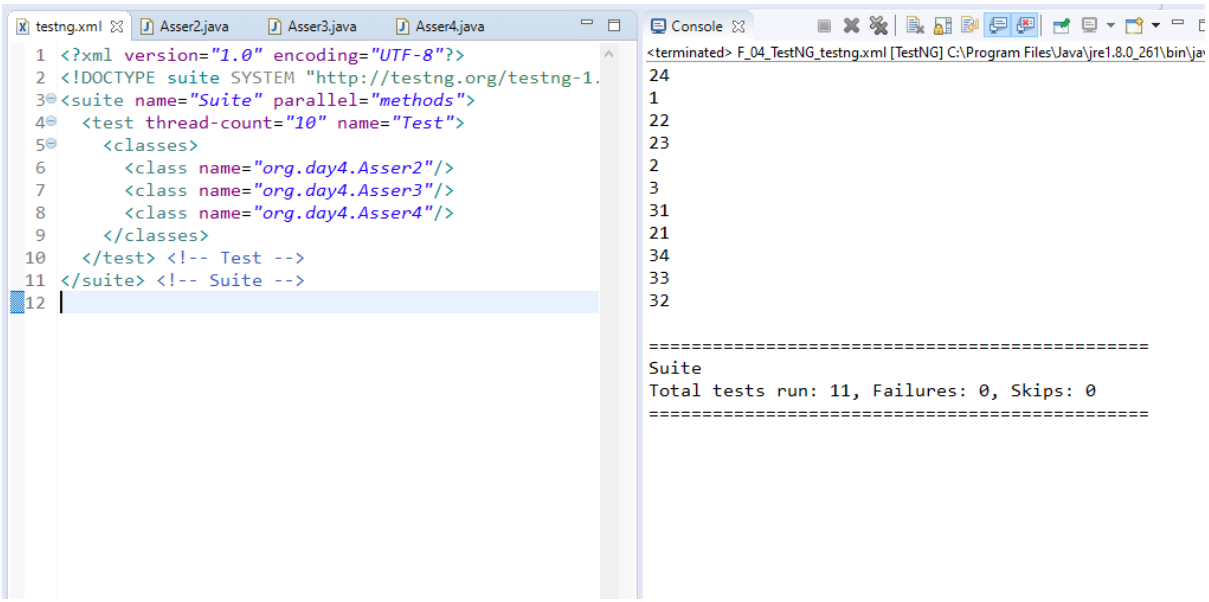
Search: Passed: 7 Failed: 0 Skipped: 0 Tests: 0/1 Methods: 7

Console

<terminated> F_04_TestNG_testng.xml [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\javaw.

```
1
21
31
22
2
3
24
32
23
33
34

===============================================
Suite
Total tests run: 11, Failures: 0, Skips: 0
===============================================
```

Using methods



testng.xml | Asser2.java | Asser3.java | Asser4.java

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.
<suite name="Suite" parallel="methods">
  <test thread-count="10" name="Test">
    <classes>
      <class name="org.day4.Asser2"/>
      <class name="org.day4.Asser3"/>
      <class name="org.day4.Asser4"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Console

<terminated> F_04_TestNG_testng.xml [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\ja

```
24
1
22
23
2
3
31
21
34
33
32

===============================================
Suite
Total tests run: 11, Failures: 0, Skips: 0
===============================================
```

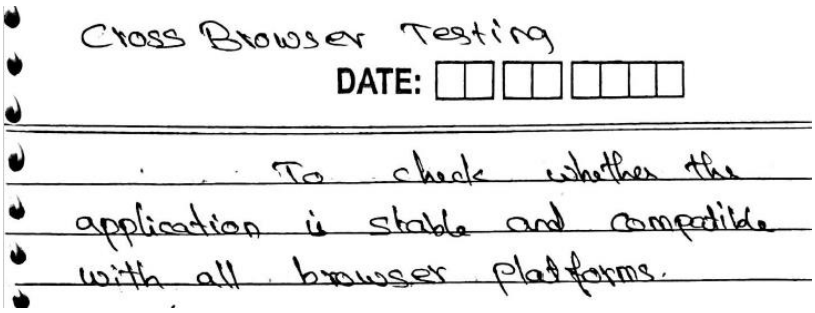
Using test

Asser.java | testng1.xml

```
import org.openqa.selenium.By;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.ie.InternetExplorerDriver;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;
import io.github.bonigarcia.wdm.WebDriverManager;

public class Asser {
    public static WebDriver driver;

    @Parameters({ "browser" })
    @Test
    public void tc1(String bwn) {

        if (bwn.equals("chrome")) {
            WebDriverManager.chromedriver().setup();
            driver = new ChromeDriver();
            driver.manage().window().maximize();
        } else if (bwn.equals("ie")) {
            WebDriverManager.iedriver().setup();
            driver = new InternetExplorerDriver();
            driver.manage().window().maximize();
        }

        driver.get("https://www.google.co.in/");
        driver.findElement(By.name("q")).sendKeys("Greens",Keys.ENTER);
    }
}
```

Asser.java | testng1.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="tests">
  <test name="chromeTest">
  <parameter name="browser" value="chrome"></parameter>
        <classes>
            <class name="org.day4.Asser" />
        </classes>
    </test> <!-- Test -->
      <test name="ieTest">
  <parameter name="browser" value="ie"></parameter>
        <classes>
            <class name="org.day4.Asser" />
        </classes>
    </test> <!-- Test -->
</suite> <!-- Suite -->
```



RERUN

testng1.xml | Asser.java

```
1  package org.day4;
2
3  import org.testng.Assert;
4  import org.testng.annotations.Test;
5
6  public class Asser {
7
8      @Test
9      public void tc1() {
10         System.out.println("Line 1");
11         System.out.println("line 2");
12         System.out.println("line 3");
13     }
14     @Test
15     public void tc2() {
16         System.out.println("line 4");
17         Assert.assertTrue(false);
18         System.out.println("line 5");
19     }
20     @Test
21     public void tc3() {
22         System.out.println("line 6");
23         System.out.println("line 7");
24     }
25 }
26
```

Results of running class Asser

Search: | Passed: 2 | Failed: 1 | Skipped: 0 | Tests: 1/1 Methods: 3 (542 ms)

Console

```
<terminated> Asser [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (15-Nov-2020, 9:12:17 PM)
Line 1
line 2
line 3
line 4
line 6
line 7
PASSED: tc1
PASSED: tc3
FAILED: tc2
java.lang.AssertionError: expected [true] but found [false]
        at org.testng.Assert.fail(Assert.java:96)
        at org.testng.Assert.failNotEquals(Assert.java:776)
        at org.testng.Assert.assertTrue(Assert.java:44)
        at org.testng.Assert.assertTrue(Assert.java:54)
        at org.day4.Asser.tc2(Asser.java:17)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native
        at sun.reflect.NativeMethodAccessorImpl.invoke(Unknown
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(Unk
        at java.lang.reflect.Method.invoke(Unknown Source)
        at org.testng.internal.MethodInvocationHelper.invokeMe
        at org.testng.internal.Invoker.invokeMethod(Invoker.ja
        at org.testng.internal.Invoker.invokeTestMethod(Invoke
```

Rerun the failed test case Manually

We can rerun the failed testcases in testng

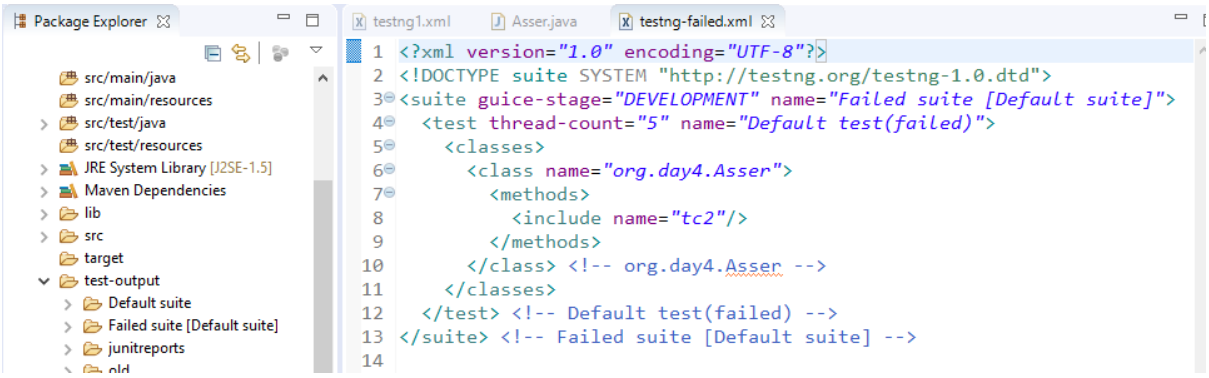
* Manually
* Automatically
  - → when you know which testcase is getting failed
  - → when you don't know which test case which is getting failed.

1) Manually Rerun.

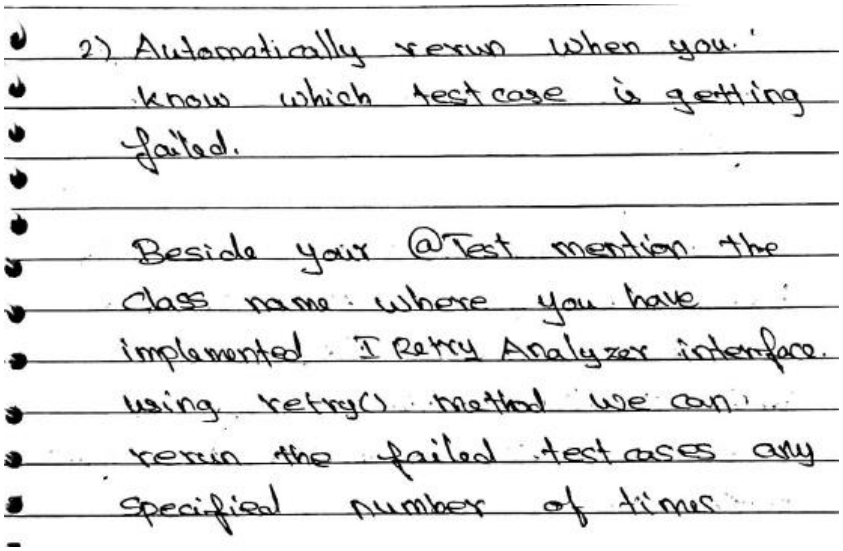
After every execution → all the failed methods will be recorded in testng-failed.xml (which is present in test-output folder)

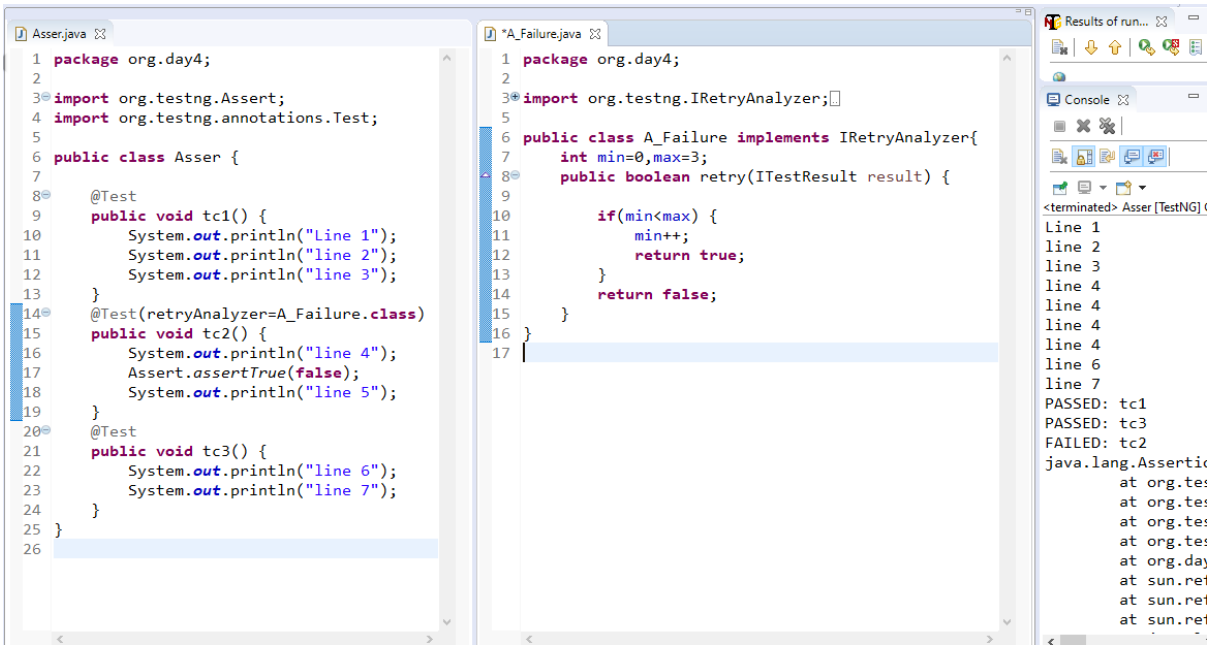
If we run testng-failed.xml only testcases failed in the previous run alone will runned again.

MANUAL RERUN



AUTOMATIC RERUN when we know which test case is getting failed





AUTOMATIC RERUN when we don't know which test case is getting failed

3) Automatically re-run when you don't know which testcase is getting failed

* First we need to add <listeners> tag in testng.xml then include listners class name (class which implements IAnnotationTransformer where we will over ride transform() method).

* Inside transform() method we use two methods.

(1) getRetryAnalyzer() is used to get listen all the failed testcases from <listeners> tag

(2) setRetryAnalyzer() → used to trigger the retry() method, so for setRetryAnalyzer() method we pass the class which implements IRetryAnalyzer as arguments.

So it will rerun the failed test cases till the maximum number of count we have specified.

```java
package org.day4;
import org.testng.Assert;

public class Asser {

    @Test
    public void tc1() {
        System.out.println("Line 1");
        System.out.println("line 2");
        System.out.println("line 3");
    }
    @Test(retryAnalyzer=A_Failure.class)
    public void tc2() {
        System.out.println("line 4");
        Assert.assertTrue(false);
        System.out.println("line 5");
    }
    @Test
    public void tc3() {
        System.out.println("line 6");
        System.out.println("line 7");
    }
}
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite">
    <listeners>
        <listener class-name="org.day4.FailedAll">
        </listener>
    </listeners>
    <test name="Test">
        <classes>
            <class name="org.day4.Asser" />
        </classes>
    </test> <!-- Test -->
</suite> <!-- Suite -->
```

Console

```
<terminated> F_04_TestNG
Line 1
line 2
line 3
line 4
line 4
line 4
line 4
line 6
line 7

===================
Suite
Total tests run: 6,
```

FailedAll.java

```java
package org.day4;

import java.lang.reflect.Constructor;
import java.lang.reflect.Method;

import org.testng.IAnnotationTransformer;
import org.testng.IRetryAnalyzer;
import org.testng.annotations.ITestAnnotation;

public class FailedAll implements IAnnotationTransformer{

    public void transform(ITestAnnotation annotation, Class testClass, Constructor testConstructor, Method testMethod) {

        IRetryAnalyzer r = annotation.getRetryAnalyzer();
        if(r==null) {
            annotation.setRetryAnalyzer(A_Failure.class);
        }
    }

```

*A_Failure.java

```java
package org.day4;
import org.testng.IRetryAnalyzer;

public class A_Failure implements IRetryAnalyzer{
    int min=0,max=3;
    public boolean retry(ITestResult result) {

        if(min<max) {
            min++;
            return true;
        }
        return false;
    }
}
```

Execution flow

*Asser2.java

```java
public class Asser2 {
    @BeforeSuite
    public void beforeSuit() {
        System.out.println("beforeSuit");
    }
    @AfterSuite
    public void afterSuit() {
        System.out.println("afterSuit");
    }
    @BeforeTest
    public void beforeTest() {
        System.out.println("beforeTest");
    }
    @AfterTest
    public void afterTest() {
        System.out.println("afterTest");
    }
    @BeforeClass
    public void beforeClass() {
        System.out.println("beforeClass");
    }
    @AfterClass
    public void afterClass() {
        System.out.println("afterClass");
    }
    @BeforeGroups("smoke")
    public void beforeGroupssmoke() {
        System.out.println("beforeGroupssmoke");
    }
    @AfterGroups("smoke")
    public void afterGroupssmoke() {
        System.out.println("afterGroupssmoke");
    }
    @BeforeGroups("regression")
    public void beforeGroupsregression() {
        System.out.println("beforeGroupsregression");
    }
    @AfterGroups("regression")
```

Asser2.java

```java
    @AfterGroups("smoke")
    public void afterGroupssmoke() {
        System.out.println("afterGroupssmoke");
    }
    @BeforeGroups("regression")
    public void beforeGroupsregression() {
        System.out.println("beforeGroupsregression");
    }
    @AfterGroups("regression")
    public void afterGroupsregression() {
        System.out.println("afterGroupsregression");
    }
    @BeforeMethod
    public void beforeMethod() {
        System.out.println("beforeMethod");
    }
    @AfterMethod
    public void afterMethod() {
        System.out.println("afterMethod");
    }
    @Test(groups="smoke")
    private void tc1() {
        System.out.println("**tc1**");
    }
    @Test(groups="smoke")
    private void tc2() {
        System.out.println("**tc1**");
    }
    @Test(groups="regression")
    private void tc3() {
        System.out.println("**tc1**");
    }
    @Test(groups="smoke")
    private void tc4() {
        System.out.println("**tc4**");
    }

}
```

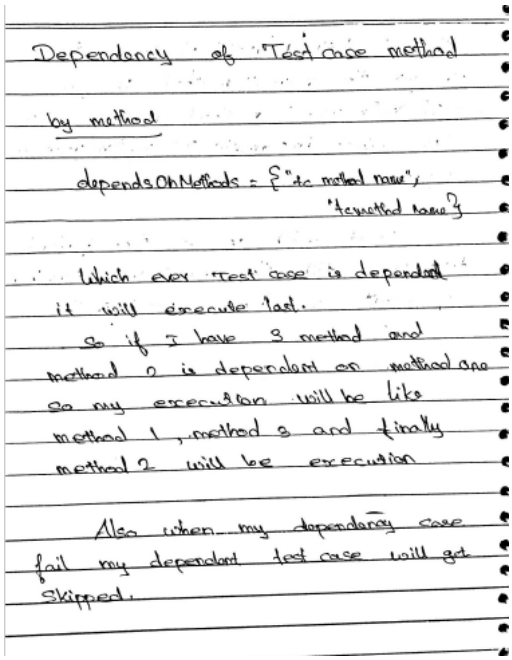
Results of running class Asser2

Search: Passed: 4 Failed: 0 Skipped: 0

Console

```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\b
beforeMethod
**tc1**
afterMethod
beforeGroupsregression
beforeMethod
**tc1**
afterMethod
afterGroupsregression
beforeMethod
**tc4**
afterMethod
afterGroupssmoke
afterClass
afterTest
PASSED: tc1
PASSED: tc2
PASSED: tc3
PASSED: tc4

===============================================
    Default test
    Tests run: 4, Failures: 0, Skips: 0
===============================================

afterSuit

===============================================
Default suite
Total tests run: 4, Failures: 0, Skips: 0
===============================================
```

DEPENDENCY OF TEST CASE METHOD



DEPENDENCY ON METHOD

Asser2.java

```
1  package org.day4;
2
3  import org.junit.Assert;
5
6  public class Asser2 {
7
8      @Test
9      private void tc1() {
10         Assert.assertTrue(true);
11         System.out.println("**tc1**");
12     }
13     @Test(dependsOnMethods= {"tc1"})
14     private void tc2() {
15         System.out.println("**tc2**");
16     }
17     @Test
18     private void tc3() {
19         System.out.println("**tc3**");
20     }
21
22 }
23
```

Results of running class Asser2

Search: Passed: 3 Failed: 0

Console

```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\
**tc1**
**tc3**
**tc2**
PASSED: tc1
PASSED: tc3
PASSED: tc2

===============================================
    Default test
    Tests run: 3, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 3, Failures: 0, Skips: 0
===============================================
```

Asser2.java

```
1  package org.day4;
2
3  import org.junit.Assert;
5
6  public class Asser2 {
7
8      @Test
9      private void tc1() {
10         Assert.assertTrue(false);
11         System.out.println("**tc1**");
12     }
13     @Test(dependsOnMethods= {"tc1"})
14     private void tc2() {
15         System.out.println("**tc2**");
16     }
17     @Test
18     private void tc3() {
19         System.out.println("**tc3**");
20     }
21
22 }
23
```

Results of running class Asser2

Search: Passed: 1 Failed: 1 Skipped: 1

Console

```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (1
**tc3**
PASSED: tc3
FAILED: tc1
java.lang.AssertionError
        at org.junit.Assert.fail(Assert.java:86)
        at org.junit.Assert.assertTrue(Assert.java:41)
        at org.junit.Assert.assertTrue(Assert.java:52)
        at org.day4.Asser2.tc1(Asser2.java:10)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native
        at sun.reflect.NativeMethodAccessorImpl.invoke(Unknown
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(Unk
        at java.lang.reflect.Method.invoke(Unknown Source)
        at org.testng.internal.MethodInvocationHelper.invokeMe
        at org.testng.internal.Invoker.invokeMethod(Invoker.ja
        at org.testng.internal.Invoker.invokeTestMethod(Invoke
        at org.testng.internal.Invoker.invokeTestMethods(Invok
```

DEPENDENCY ON GROUP

Dependency on group

When my dependency group fail my dependant test case will get skipped.

alwaysRun = true

if my dependency case fail also my dependant method will get executed.

@Test (dependsOnMethods = {"tc1"}, alwaysRun=true)

Asser2.java | testng123.xml

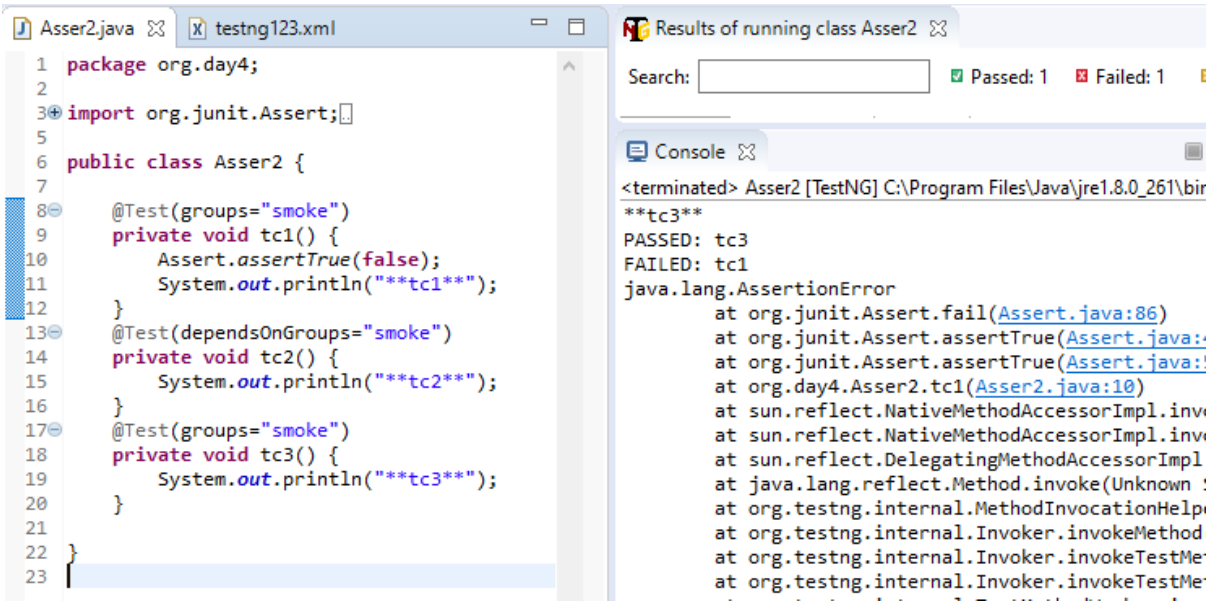
```
1  package org.day4;
2
3  import org.junit.Assert;
5
6  public class Asser2 {
7
8      @Test(groups="smoke")
9      private void tc1() {
10         Assert.assertTrue(true);
11         System.out.println("**tc1**");
12     }
13     @Test(dependsOnGroups="smoke")
14     private void tc2() {
15         System.out.println("**tc2**");
16     }
17     @Test(groups="smoke")
18     private void tc3() {
19         System.out.println("**tc3**");
20     }
21
22 }
23
```

Results of running class Asser2

Search: | Passed: 3 | Failed: 0

Console

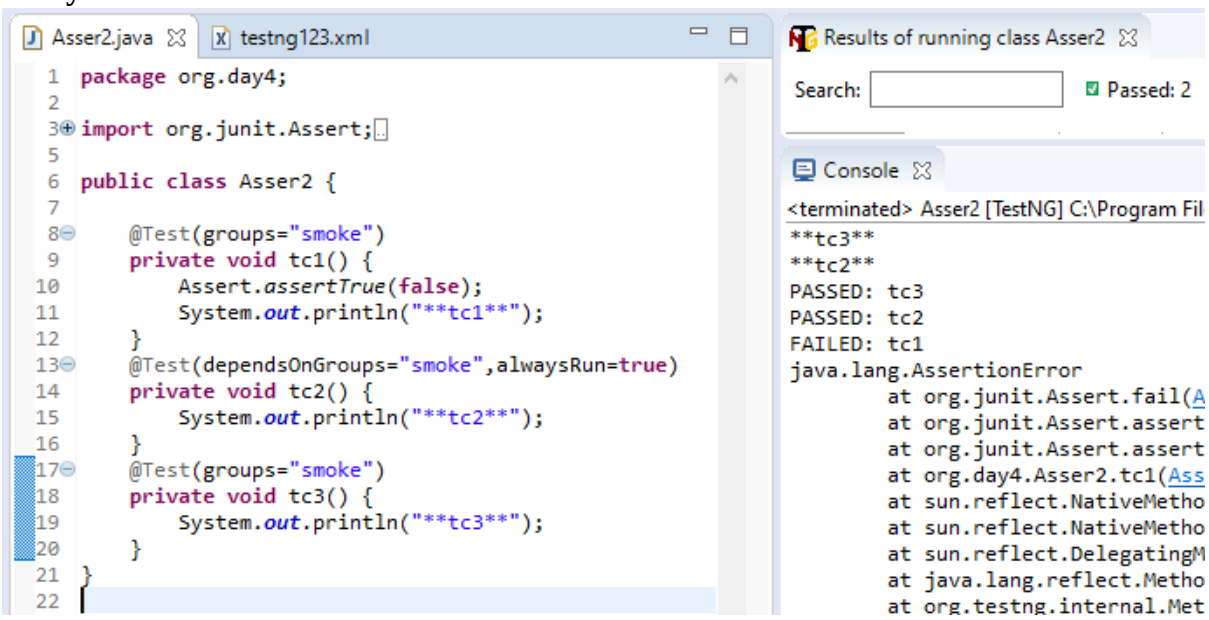
```
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_26
**tc1**
**tc3**
**tc2**
PASSED: tc1
PASSED: tc3
PASSED: tc2

===============================================
    Default test
    Tests run: 3, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 3, Failures: 0, Skips: 0
===============================================
```



```
Asser2.java | testng123.xml

1  package org.day4;
2
3  import org.junit.Assert;
5
6  public class Asser2 {
7
8      @Test(groups="smoke")
9      private void tc1() {
10         Assert.assertTrue(false);
11         System.out.println("**tc1**");
12     }
13     @Test(dependsOnGroups="smoke")
14     private void tc2() {
15         System.out.println("**tc2**");
16     }
17     @Test(groups="smoke")
18     private void tc3() {
19         System.out.println("**tc3**");
20     }
21
22 }
23
```

```
Results of running class Asser2
Search:  Passed: 1  Failed: 1

Console
<terminated> Asser2 [TestNG] C:\Program Files\Java\jre1.8.0_261\bir
**tc3**
PASSED: tc3
FAILED: tc1
java.lang.AssertionError
        at org.junit.Assert.fail(Assert.java:86)
        at org.junit.Assert.assertTrue(Assert.java:
        at org.junit.Assert.assertTrue(Assert.java:
        at org.day4.Asser2.tc1(Asser2.java:10)
        at sun.reflect.NativeMethodAccessorImpl.inv
        at sun.reflect.NativeMethodAccessorImpl.inv
        at sun.reflect.DelegatingMethodAccessorImpl
        at java.lang.reflect.Method.invoke(Unknown
        at org.testng.internal.MethodInvocationHelp
        at org.testng.internal.Invoker.invokeMethod
        at org.testng.internal.Invoker.invokeTestMe
        at org.testng.internal.Invoker.invokeTestMe
```

alwaysRun=true



```
Asser2.java | testng123.xml

1  package org.day4;
2
3  import org.junit.Assert;
5
6  public class Asser2 {
7
8      @Test(groups="smoke")
9      private void tc1() {
10         Assert.assertTrue(false);
11         System.out.println("**tc1**");
12     }
13     @Test(dependsOnGroups="smoke",alwaysRun=true)
14     private void tc2() {
15         System.out.println("**tc2**");
16     }
17     @Test(groups="smoke")
18     private void tc3() {
19         System.out.println("**tc3**");
20     }
21 }
22
```

```
Results of running class Asser2
Search:  Passed: 2

Console
<terminated> Asser2 [TestNG] C:\Program Fil
**tc3**
**tc2**
PASSED: tc3
PASSED: tc2
FAILED: tc1
java.lang.AssertionError
        at org.junit.Assert.fail(A
        at org.junit.Assert.assert
        at org.junit.Assert.assert
        at org.day4.Asser2.tc1(Ass
        at sun.reflect.NativeMetho
        at sun.reflect.NativeMetho
        at sun.reflect.DelegatingM
        at java.lang.reflect.Metho
        at org.testng.internal.Met
```