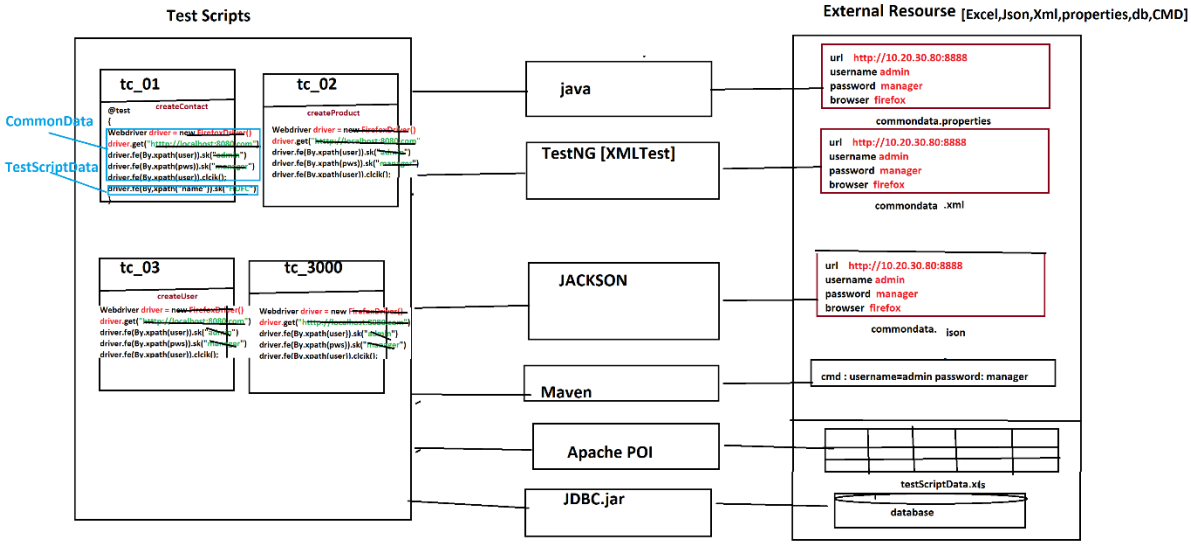
Data Driven testing

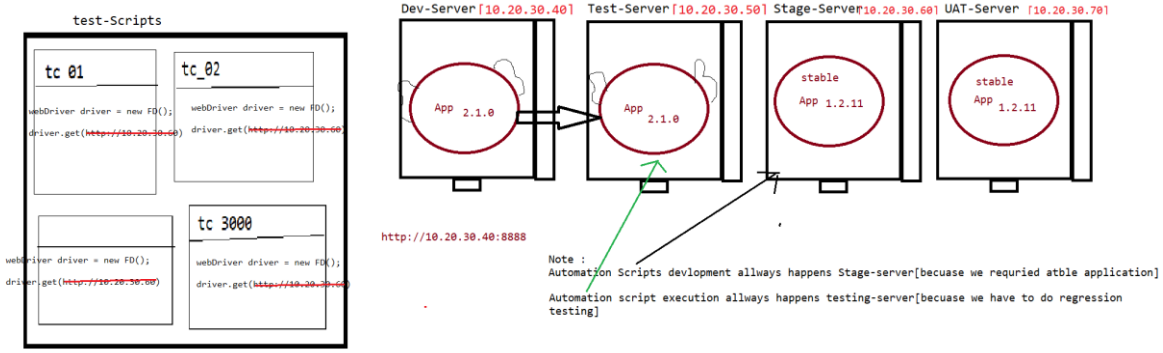
1. What is Data driven testing?

Read the data from external recourse & run the test is called Data driven testing (parameterization)



2. Why Data Driven testing?

As per the rule of the automation data shouldn't not hardcoded(fixed) with in a test scripts, because data modification & maintenance is tedious job when you want to run the test with different data, instead we should get the data from external resource like xlsx, .properties file , db , XML, JSON, CMD Line Data



Note : if we hardCode[fix] with in a testscripts , we can't run same test with different env/server without modification , but modification in 3000 testscript is tedious job

3. What is Advantages of Data driven testing
    1. Maintenance of the test data is easy
    2. Modification of the test data in external recourse is easy
    3. Cross browser /platform testing is easy (means change the browser in property File)
    4. Running test scripts in different Environment is easy
    5. Running test scripts in different credentials is easy
    6. We can create the test data prior the Suite execution (we can also get the data from testData team)
    7. Rerunning same test Script with multiple time with different data is easy

Data driven testing from Properties File

1. What is Properties File?

Properties is java feature file where we can store the data in from of key & values pair,

Key & value data type should be always string .

```
url http://10.20.30.40:8888
browser firefox
username root
password manager
timeout  10
```

data.properties

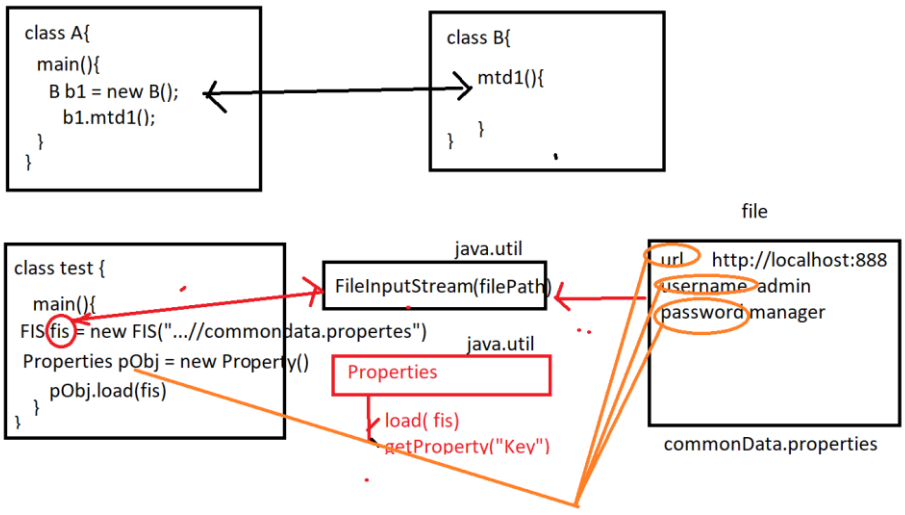
2. Why Property file ?

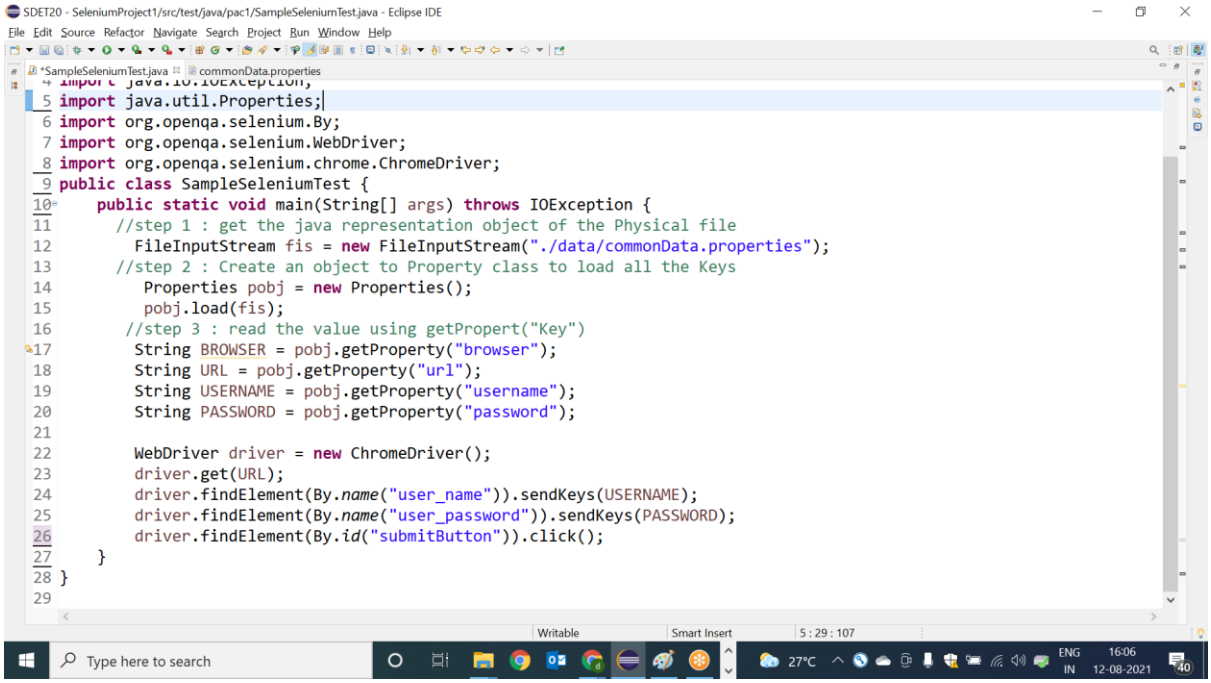
Property file is light weight & faster to read the data compare to any other file , & java as own Class to read the data from property

3. How to read data from properties File?

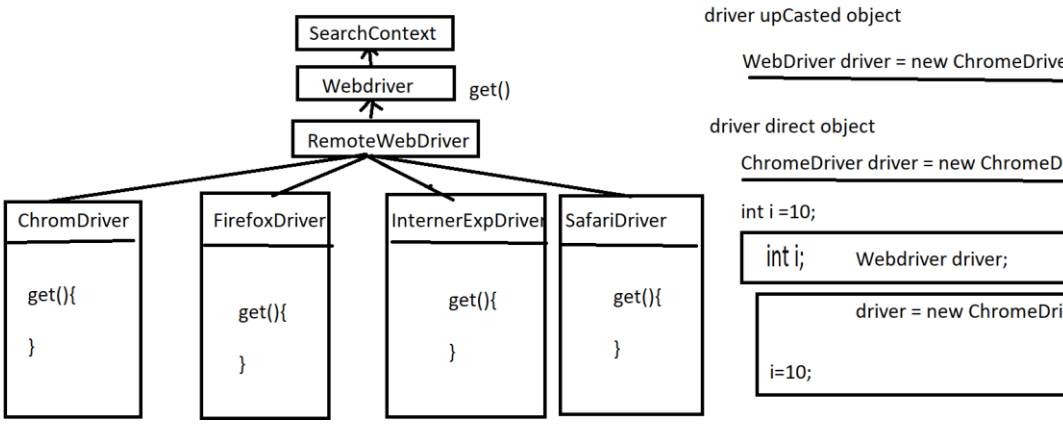
- Get the java representation Object of the Physical file using “FileInputSteam
- Create a Object of “Properties” class & load all the keys
- Read the data using getProperty(“Key”)

Note : properties file light weight & faster in execution compare to Excel





How to use Browser data in Seleniumtest



EG : best example for run time Polymorphism , driver object behave differently in run time

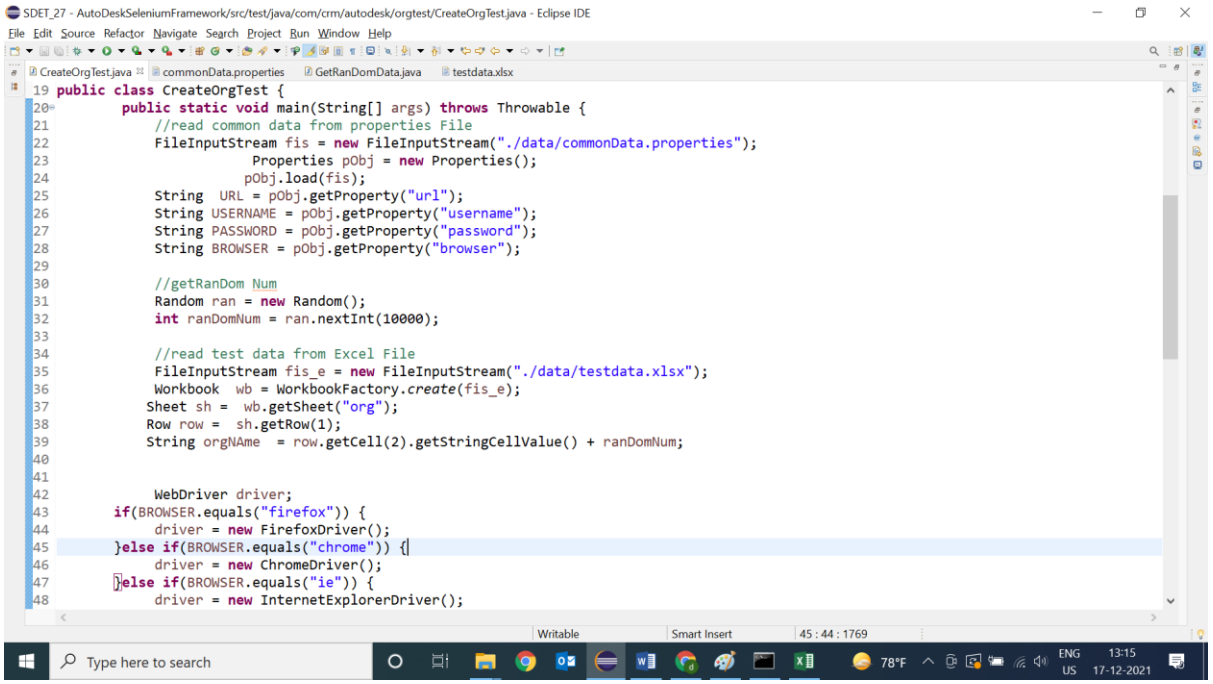
Assignment:

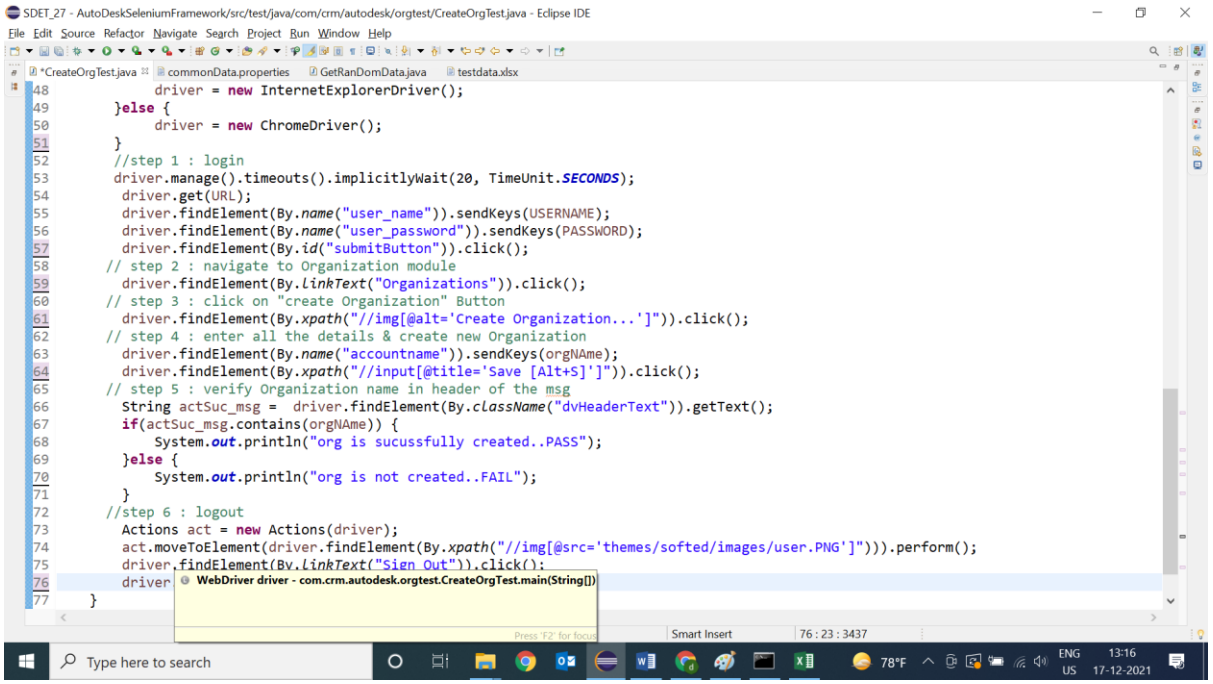
Test Case : CreateOrganization

step 1 : login
step 2 : navigate to Organization module
step 3 : click on "create Organization" Button
step 4 : enter all the details & create new Organization
step 5 : verify Organization name in header of the msg
step 6 : logout

Note :

1. Any data should not be hardcoded
2. test should able to run in different browser with minimal changes
3. test script data should get it from EXCEL sheet





Data driven testing from Excel File

➔Apache Poi is the open source libraries used to get & write data from all Microsoft documents like Excel , docx , ppt etc

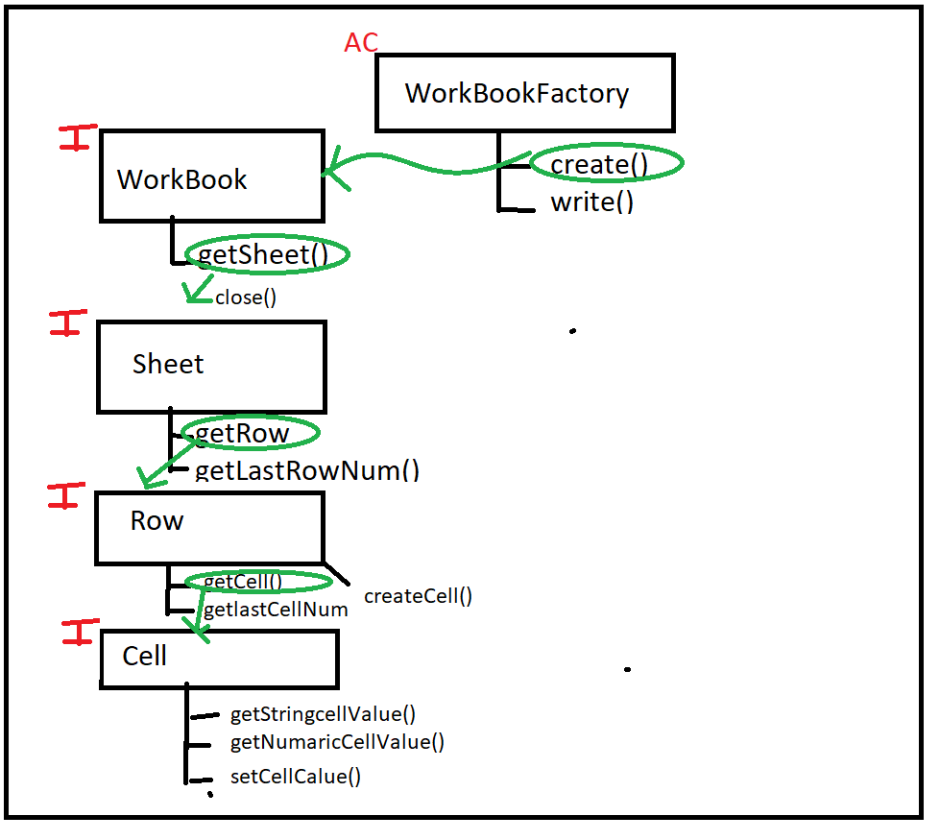
➔In real time most the company preferred the keep the test script data in Excel, because data will be in well-organized manner , so that modification & maintenance is easier.

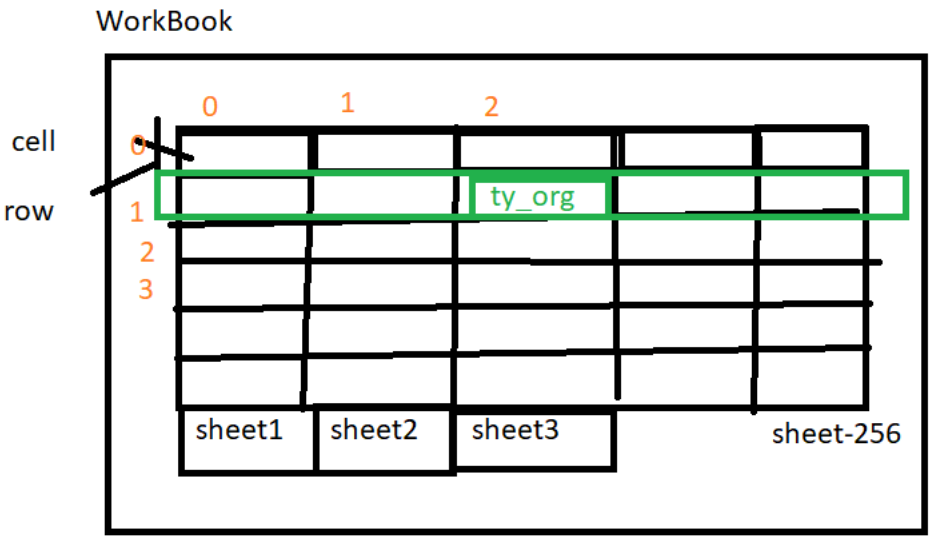
Installation steps:

1. Go the Maven Project
2. Edit POM.xml file
3. Go to https://mavenrepository.com
4. Search for apache-POI
5. Copy the dependency
6. Add dependency inside the <dependencies> in POM.xml

```
<dependency>
   <groupId>org.apache.poi</groupId>
   <artifactId>poi</artifactId>
   <version>4.0.0</version>
</dependency>
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi-ooxml</artifactId>
    <version>4.0.0</version>
</dependency>
```

Class diagram of Apache POI

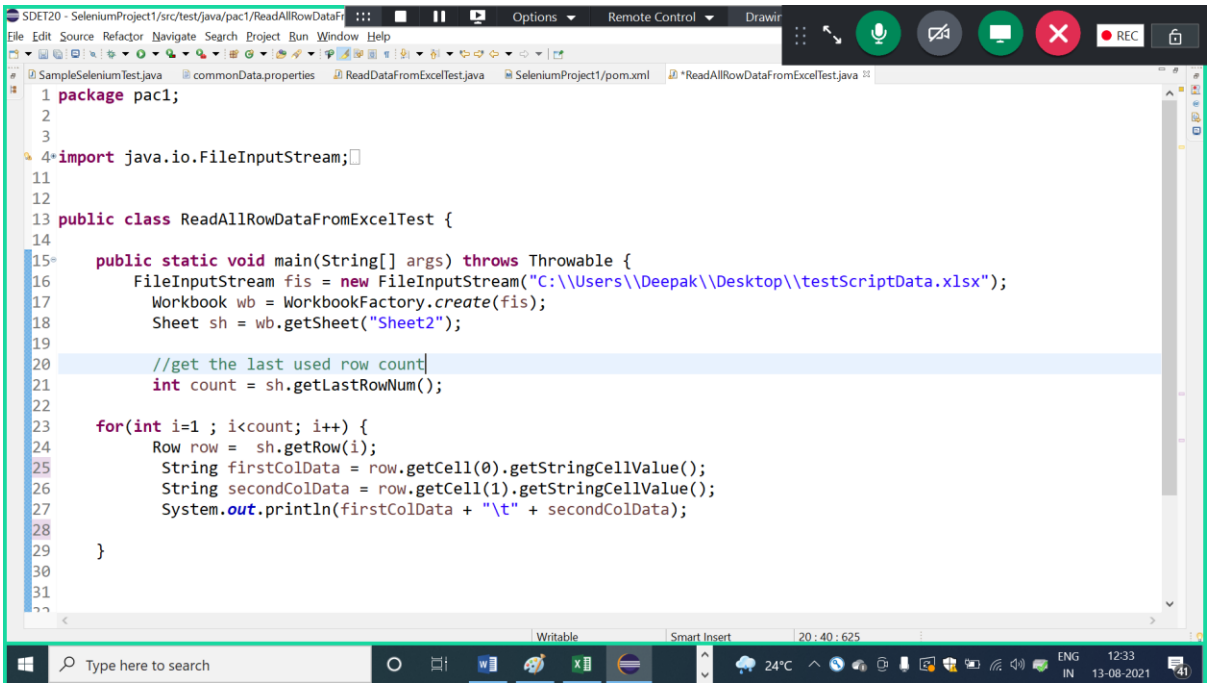




1. Program to read the data from WorkBook

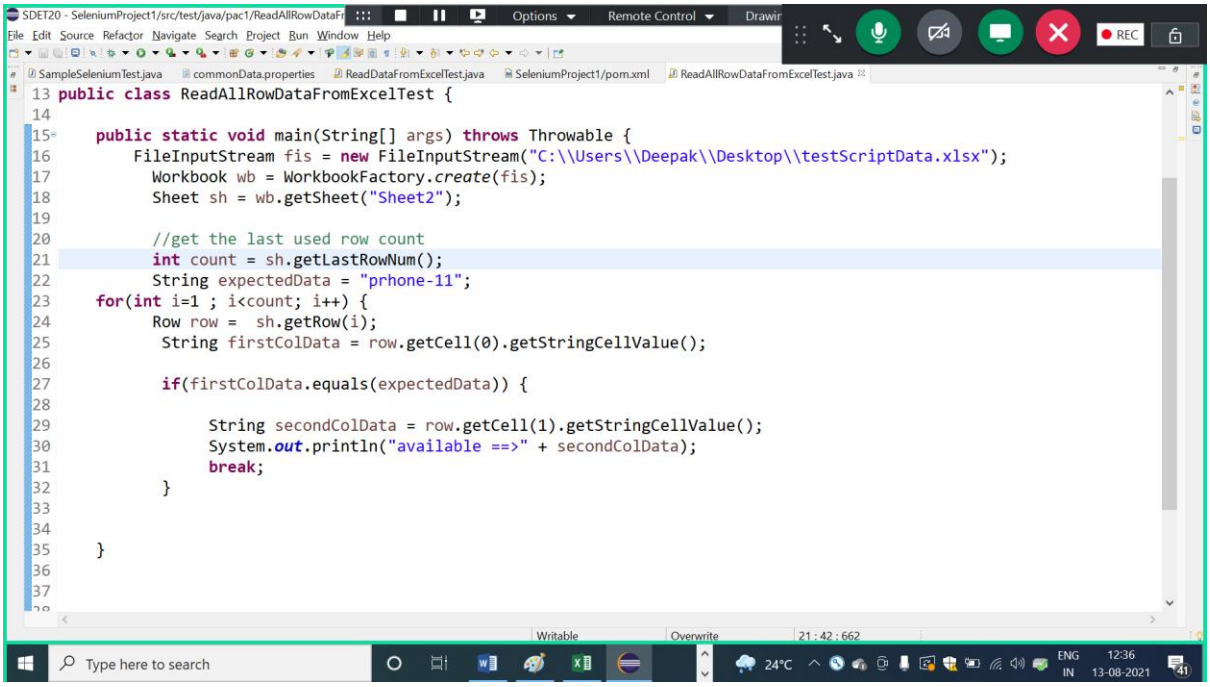
```java
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;


public class ReadDataFromExcelTest {

    public static void main(String[] args) throws Throwable {
        FileInputStream fis = new FileInputStream("C:\\Users\\Deepak\\Desktop\\testScriptData.xlsx");
        //step 1 : Open WorkBook in read mode
          Workbook wb = WorkbookFactory.create(fis);
        //step 2 : get the control of the Sheet-1
          Sheet sh = wb.getSheet("Sheet1");
        //step 3 : get the control of the 1 st Row
          Row row =  sh.getRow(1);
        //step 4 : get the control of the 2 nd Cell & copy the data
           Cell cel = row.getCell(2);
            String data = cel.getStringCellValue();
             System.out.println(data);
        //step 5 :close the WorkBook
             wb.close();
    }
}
```

2. Program to read 1 st & 2 nd coloum data from all the Rows



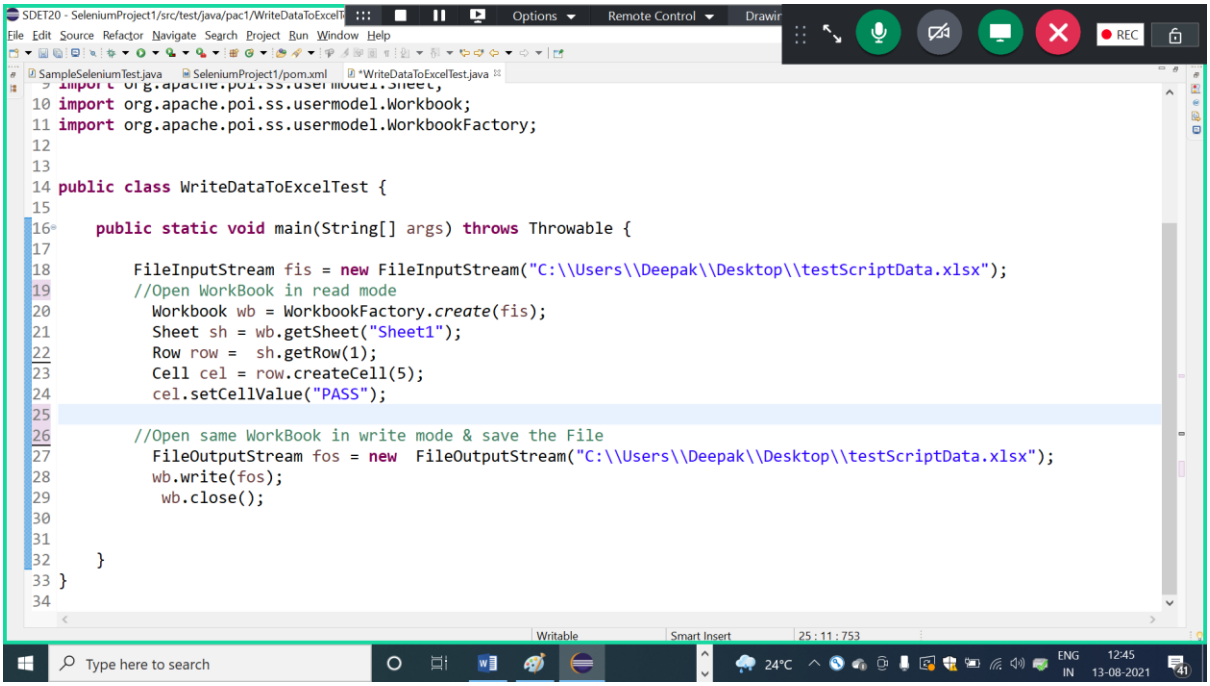
```java
package pac1;


import java.io.FileInputStream;


public class ReadAllRowDataFromExcelTest {

    public static void main(String[] args) throws Throwable {
        FileInputStream fis = new FileInputStream("C:\\Users\\Deepak\\Desktop\\testScriptData.xlsx");
          Workbook wb = WorkbookFactory.create(fis);
          Sheet sh = wb.getSheet("Sheet2");

          //get the last used row count
          int count = sh.getLastRowNum();

    for(int i=1 ; i<count; i++) {
          Row row =  sh.getRow(i);
           String firstColData = row.getCell(0).getStringCellValue();
           String secondColData = row.getCell(1).getStringCellValue();
           System.out.println(firstColData + "\t" + secondColData);

    }
```

3. Write a program to find specific data from 1 st column , if data is available then read & dispaly next cell data

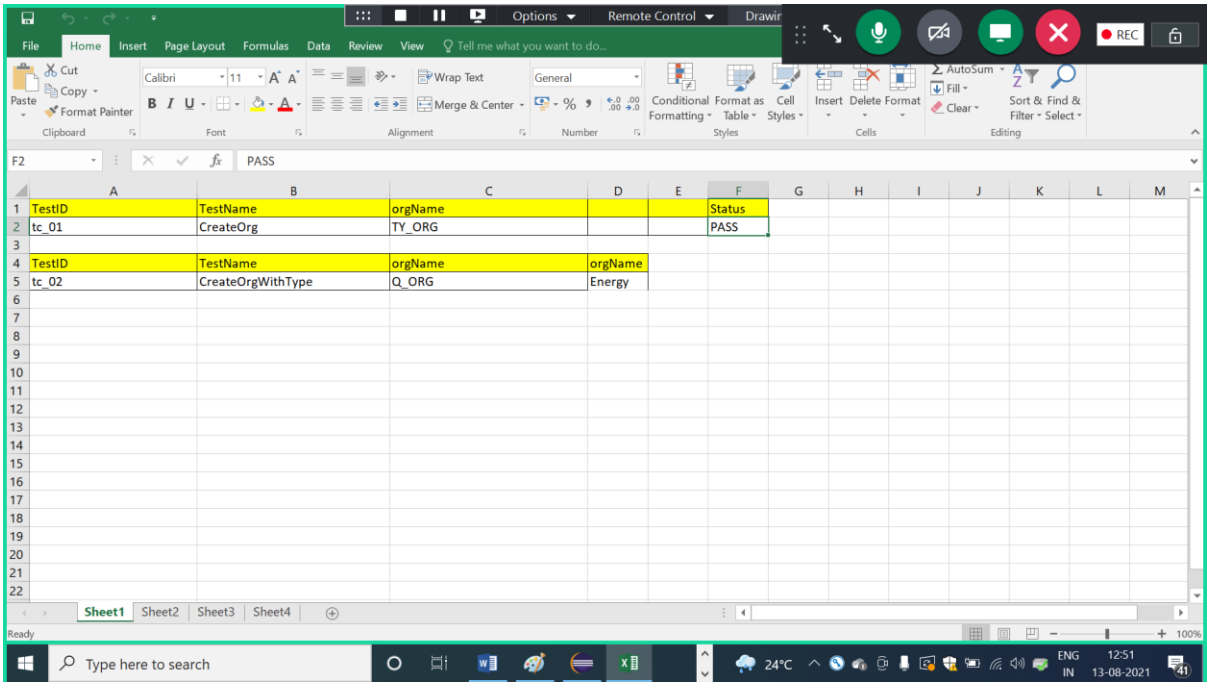


```java
public class ReadAllRowDataFromExcelTest {

    public static void main(String[] args) throws Throwable {
        FileInputStream fis = new FileInputStream("C:\\Users\\Deepak\\Desktop\\testScriptData.xlsx");
          Workbook wb = WorkbookFactory.create(fis);
          Sheet sh = wb.getSheet("Sheet2");

          //get the last used row count
          int count = sh.getLastRowNum();
          String expectedData = "prhone-11";
    for(int i=1 ; i<count; i++) {
          Row row =  sh.getRow(i);
           String firstColData = row.getCell(0).getStringCellValue();

           if(firstColData.equals(expectedData)) {

                String secondColData = row.getCell(1).getStringCellValue();
                System.out.println("available ==>" + secondColData);
                break;
           }


    }
```

4. Program to write data back to WorkBook/ Excel

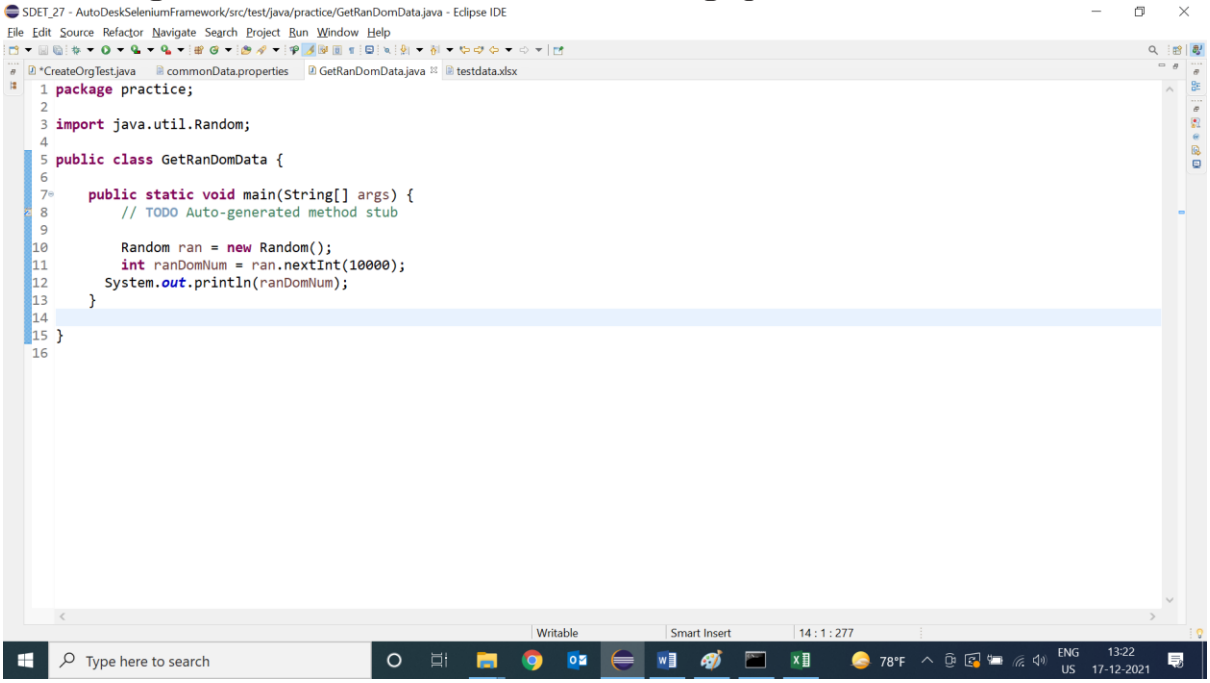


```
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;


public class WriteDataToExcelTest {

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

        FileInputStream fis = new FileInputStream("C:\\Users\\Deepak\\Desktop\\testScriptData.xlsx");
        //Open WorkBook in read mode
          Workbook wb = WorkbookFactory.create(fis);
          Sheet sh = wb.getSheet("Sheet1");
          Row row =  sh.getRow(1);
          Cell cel = row.createCell(5);
          cel.setCellValue("PASS");

        //Open same WorkBook in write mode & save the File
          FileOutputStream fos = new  FileOutputStream("C:\\Users\\Deepak\\Desktop\\testScriptData.xlsx");
          wb.write(fos);
           wb.close();


    }
}
```



	A	B	C	D	E	F
1	TestID	TestName	orgName			Status
2	tc_01	CreateOrg	TY_ORG			PASS
3						
4	TestID	TestName	orgName	orgName		
5	tc_02	CreateOrgWithType	Q_ORG	Energy		

How to generate random number using java



Data-Driven Testing from CommandLine – using Maven Parameters

Why Java Main method argument is String array ?

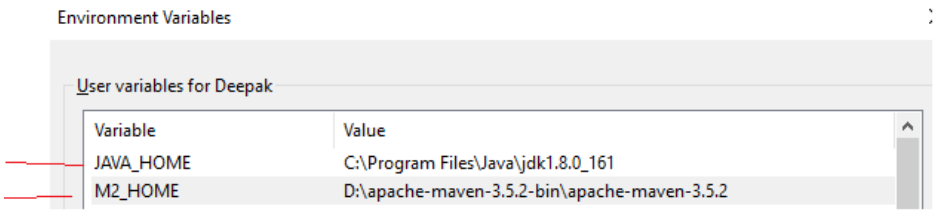
Because its always accept String array of data from the cammadline

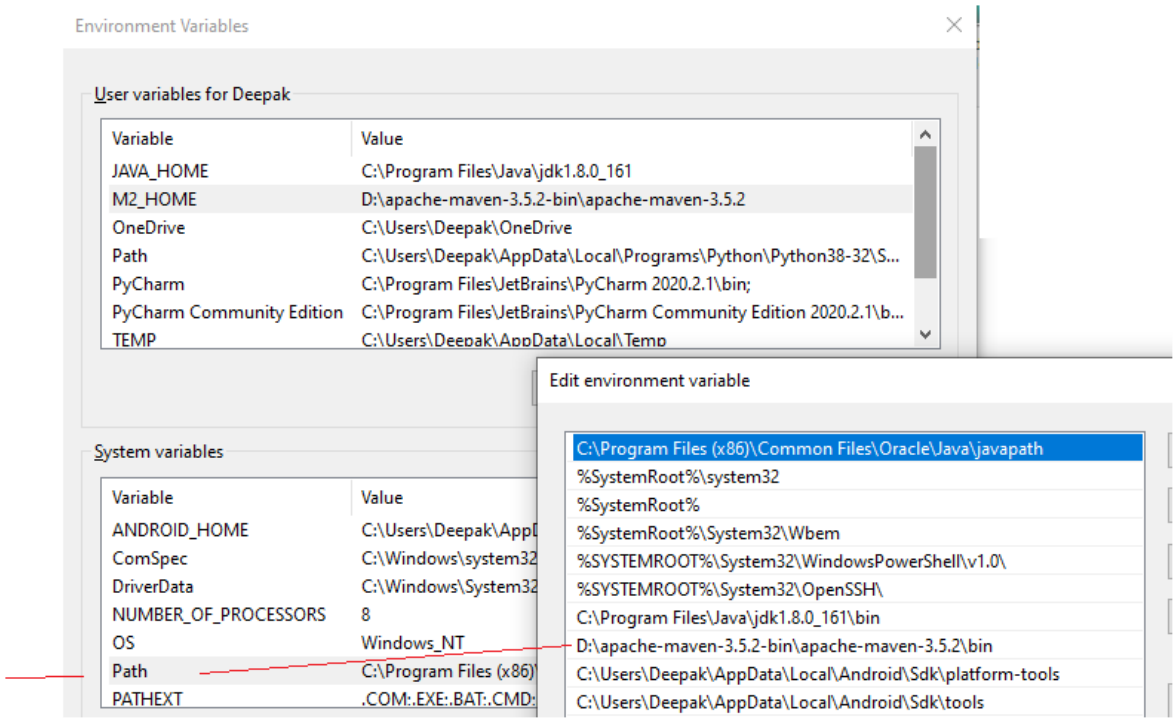
4. How to get Maven Parameters(data like url , username , password) from the CMD line

1. Download maven Command line plugin & set Environment variables

Intsllation steps

1. go to google
2. search download maven
3. click on first link
4. click on " apache-maven-3.6.3-bin.zip" beside "Binary zip archi"
5. download -ZIP & extract
6. get inside the folder & copy the "bin" path location
7. Go to Local system Env varibale window & set below path





8. go to cammond line , & execute below command to check maven installation

```
C:\Users\Deepak>mvn -version
Apache Maven 3.5.2 (138edd61fd100ec658bfa2d307c43b76940a5d7d; 2017-10-18T13:28:13+05:30
Maven home: D:\apache-maven-3.5.2-bin\apache-maven-3.5.2\bin\..
Java version: 1.8.0_161, vendor: Oracle Corporation
Java home: C:\Program Files\Java\jdk1.8.0_161\jre
Default locale: en_IN, platform encoding: Cp1252
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"
```

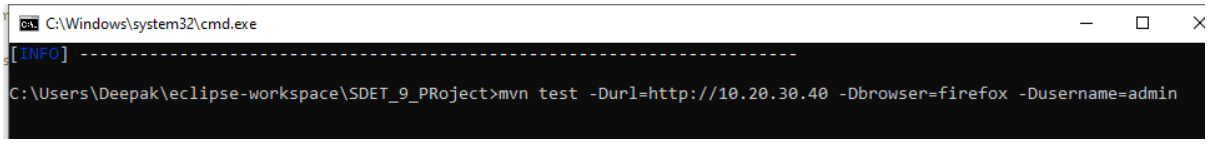
2. Create a Maven Project in Eclipse

3. Add testNG dependency in POM.xml

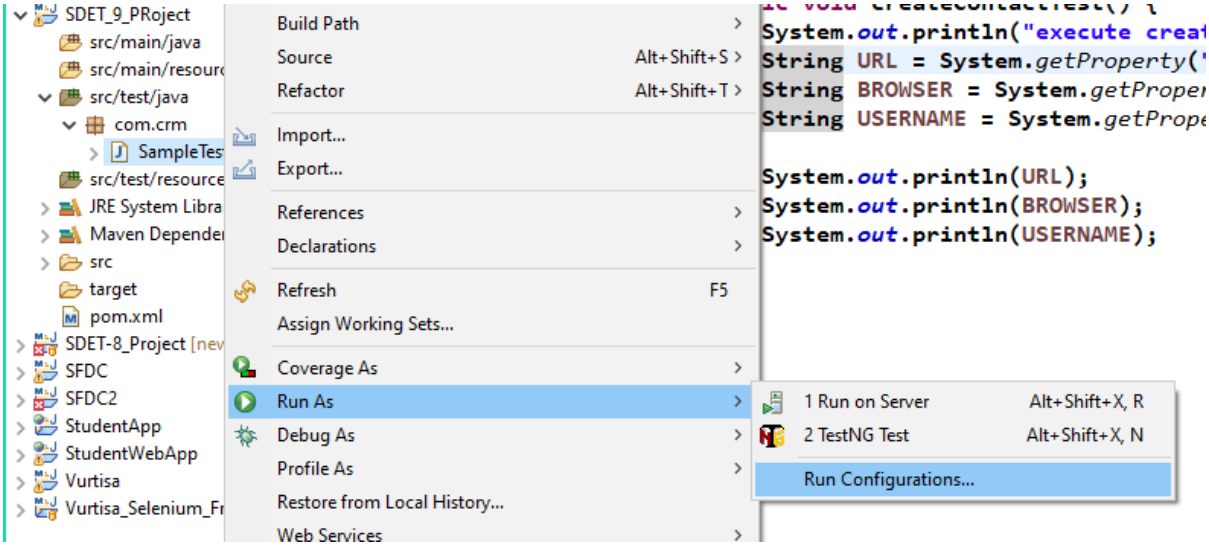
4. Create TestNg class

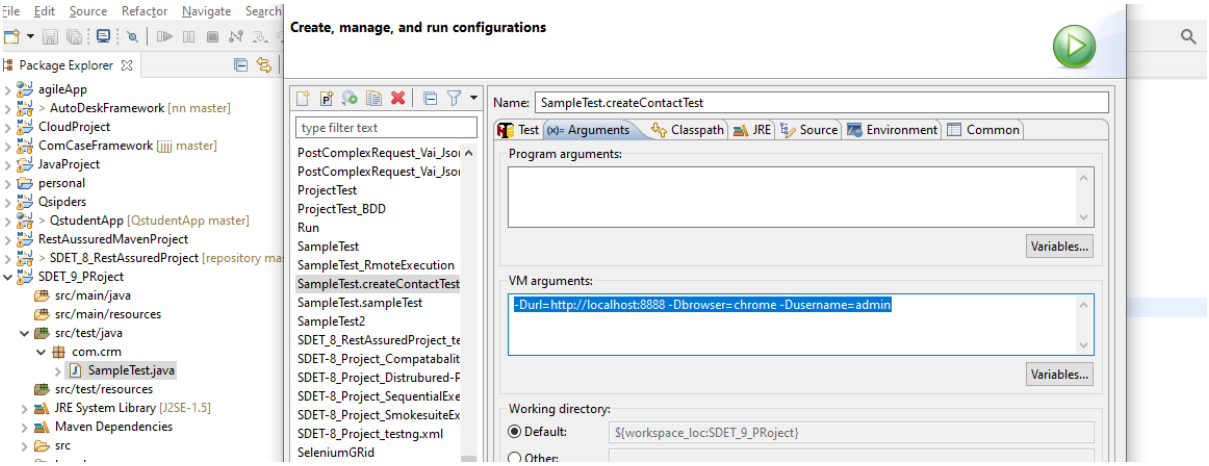
```
public class SampleTest {

    @Test
    Run | Debug
    public void createContactTest() {
        System.out.println("execute createContactTest");
        String URL = System.getProperty("url");
        String BROWSER = System.getProperty("browser");
        String USERNAME = System.getProperty("username");

        System.out.println(URL);
        System.out.println(BROWSER);
        System.out.println(USERNAME);

    }

}
```

5. Copy the location of the Maven Project
   Project=➔ RightClick➔Properties➔ Recourse➔ copy the ➔ location
6. Go to Cmd line
7. Go to the location of the Project
   Cmd >cd C:\Users\Deepak\eclipse-workspace\SDET_9_Project
8. Execute below Command



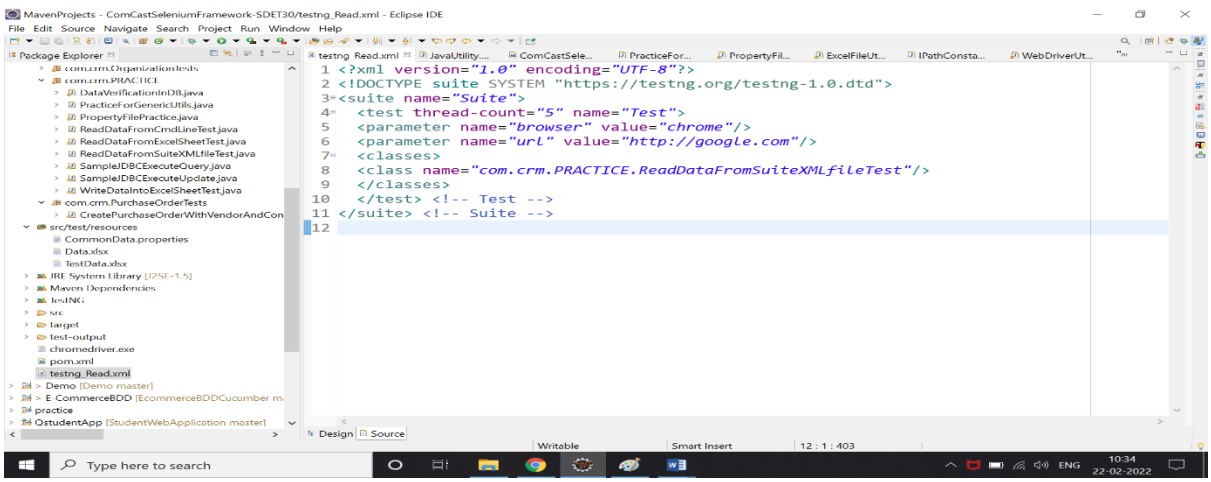
5. How to get Maven Parameters(data like url , username , password) from the Eclipse

1. 

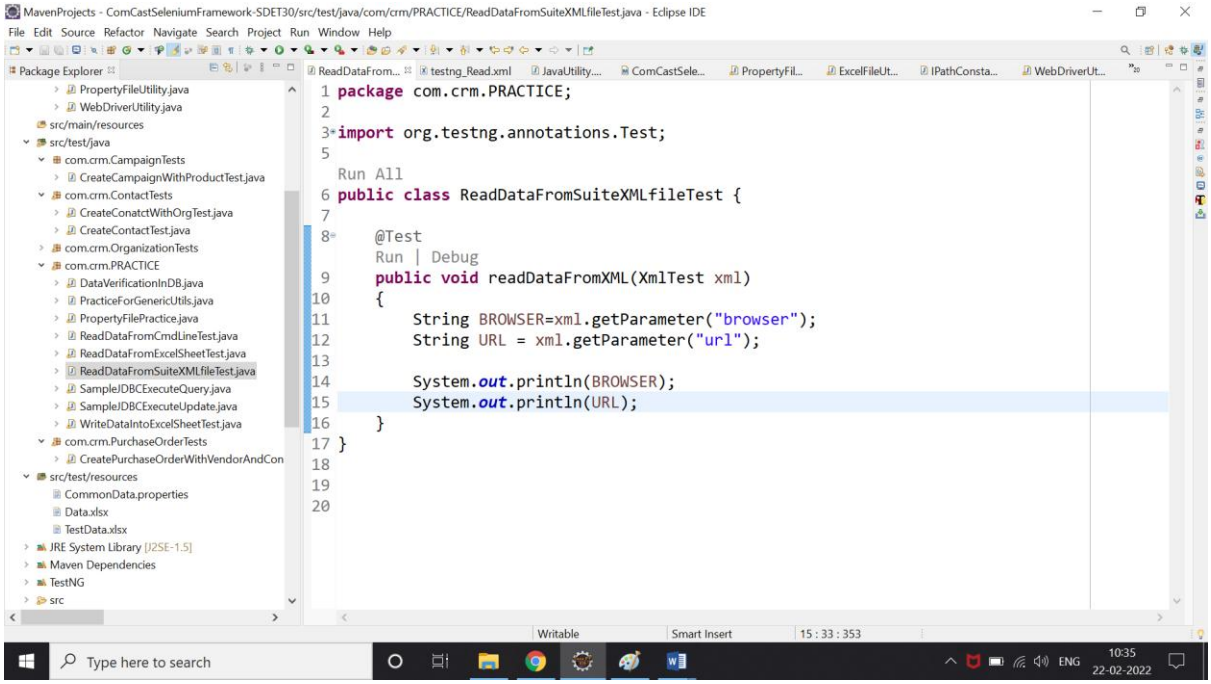


Data – Driven testing from Suite xml file

->Use parameter tag in suite xml file

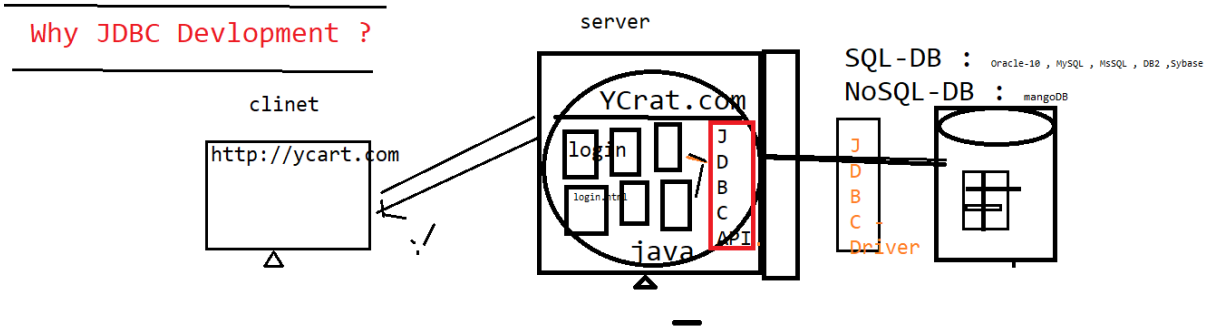


➔Using xmlTest parameterise the method to read from suite xml file to the test script with the help of getParameter().



What is JDBC ?

It's a J2AEE API , which is used to connect to any data base from the java code



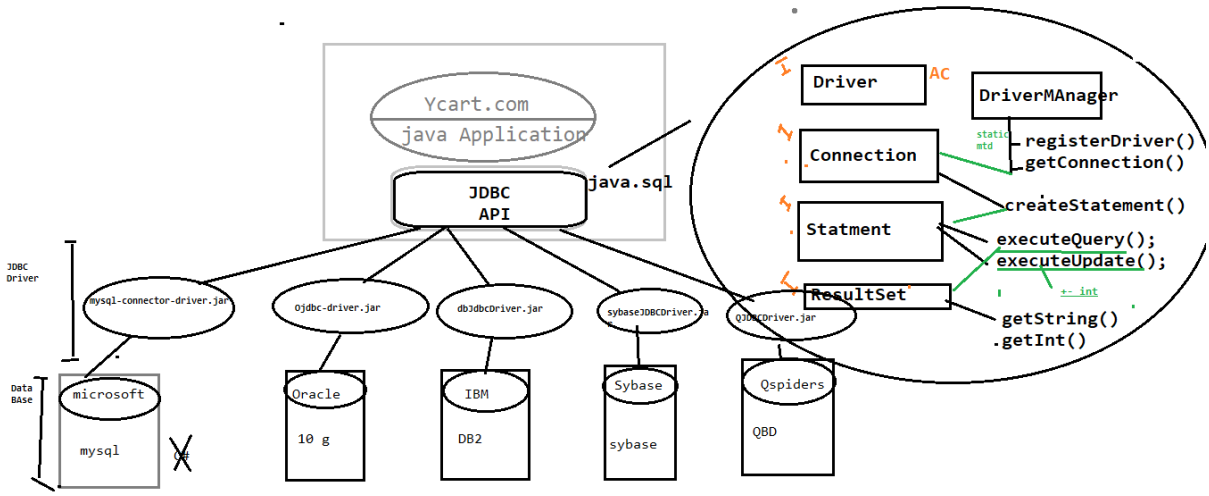
Every Application requried pysical database to store the data in the from table

Why JDBC in Devlopment ?

JDBC helps java program to store / retrieve data from the physical database

In Real time, every application required data base to store/retrieve the user information in data base

What is JDBC Drivers & JDBC API ?

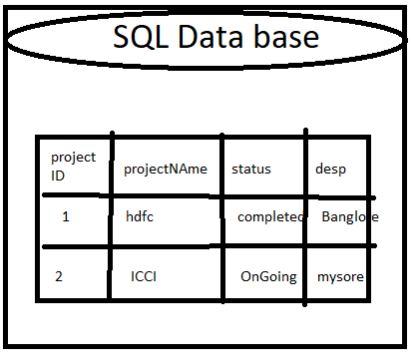


JDBC API : its an API's available in java.sql package , which is collection interfaces with abstract method & class , & it's database independent

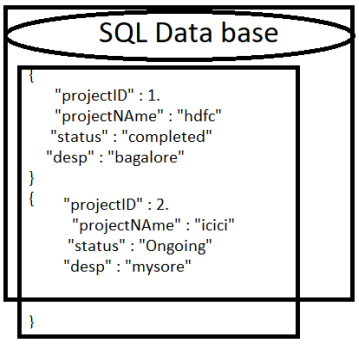
JDBC Driver : its implementation class of JDBC API provided by data base vender . it's a db dependent

There are types of database

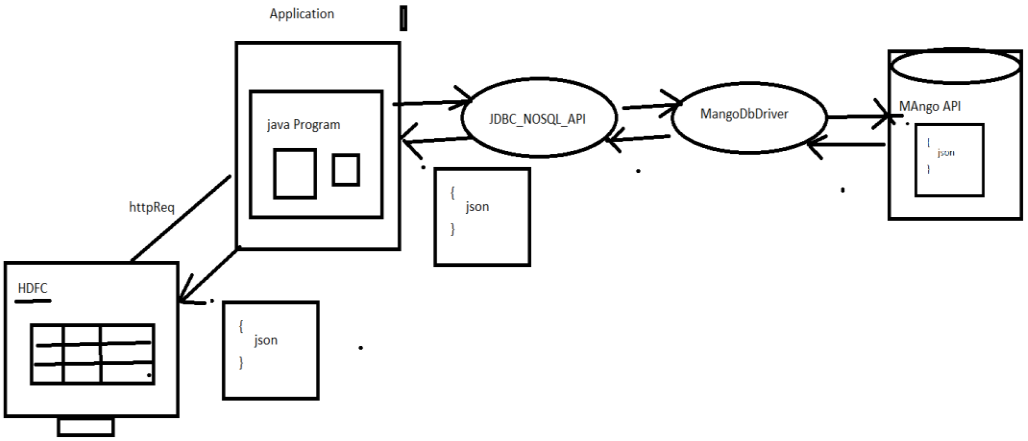
1. SQL Database
   ⇨ Data will be stored in form of table
   EG : mysql , MsSql , Oracle 11 g , DB2 ,



2. No SQL Database
   ⇨ Data will be stored in the from JSON format
   EG : Mango db , Casandra , NoSQL



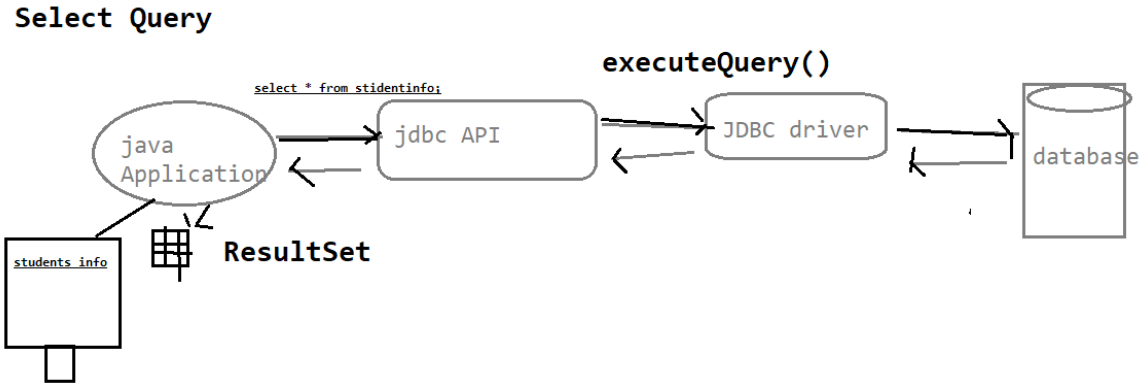
projectinfo.json

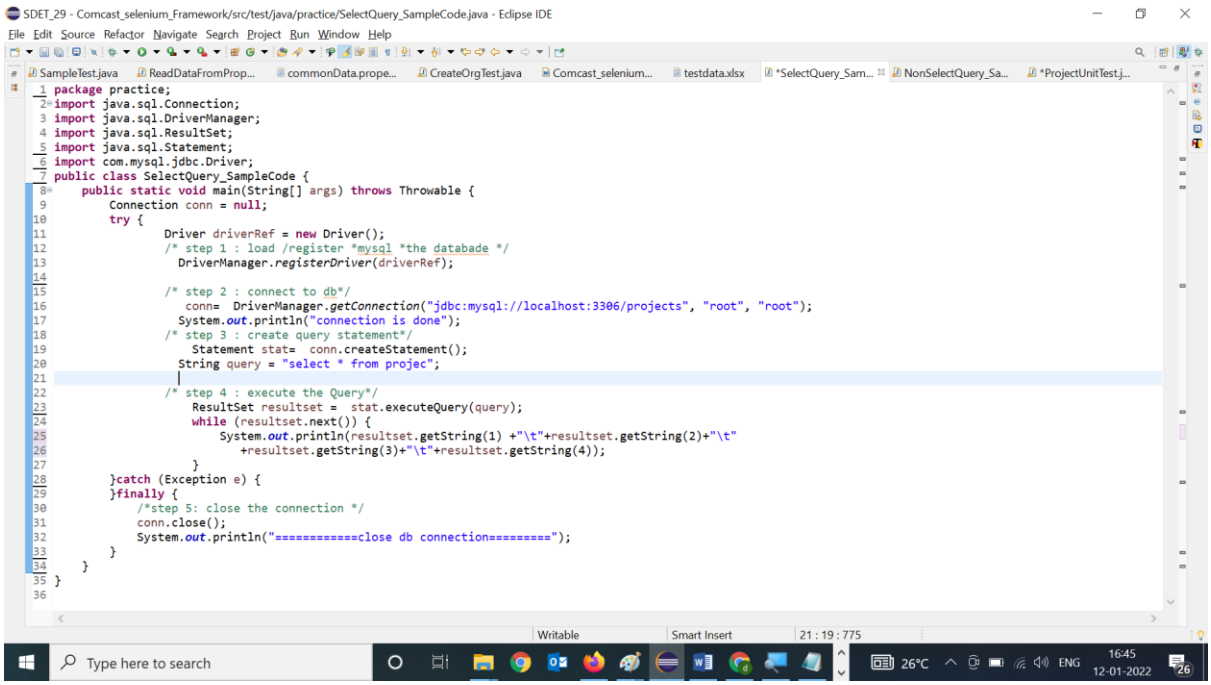


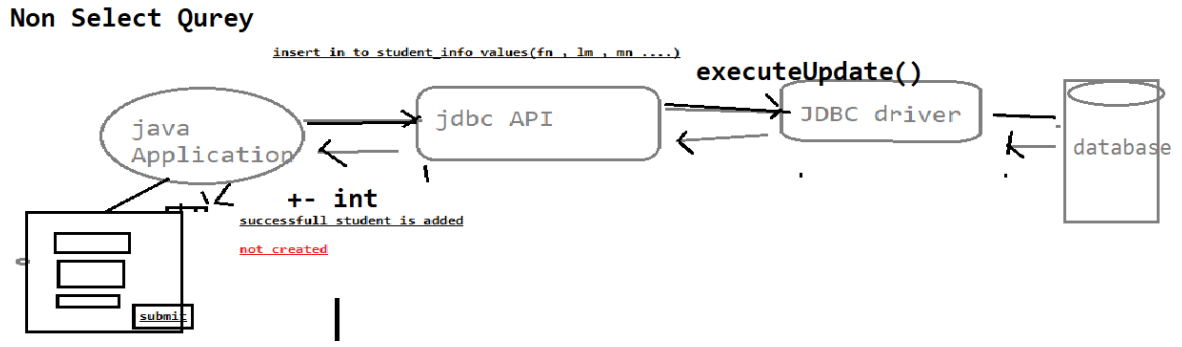
Dependency code to get MYSQL drivers

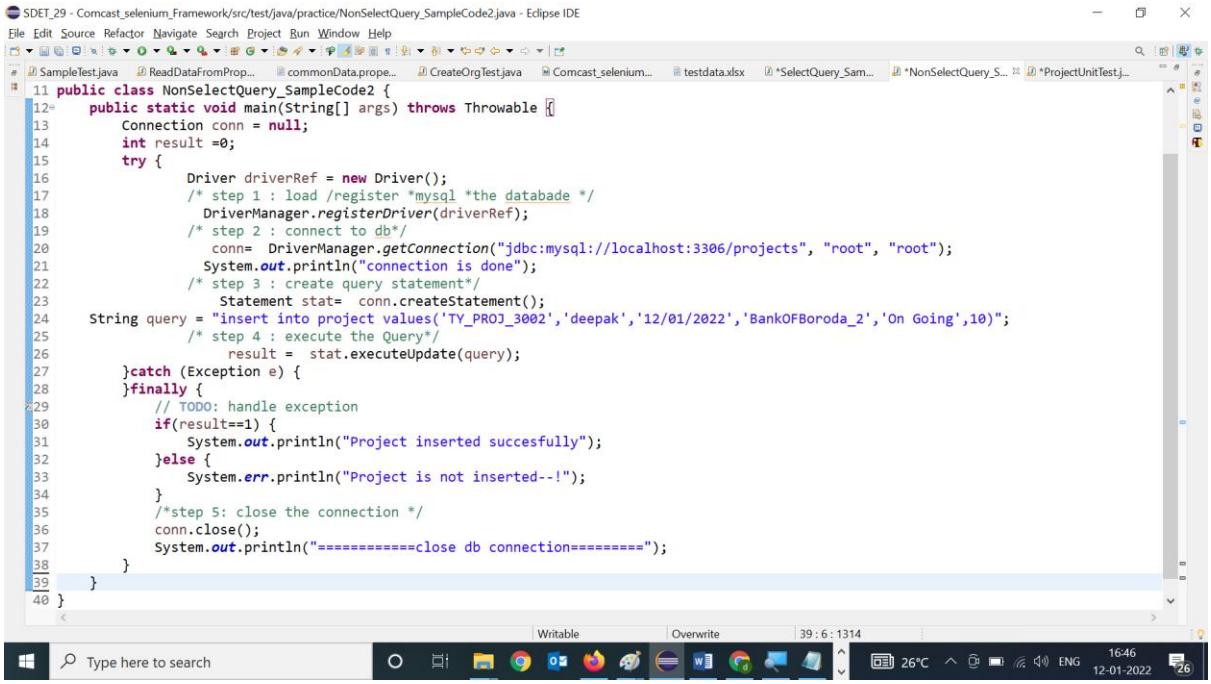
```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.27</version>
</dependency>
```

There are two types Query ?









Why JDBC in Automation ?

In Automation also required JDBC Connection , Most of the test case Preconditions , required few set of data in Database & also validation GUI WRT Database

Case 1:

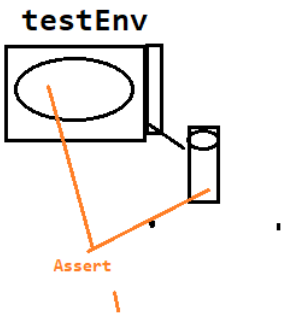
testCase : sendAmount

precondtion : FAN should have minimum 10 K balence

insert into account values (Acc1, active , 10000)

1. login to app
2. navigate to transaction page
3. enter FAN . TAN , Amount(6k) & click on send button
4. verify the sucessfull msg
   verify the balance in GUI
   verify the balance in Database

select * from account where accountNum = Acc1;



Case 2

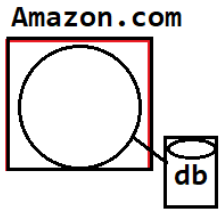
precondition : insert product in to database , which is req for testcase

@test

1.login to Amazon
2. enter *product* in serch feild
3. click on srach button
4. verify the product list & verify

testData

prodctName	cat	brand	qty
iphone12	apple	pple	10
iphone-7	mobile	apple	20
samsung	mobile	ase	15
moto			



Case :3

Test Case : crete Order

2 min

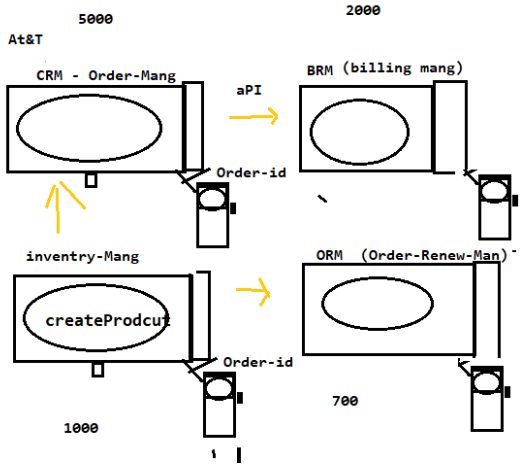
1. login to CRM
2. Go to Odrer page
3. create an Order
4. verify an Odrer

fk-010

Test Case : BillOnOder

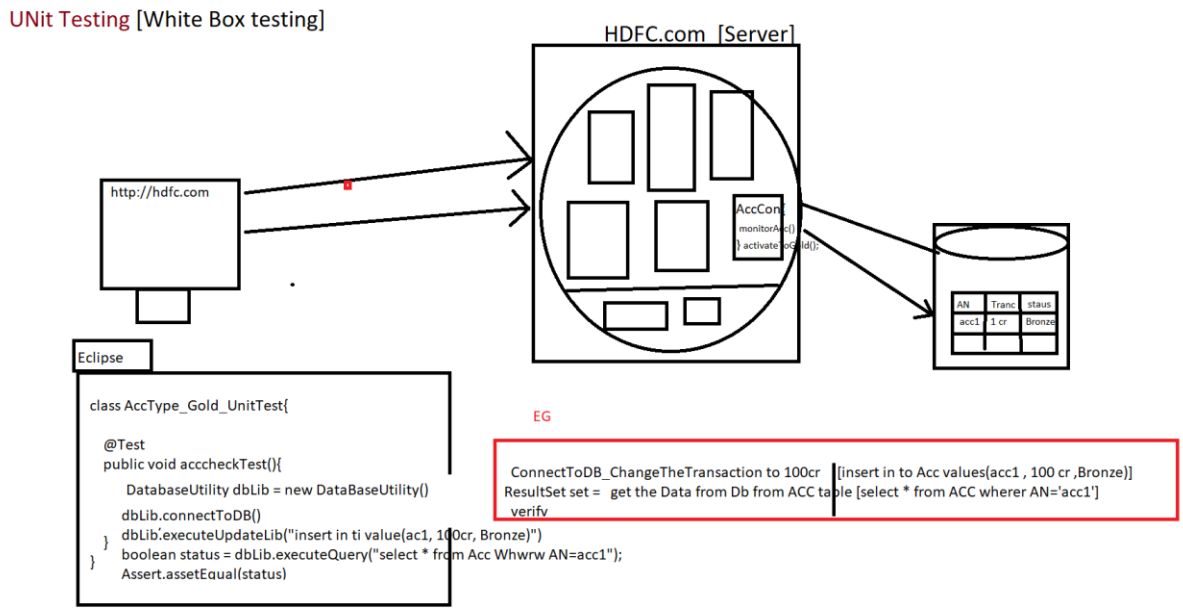
precondtion is : create an order in database via Query

1. login to BRM
2. Go to Billing Page
3. do payemnet
4. verify

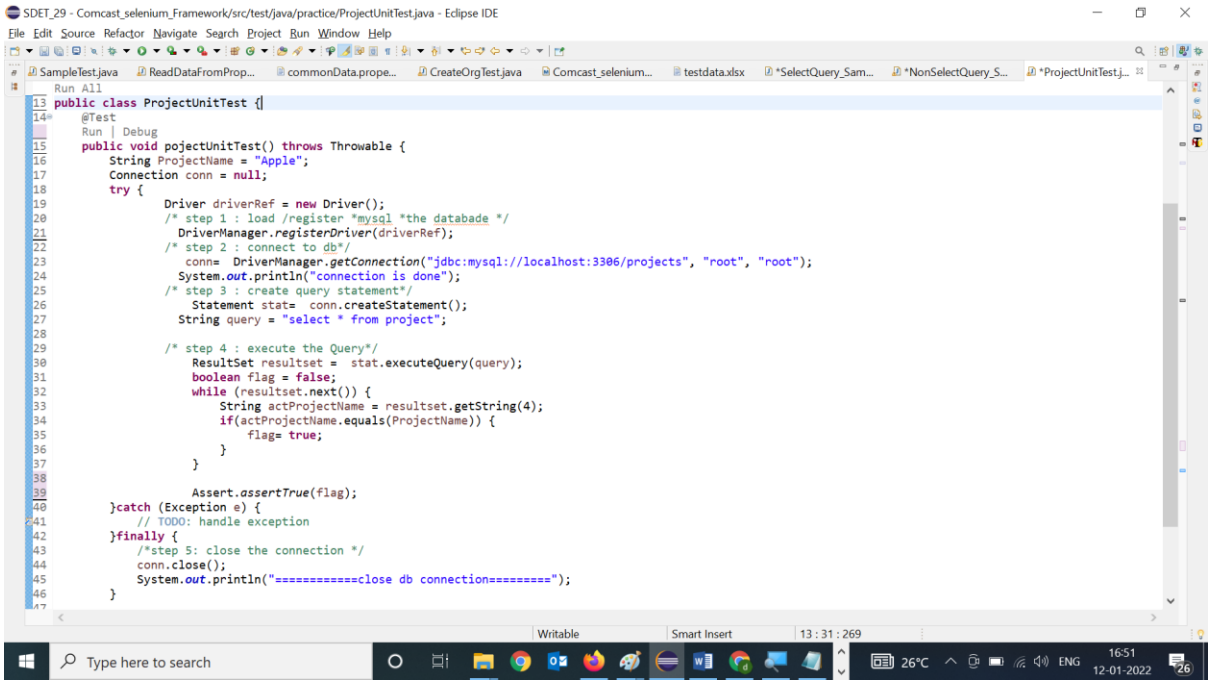


JDBC usage in UNIT Testing[White box Testing]

In case unit testing , [jdbc]database code is required to connect to database & get the data ten verify the data WRT to business logic



JDBC [Java Data base Connectivity]



Generic Libraries

What is Generic components in Automation Framework?

➔it's one of the automation framework components which is common for all the application

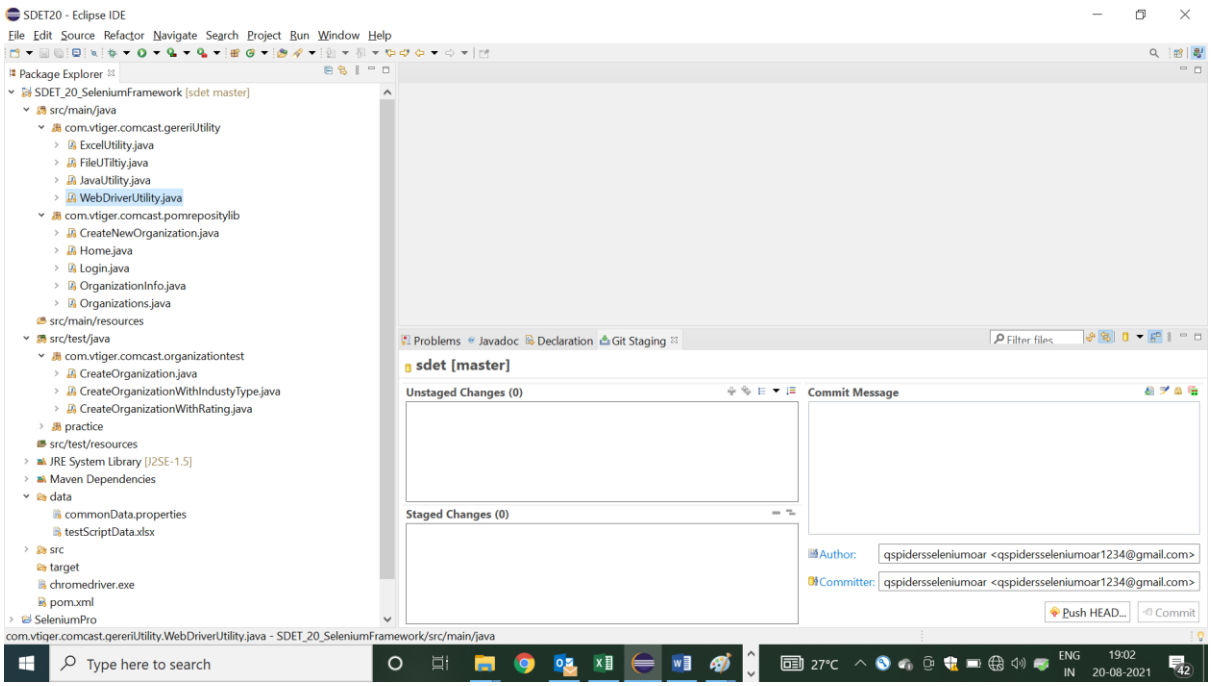
➔its collection of generic class contains reusable methods / libraries

➔The methods which can be used to any application is called Generic/common methods

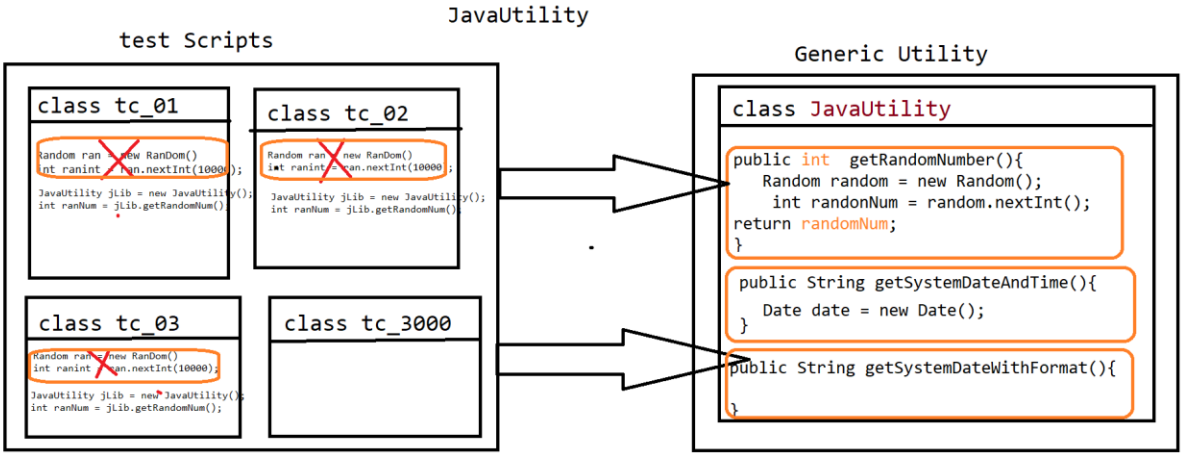
What is the advantages of Generic components?

- ⇨ Reusability of code
- ⇨ Code Optimization
- ⇨ Test script development is faster
- ⇨ Test scripts Code readability
- ⇨ Generic libraries are common to all automation projects
- ⇨ Avoid duplicate Code
- ⇨ no need to remember the syntax every time , just create once & use multiple times

Generic Utility Structure in Automation Project



1. Java Utility Libraries



➔ Java Utility is one class in generic component, which contain java specific methods which can be used across the test Scripts / Application

➔ its contains several generic reusable methods like

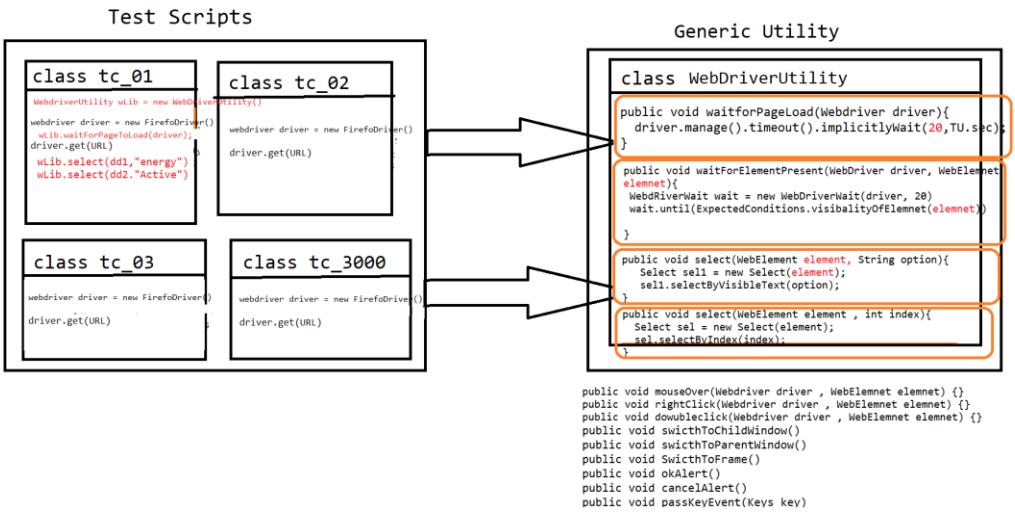
- ⇨ getRandomNum() : it's used to generate random number for every invocation
- ⇨ getSystemDate() : it's used to generate system date and time

==========Code==========

```
package com.vtiger.comcast.gereriUtility;

import java.awt.Robot;
import java.awt.event.KeyEvent;
import java.util.Date;
import java.util.Random;

/**
 * this class contains java specific generic libraries
 * @author Deepak
 *
 */
public class JavaUtility {

	/**
	 *  its used to generate the integer RanDom number with in the boundary of 0 to 10000
	 * @return intData
	 */
	public int getRanDomNumber() {
		Random ranDom = new Random();
		int ranDomNum = ranDom.nextInt(10000);
		return ranDomNum;
	}

	/**
	 * its used to get the current System date & time
	 * @return
	 */
	public String getSystmeDate() {
		Date date = new Date();
		String systemDateAndTime = date.toString();
		return systemDateAndTime;
```

```
		}
		/**
		 * its used to get the current System date with YYYY-MM-DD format
		 * @return
		 */
		public String getSystmeDate_YYYY_MM__DD() {
			Date date = new Date();
			String systemDateAndTime = date.toString();
			System.out.println(systemDateAndTime);
			String[] arr = systemDateAndTime.split(" ");
	String DD = arr[2];
	String YYYY = arr[5];
	int MM = date.getMonth()+1;

	String finalFromat = YYYY+"-"+MM+"-"+DD;
			return finalFromat;
		}

	/**
	 * used to pass Virtual Key to OS
	 * @throws Throwable
	 */
	public void pressVurtualEnterKey() throws Throwable {

		Robot rc=new Robot();
		rc.keyPress(KeyEvent.VK_ENTER);
		rc.keyRelease(KeyEvent.VK_ENTER);
	}

}
```

2. WebDriver Utility Libraries



- ⇨ WebdriverUtility is a Generic class , which contains webdriver specific reusable actions like
- ⇨ waitForPageToLoad()
- ⇨ waitForElement()
- ⇨ select()
- ⇨ accpertAlert()
- ⇨ cancelAlert() .Etc

==========Code==========

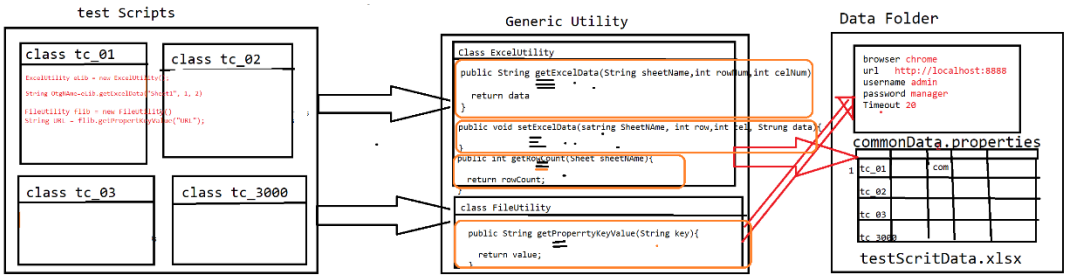
```
package com.vtiger.comcast.gereriUtility;

import java.awt.AWTException;
import java.awt.Robot;
import java.awt.event.KeyEvent;
import java.io.File;
import java.io.IOException;
import java.util.Iterator;
import java.util.Set;
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Keys;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;

import com.google.common.io.Files;
```

```
/**
 * This class contains webdriver specific generic methods
 * Deepak
 */

public class WebDriverUtility {

		/**
		 * this method wait for 20 sec for page loading
		 * @param driver
		 */
	public void waitUntilPageLoad(WebDriver driver)
	{
			driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

	}

	/**
	 * This method wait for the element to be visible
	 * @param driver
	 * @param element
	 *
	 */
	public void waitForElementVisibility(WebDriver driver,WebElement element)
	{
			WebDriverWait wait = new WebDriverWait(driver, 20);
			wait.until(ExpectedConditions.visibilityOf(element));
	}
	/**
	 * This method wait for the element to be clicked , its custom wait created to avoid elemenInterAceptable Exception
	 * @param element
	 * @throws throwable
	 *
	 */

	public void waitAndClick(WebElement element) throws InterruptedException
	{
			int count = 0;
	   while(count<20) {
				try {
				   element.click();
				   break;
				}catch(Throwable e){
						Thread.sleep(1000);
						count++;
				}
	   }

	}
	/**
	 * this methods enables user to handle dropdown using visible text
	 * @param element
	 * @param option
	 */

	public void select(WebElement element, String option)
	{
			Select select=new Select(element);
			select.selectByVisibleText(option);

	}

	/**
	 * this methods enables user to handle dropdown using index
	 * @param element
	 * @param index
```

```
 */

public void select(WebElement element, int index)
{
	Select select=new Select(element);
	select.selectByIndex(index);
}

/**
 * This method will perform mouse over action
 * @param driver
 * @param element
 */

public void mouseOver(WebDriver driver,WebElement element)
{
	Actions act = new Actions(driver);
	act.moveToElement(element).perform();

}

/**
 * This method performs right click operation
 * @param driver
 * @param element
 */
public void rightClick(WebDriver driver,WebElement element)
{
	Actions act = new Actions(driver);
	act.contextClick(element).perform();
}

/**
 * This method helps to switch from one window to another
 * @param driver
 * @param partialWinTitle
 */
public void switchToWindow(WebDriver driver, String partialWinTitle)
{
	Set<String> window = driver.getWindowHandles();
	Iterator<String> it = window.iterator();
	while(it.hasNext())
	{
		String winId=it.next();
		String title=driver.switchTo().window(winId).getTitle();
      if(title.contains(partialWinTitle))
      {
	break;
      }

	}

}
/**
 * Accept alert
 * @param driver
 */

public void acceptAlert(WebDriver driver)
{
	driver.switchTo().alert().accept();
}

/**
 * Cancel Alert
 * @param driver
```

```
 */
public void cancelAlert(WebDriver driver)
{
	driver.switchTo().alert().dismiss();
}
/**
 * This method used for scrolling action in a webpage
 * @param driver
 * @param element
 */
public void scrollToWebElement(WebDriver driver, WebElement element) {
	JavascriptExecutor js=(JavascriptExecutor)driver;
	int y= element.getLocation().getY();
	js.executeScript("window.scrollBy(0,"+y+")", element);
}


public void switchFrame(WebDriver driver,int index) {
	driver.switchTo().frame(index);
}

public void switchFrame(WebDriver driver,WebElement element) {
	driver.switchTo().frame(element);
}

public void switchFrame(WebDriver driver,String idOrName) {
	driver.switchTo().frame(idOrName);
}

public void takeScreenshot(WebDriver driver, String screenshotName) throws Throwable {
	TakesScreenshot ts=(TakesScreenshot)driver;
	File src=ts.getScreenshotAs(OutputType.FILE);
	File dest=new File("./screenshot/"+screenshotName+".PNG");
	Files.copy(src, dest);
}

/**
 * pass enter Key appertain in to Browser
 * @param driver
 */
public void passEnterKey(WebDriver driver) {
	Actions act = new Actions(driver);
	act.sendKeys(Keys.ENTER).perform();
}



}
```

3. Excel Utility libraries



- ⇨ As per the rule of automation, data should not be hardcoded with in the test scripts, so that to get the data from external file like Excel & .propertes file We go for ExcelUtility & FileUtlity
- ⇨ Excel Utility class is developed using apache Poi libraries, which is used to read the data from Excel
- ⇨ FileUtility is used to get the data from .properties file

==========Code==========

```
package com.vtiger.comcast.gereriUtility;

import java.io.FileInputStream;
import java.io.FileOutputStream;

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.ss.usermodel.WorkbookFactory;
/**
 * its developed using Apache POi libraries , which used to handle Microsoft Excel sheet
 * @author Deepak
 *
 */
public class ExcelUtility {
	/**
	 *  its used read the data from excel base don below arguments
	 * @param sheetName
	 * @param rowNum
	 * @param celNum
	 * @return Data
	 * @throws Throwable
	 */
	public String getDataFromExcel(String sheetName , int rowNum, int celNum) throws Throwable {
		FileInputStream fis  = new FileInputStream("./data/testScriptData.xlsx");
		Workbook wb = WorkbookFactory.create(fis);
		Sheet sh = wb.getSheet(sheetName);
		Row row = sh.getRow(rowNum);
		String data = row.getCell(celNum).getStringCellValue();
		wb.close();
		return data;
	}
	/**
	 * used to get the last used row number on specified Sheet
	 * @param sheetName
	 * @return
	 * @throws Throwable
	 */
	public int getRowCount(String sheetName) throws Throwable {
```

```
                FileInputStream fis  = new FileInputStream("./data/testScriptData.xlsx");
                Workbook wb = WorkbookFactory.create(fis);
                Sheet sh = wb.getSheet(sheetName);
                wb.close();
                return sh.getLastRowNum();
        }

        public void setDataExcel(String sheetName , int rowNum, int celNum ,String data) throws
Throwable {
                FileInputStream fis  = new FileInputStream("./data/testScriptData.xlsx");
                Workbook wb = WorkbookFactory.create(fis);
                Sheet sh = wb.getSheet(sheetName);
                Row row = sh.getRow(rowNum);
                Cell cel = row.createCell(celNum);
                cel.setCellValue(data);
                FileOutputStream fos = new FileOutputStream("./data/testScriptData.xlsx");
                wb.write(fos);
                wb.close();

        }
}
```

==========Code==========

```
package com.vtiger.comcast.gereriUtility;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.Properties;

/**
 *
 * @author Deepak
 *
 */
public class FileUTiltiy {
        /**
         * its used to read the data from commonData.properties File based on Key which you pass as an
argument
         * @param key
         * @throws Throwable
         */
    public String getPropertyKeyValue(String key) throws Throwable {
         FileInputStream fis = new FileInputStream("./data/commonData.properties");
         Properties pobj = new Properties();
         pobj.load(fis);
         String value = pobj.getProperty(key);
                return value;

    }
}
```

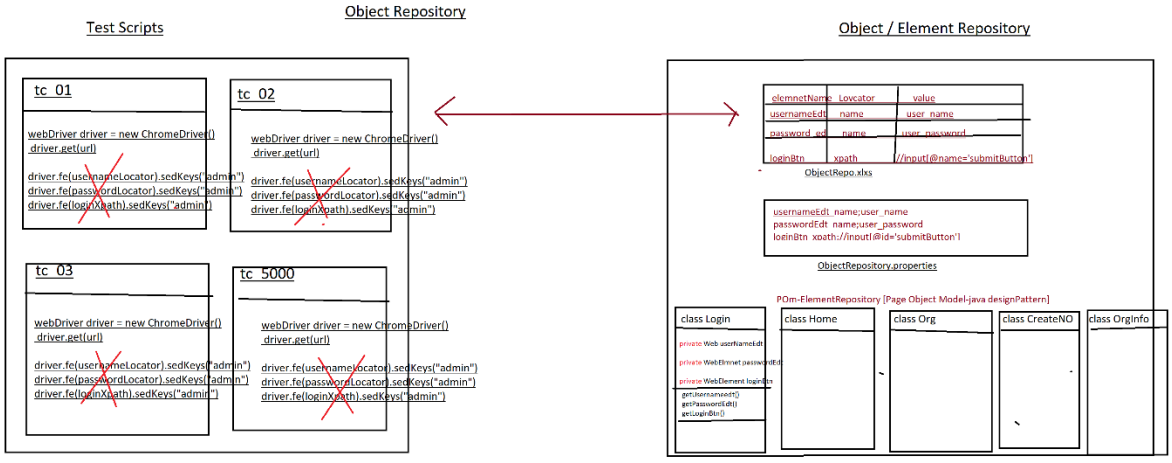
==========Sample Test Using Generic Utility=======

```
package com.vtiger.comcast.organizationtest;

import java.util.Random;
import java.util.concurrent.TimeUnit;

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

import com.vtiger.comcast.gereriUtility.ExcelUtility;
import com.vtiger.comcast.gereriUtility.FileUTiltiy;
import com.vtiger.comcast.gereriUtility.JavaUtility;
import com.vtiger.comcast.gereriUtility.WebDriverUtility;
import com.vtiger.comcast.pomrepositylib.CreateNewOrganization;
import com.vtiger.comcast.pomrepositylib.Home;
import com.vtiger.comcast.pomrepositylib.Login;
import com.vtiger.comcast.pomrepositylib.OrganizationInfo;
```

```java
import com.vtiger.comcast.pomrepositylib.Organizations;

public class CreateOrganization {

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

	  /*Object Creation for Lib*/
		JavaUtility jLib = new JavaUtility();
		WebDriverUtility wLib = new WebDriverUtility();
		FileUTiltiy fLib = new FileUTiltiy();
		ExcelUtility eLib = new ExcelUtility();

		int randomInt = jLib.getRanDomNumber();

		/*common Data*/
		String USERNAME = fLib.getPropertyKeyValue("username");
		String PASSWORD = fLib.getPropertyKeyValue("password");
		String URL = fLib.getPropertyKeyValue("url");
		String BROWSER = fLib.getPropertyKeyValue("browser");

		/*test script Data*/
		String orgName = eLib.getDataFromExcel("Sheet1", 1, 2) + randomInt;

		/* Navigate to app*/
		WebDriver driver = new ChromeDriver();
		wLib.waitUntilPageLoad(driver);
	driver.get(URL);

	/* step 1 : login */
	Login loginPage = new Login(driver);
	loginPage.loginToApp(USERNAME, PASSWORD);

	/*step 2 : navigate to organization*/
	Home homePage = new Home(driver);
	homePage.getOrganizationLnk().click();

	/*step 3 : navigate to "create new organization"page by click on "+" image */
	Organizations orgPage = new Organizations(driver);
	orgPage.getCreateOrgImg().click();

	/*step 4 : create organization*/
	CreateNewOrganization cno = new CreateNewOrganization(driver);
	cno.createOrg(orgName);

	/*step 5 : verify the successful msg with org name*/
	OrganizationInfo orginfoPage = new OrganizationInfo(driver);
	String  actSuccesfullMg =  orginfoPage.getSuccesfullMsg().getText();
	if(actSuccesfullMg.contains(orgName)) {
	System.out.println(orgName + "==>created successfully");
	}else {
	System.out.println(orgName + "==> not created successfully");

	}
	/*step 6 : logout*/
	homePage.logout();
	}

}
```

1. What is Object/Elements/POM Repository

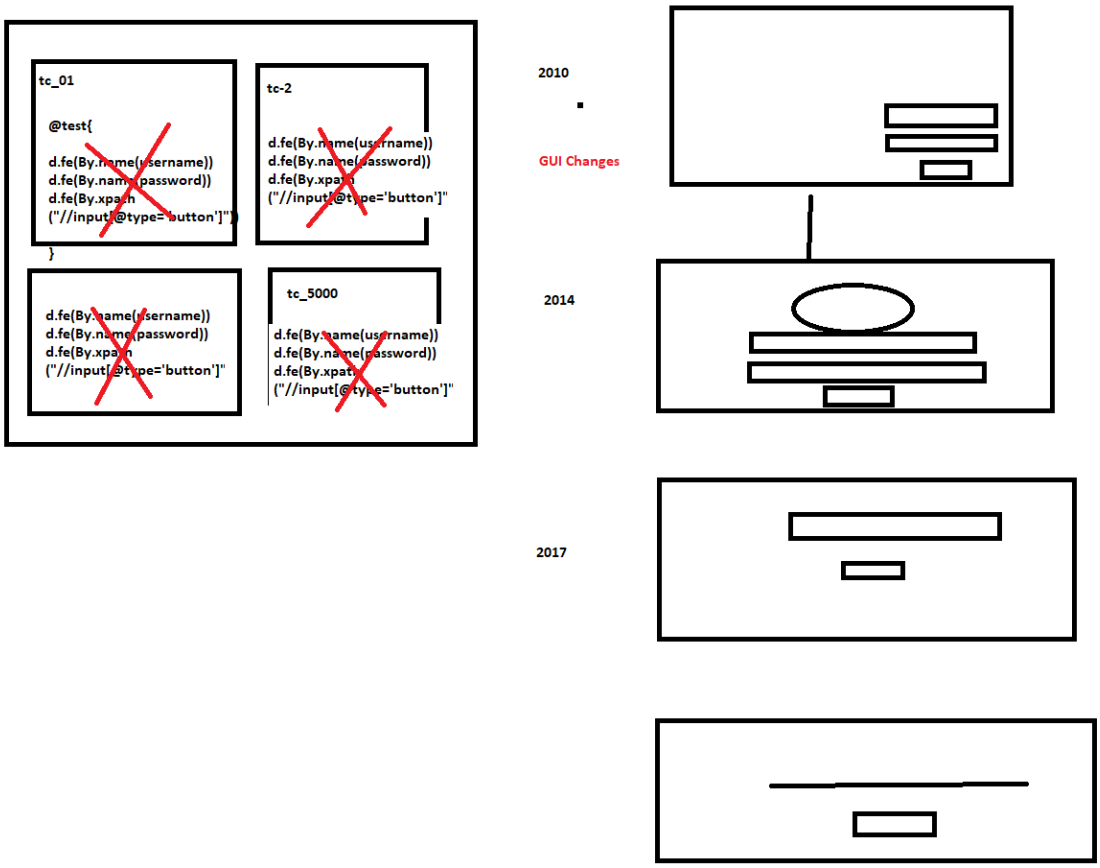
Its collection of elements locators & business libraries in one place & its developed using POM design pattern



2. Why Object repository ?

As per the rule of the automation, we should not hardcode[fixed]elements with in test Scripts instead, we should get elements from Object Repository , because in Agile process due to frequent requirement changes , modification & maintenance of elements is tedious job

EG : below is the example of Gmail application GUI changes



3. What is the advantages repository?
    a. Reusability of elements, no need to write xpath & other locators again & again
    b. Modification in Repository is easy, when GUI changes frequently
    c. Maintenance is easy, because all the elements we kept in one place
    d. Test Script Code Optimized via business reusable libraries
    e. More Readability
    f. Test Script development is faster due to business lib
    g. Test Script is more robust
    h. Handle Stale Elements Exception.
4. What is POM?

POM is a java design pattern preferred by google to develop object repository.

5. Why POM ?

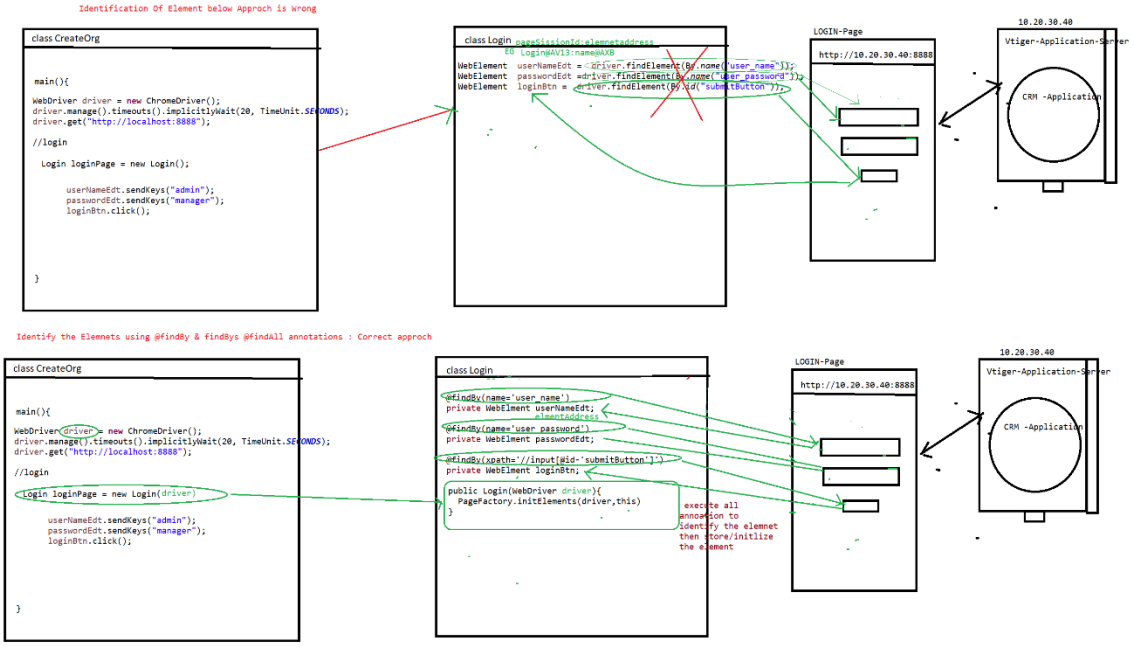
It's a well-organized structured design pattern, where we can maintain all the web elements in page wise, due to POM design pattern maintains & modification is easy & faster.

6. Advantages of POM:
    1. Well organized structure
    2. Handle stale element exception.
    3. maintains & modification of element is easy
    4. We can directly store Web Elements in java class
    5. Better fit for Agile processes
    6. Support Auto heal feature

7. Why @FindBy(locator) annotation instead of driver.findElement("locator")

   Ans : to avoid staleElementReferenceException

8. What is staleElementReferenceException ?
   It's one of selenium Execption , whenever webdriver try to identify an element , element was available in GUI, but at time of performing an action on the elements element was not recognized due to page got refreshed or elements may become old or element not attached to page in such case we get staleElementReferenceException



9. Rules of POM

Rule 1 : create separte java class for every page in a application & class name should be same page name

Rule 2 : Identify all the elements using @findBy & @findAll , @findbys annotations & store them in speific pom / java class (Element declartion)

Rule3 : For Every POM class create Constructor to get an Object of the class & initialize the Page Elements , in order to initialize all the page Elements we should use Pagefactory.initElement() (Element initialization)

Rule 4 : declare all the WebElements as private & provide getters methods to accesses elements in testScripts class [this processes is called Encapsulation]

Note : to create getters mtds inside the java class fallow below steps

=>place cursor inside the class➔ Right click➔source ➔generate getters & setters ➔ select the getters check box ➔ click on ok button

Rule 5 : Go to every & identify the reusable business libraries & implement them in same POM class

10. Difference between POM & PageFactory design pattern?

POM is java design pattern, where will maintain all the Web element locator in well-organized manner

Pagefactory it's an extended design pattern of POM , which is used to create an Object to POM classes , & at the time of object creation it will execute all @findBy @findbys annotation then initialize all the elements

Difference between @findBy , @findAll &@findBys annotation

All annotation available in Selenium webdriver, its traditional ways to identify the elements in GUI.

@findBy : used to identify the element using one locator or one condition

@findAll : it contains multiple @findBy annotation , it mean we can identify the same element using multiple locator (multiple conditions) , it will use OR condition during execution of locator

```
@findALL({ @findBy(@id='username') , @findBy(name='user')})
Private Webelements userNAmeEdt;
```

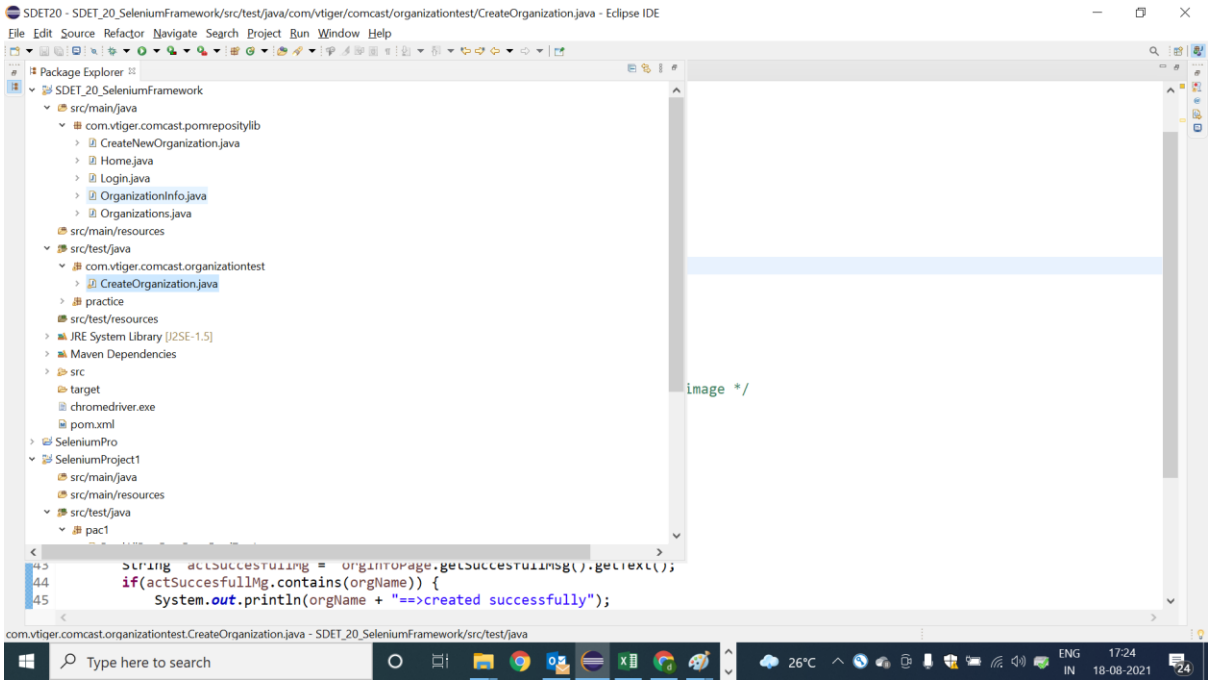
Note : using above concepts we can achieve **Autohealing** technique

AutoHealing : during execution , if one locator fails to identify the element , it will retry to identify the same element using another locator

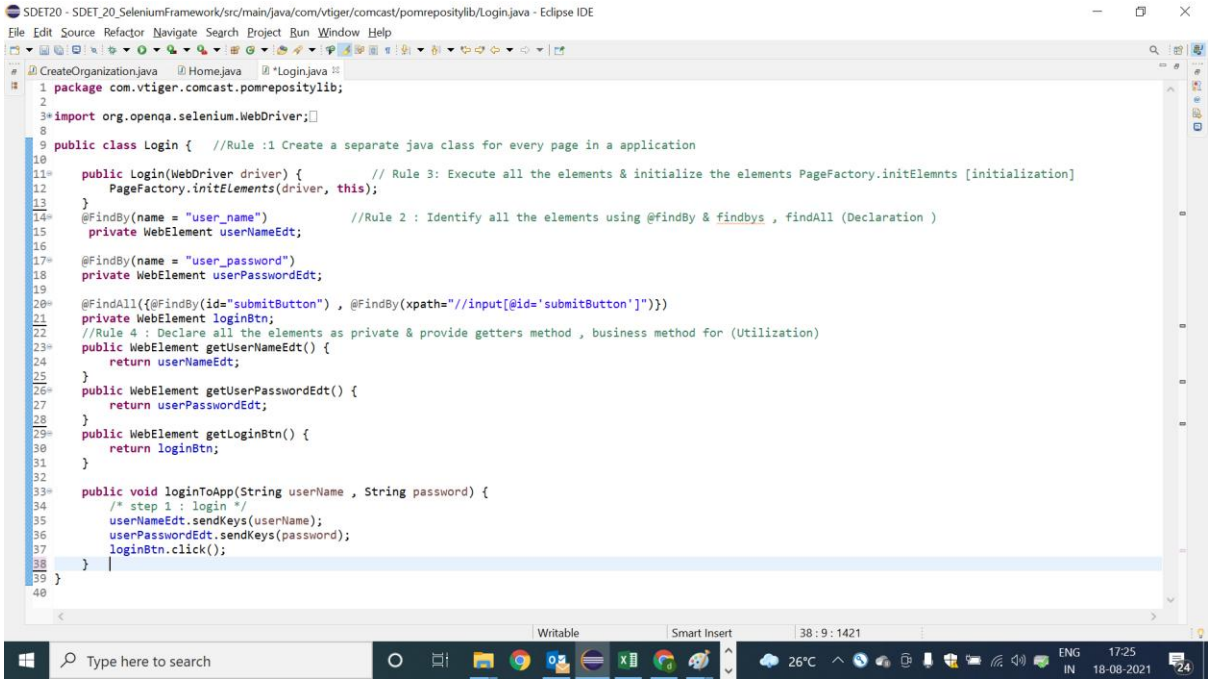
@FindBys : it contains multiple @findBy annotation , it mean we can identify the elements using multiple locator (multiple conditions) , it will use AND condition to during execution of locator

```
@findBys({ @findBy(@id='username') , @findBy(name='user')})
Private Webelements userNAmeEdt;
```

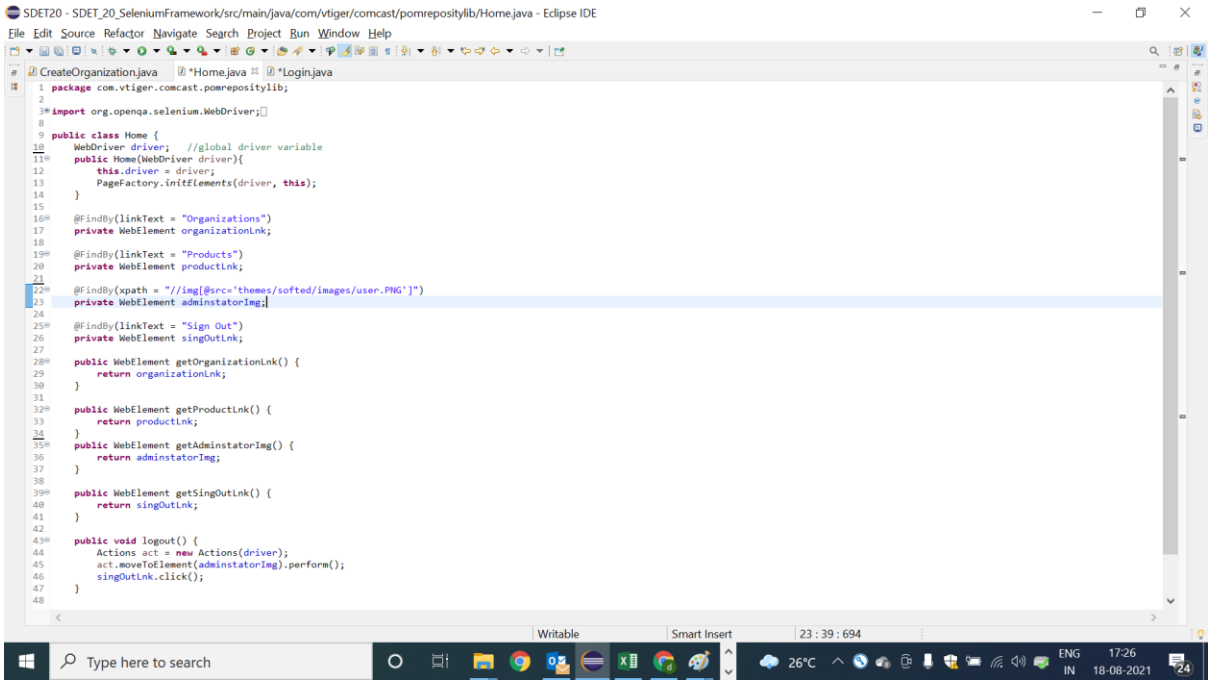
Project Structure



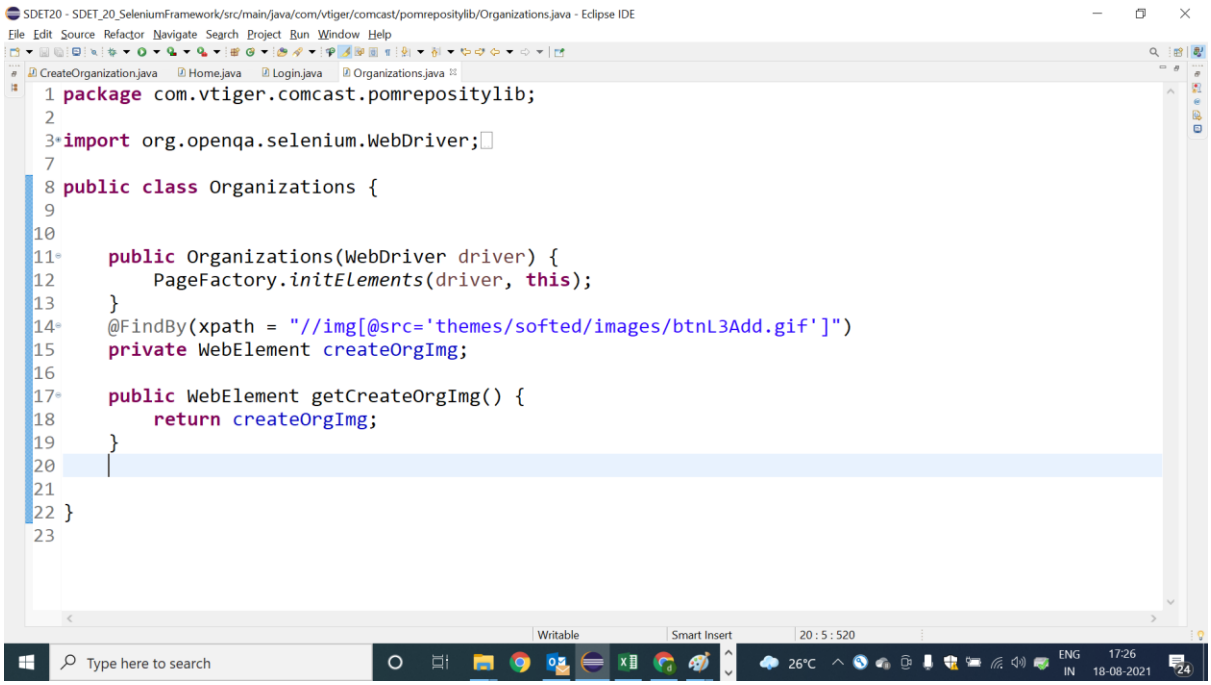
Pom Classes : Login



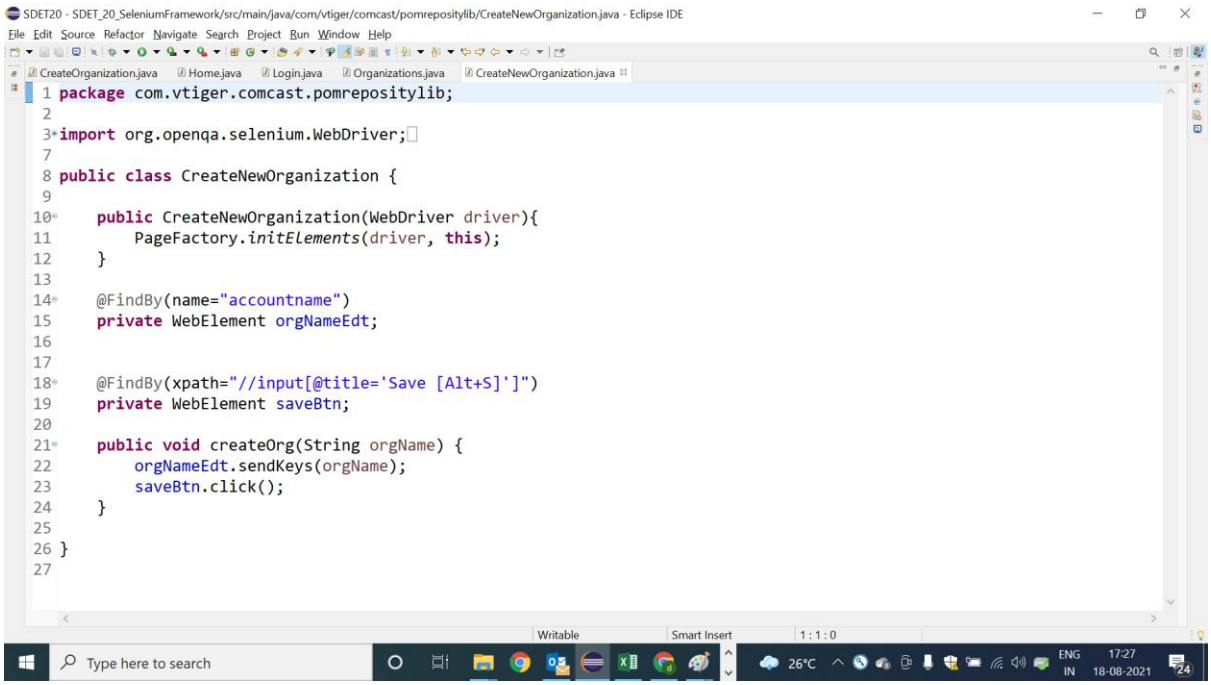
Pom Classes : Home



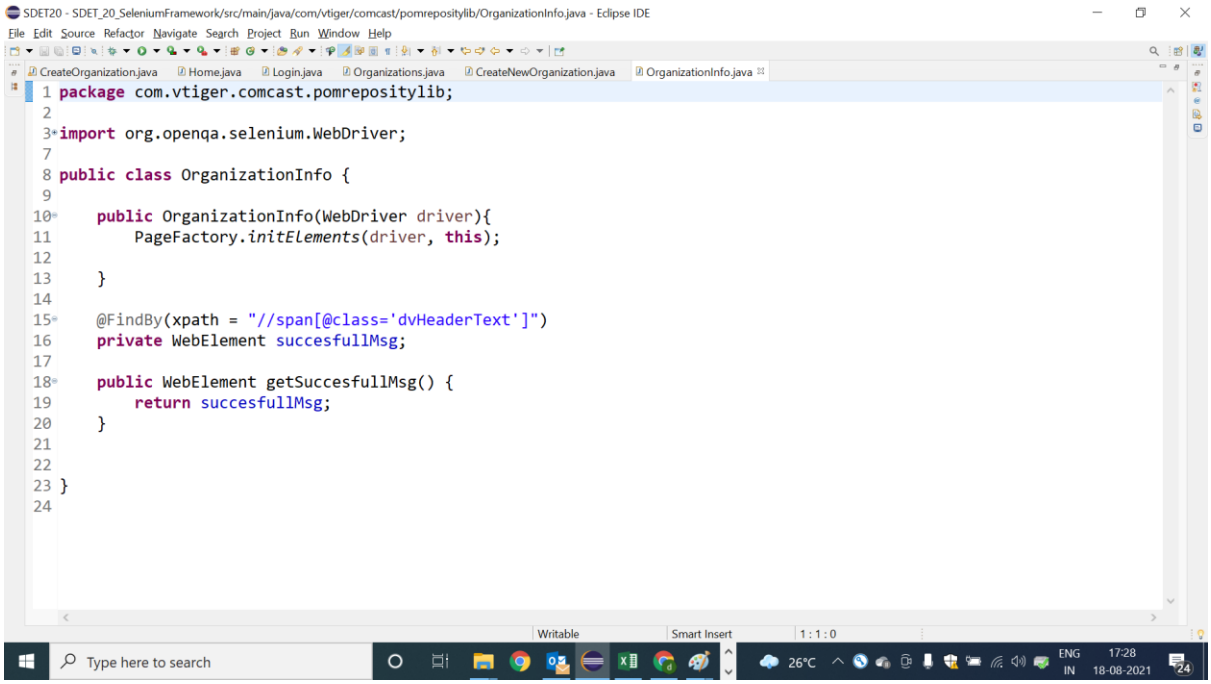
POM class : Organization



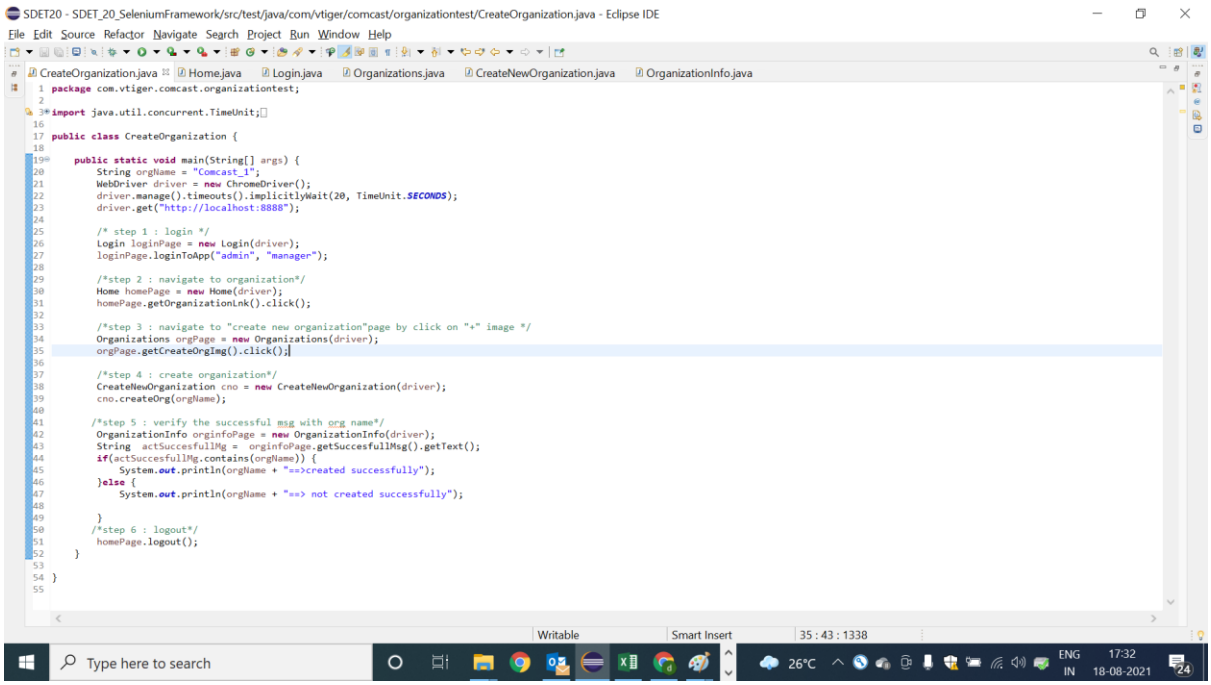
POM class : Create new Organization Page

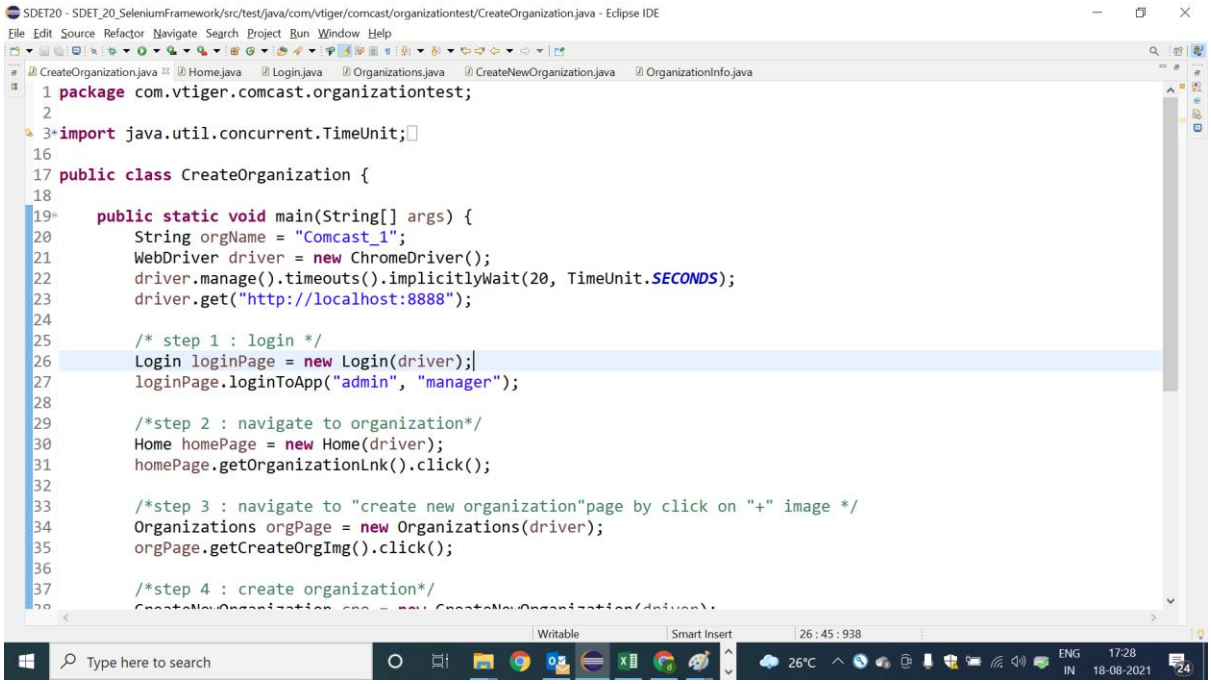


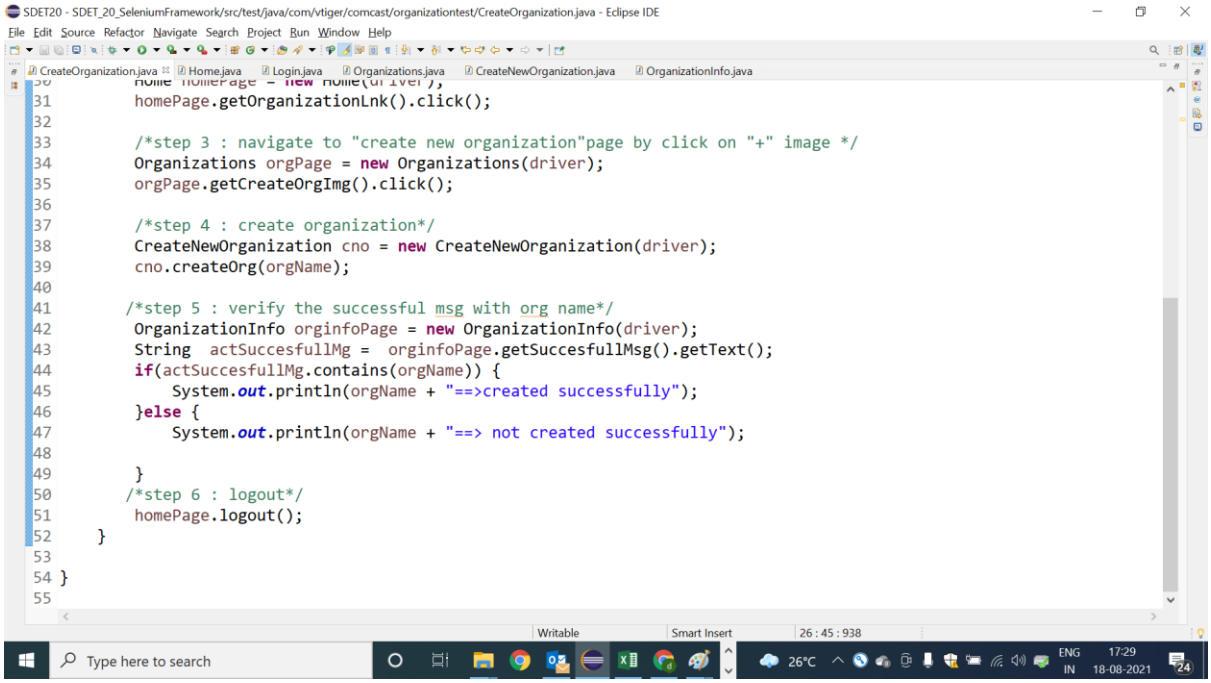
POm class : OrganizationInfo



Test Scripts using POM class




SDET20 - SDET_20_SeleniumFramework/src/test/java/com/vtiger/comcast/organizationtest/CreateOrganization.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
CreateOrganization.java Home.java Login.java Organizations.java CreateNewOrganization.java OrganizationInfo.java
1 package com.vtiger.comcast.organizationtest;
2
3 import java.util.concurrent.TimeUnit;
16
17 public class CreateOrganization {
18
19 public static void main(String[] args) {
20 String orgName = "Comcast_1";
21 WebDriver driver = new ChromeDriver();
22 driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
23 driver.get("http://localhost:8888");
24
25 /* step 1 : login */
26 Login loginPage = new Login(driver);
27 loginPage.loginToApp("admin", "manager");
28
29 /*step 2 : navigate to organization*/
30 Home homePage = new Home(driver);
31 homePage.getOrganizationLnk().click();
32
33 /*step 3 : navigate to "create new organization"page by click on "+" image */
34 Organizations orgPage = new Organizations(driver);
35 orgPage.getCreateOrgImg().click();
36
37 /*step 4 : create organization*/
Writable Smart Insert 26 : 45 : 938
Type here to search
26°C
ENG IN 17-28 18-08-2021


SDET20 - SDET_20_SeleniumFramework/src/test/java/com/vtiger/comcast/organizationtest/CreateOrganization.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
CreateOrganization.java Home.java Login.java Organizations.java CreateNewOrganization.java OrganizationInfo.java
31 homePage.getOrganizationLnk().click();
32
33 /*step 3 : navigate to "create new organization"page by click on "+" image */
34 Organizations orgPage = new Organizations(driver);
35 orgPage.getCreateOrgImg().click();
36
37 /*step 4 : create organization*/
38 CreateNewOrganization cno = new CreateNewOrganization(driver);
39 cno.createOrg(orgName);
40
41 /*step 5 : verify the successful msg with org name*/
42 OrganizationInfo orginfoPage = new OrganizationInfo(driver);
43 String actSuccesfullMg = orginfoPage.getSuccesfullMsg().getText();
44 if(actSuccesfullMg.contains(orgName)) {
45 System.out.println(orgName + "==>created successfully");
46 }else {
47 System.out.println(orgName + "==> not created successfully");
48
49 }
50 /*step 6 : logout*/
51 homePage.logout();
52 }
53
54 }
55
Writable Smart Insert 26 : 45 : 938
Type here to search
26°C
ENG IN 17-29 18-08-2021

Unit Testing Framework Tool

Available Unit testing framework tools in Current Market :

TestNG ----------> Java, .Net [Eclipse / JDevelopers / Intellj]

Junit -------> Java [Eclipse / JDevelopers / Intellj]

Nunit ---------> .Net [Visual Studio]

Pydev --------> Python [PyChram]

Rspsc ----------> Rubby [Eclipse]

Jasmin ➔ javascript [WebStrom]

All unit testing framework tool is implemented as plugin for eclipse IDE, but Junit is a default plugin for eclipse IDE.

What is testNG ?

➔TestNG is a unit test **TDD**[test Driven Development] framework , which support java & .Net

➔TestNG is an open source unit test framework tool, where NG stands for **Next Generation**.

➔TestNg developed as addition plugin for Eclipse

➔TestNG is inspired from JUNIT & NUNIT, it means it as all the feature of Nunit & Junit & also contains addition features that makes TestNg become more powerFull

Installation steps of TestNG:

Go to Eclipse window

Click on help option-> Eclipse Marketplace

- ➔ Write TestNG in find edit box and click on go button.

➔ Find TestNG for eclipse division and click on **install** button

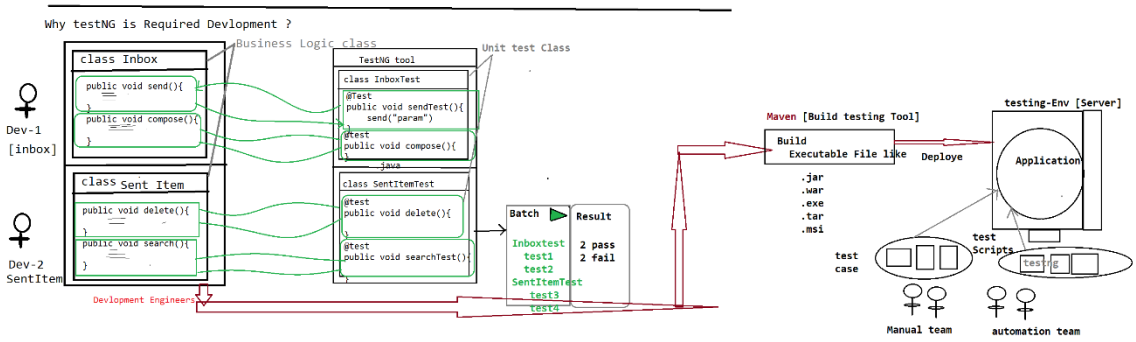
➔ Click on confirm button and I accept the terms and conditions and click on finish

⇨ **In order to verify the** TestNG installation- → Go to windows

→ Show view → others →expand java folder, TestNG symbol will be present

TestNG usage in development:

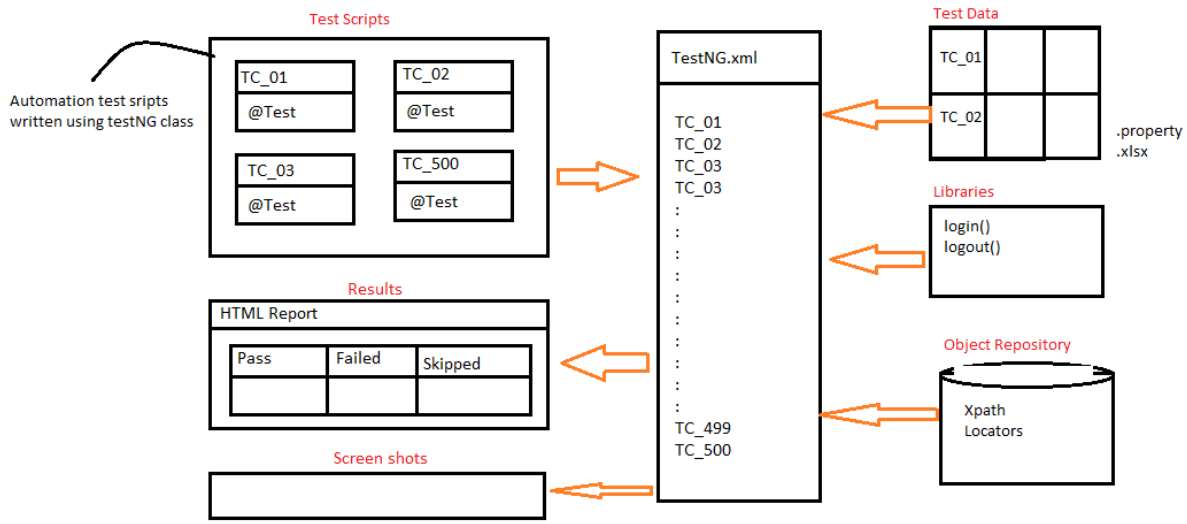
TestNG is used in development to write white box/ unit test cases and each unit test case will be used to test the source code of the application



➔In case of development, testNG will used to develop unit test cases and each unit test case check the business logic of the source code.

➔Used to achieve WBT

TestNG usage in selenium automation



➔In case of automation, testNG will be used to develop all the scripts using testNG

annotations and achieve batch execution without any manual interaction.

➔TestNG will be used to handle all framework component & help us to run all the test scripts in batch / parallel / group without any manual intervention

➔TestNG.xml is main controller of the selenium framework , where we start the execution

Why TestNG ? Why not Junit ?

- ➔ **Annotation**
- ➔ **Batch Execution**
- ➔ **Assesrtions**

New functionalities are:

->Html report

->Parallel execution

->Grouping execution

->Additional annotations

->batch execution is easier

-> iTest Listeners [used to take ScrrenShot]

->Retry Analyser [used to rerun the failed test script]

Annotations:

There are 4 blocks in java

1. **classs Block**
2. **Interface Block**
3. **Enum block**
4. **Annoation block**

```
class A{

}
```

```
interface I{

}
```

```
Enum Key{

}
```

```
Annoation Test{

}
```

Its Java block , which is used to provide metadata(information/instruction) to the JVM , at the time of execution in RUN-Time.

⇨ Annotation always start with @symbol

➔ @Test

➔ @BeforeMethod

➔ @AfterMethod

➔ @BeforeClass

➔ @AfterClass

-==========addition annotation available only in TestNG

➔ @BeforeTest

➔ @AfterTest

➔ @beforeSuite

➔ @afterSuite

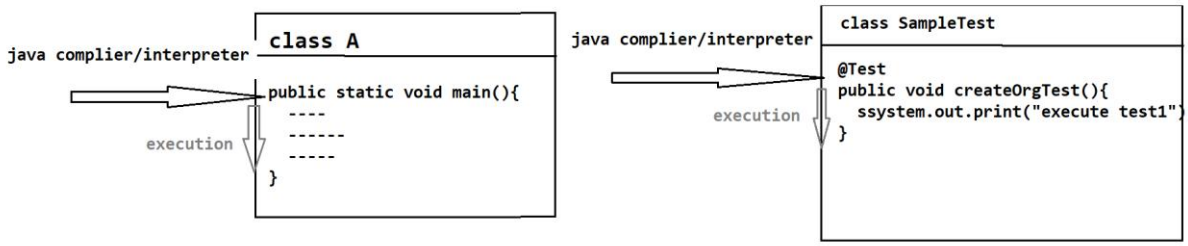
➔ @paramaters

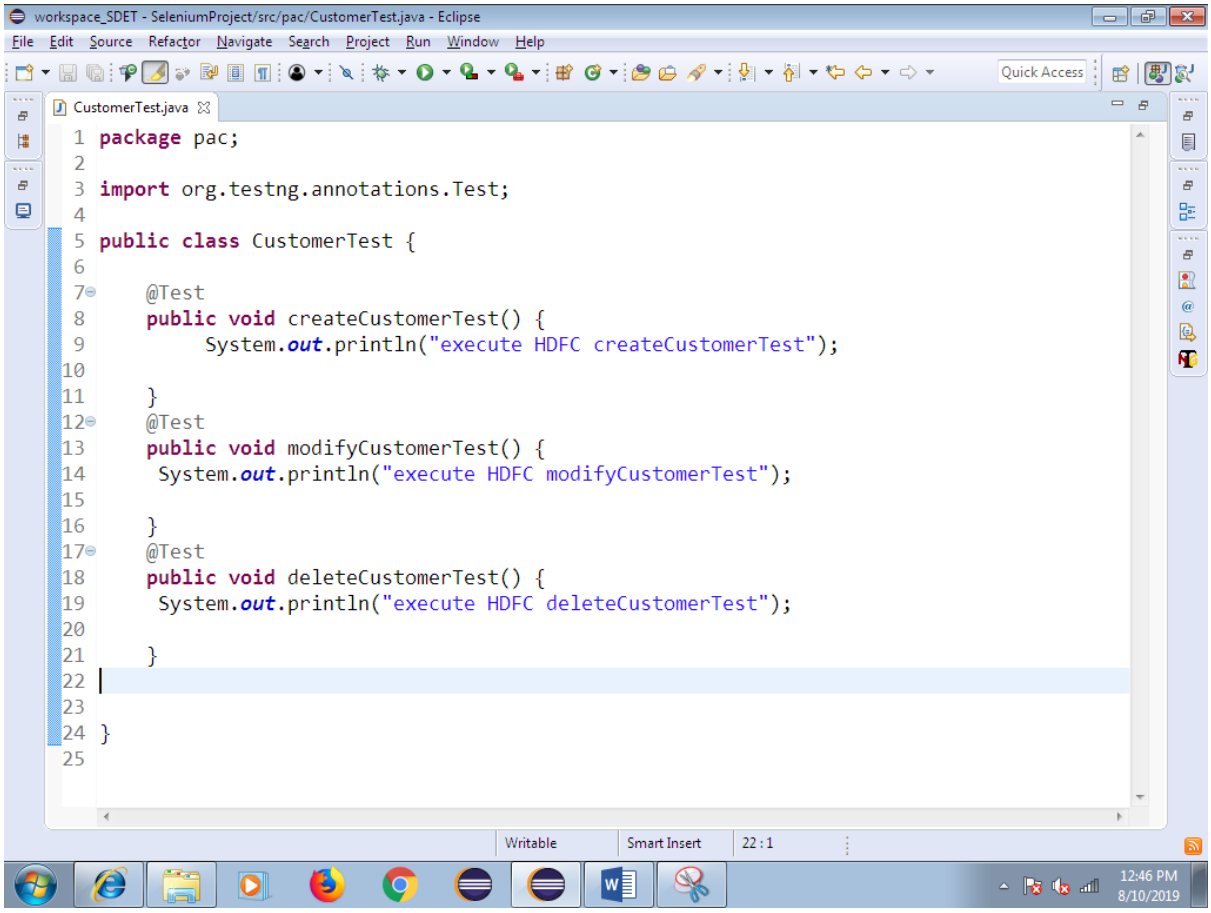
➔ @dataProvider

➔ @Listner

==========

@Test





1. Whenever we execute testng class, javaCompiler/Interpeter always looks for @Test Annotation method to start the execution.
2. **Without @Test , testNG class will not be executed, @test annotation method act like main method in testNG**
3. In one testng class we can have multiple @test methods, but each test method should have @Test annotation before method signature.
4. Annotation method return type should be “void” and access specifier should be public., but method name can be anything (we have provide manual testNameTest).
5. As per the Rule of the Automation , TESTNG class Name should be ModuleNAme , @test method name should be manual testCase Name
6. As per the Rule TestNG class Name & testNG method Name should end With “Test”
7. One Manual test case contains multiple steps all those steps should be automated using one @test annotation & test name should be manual test-case Name & end with Test
8. In one class we can keep multiple @test annotation , but in real time we are going maintain maximum 10 to 15 @test script , because maintenance will be easy

How to Verify html report

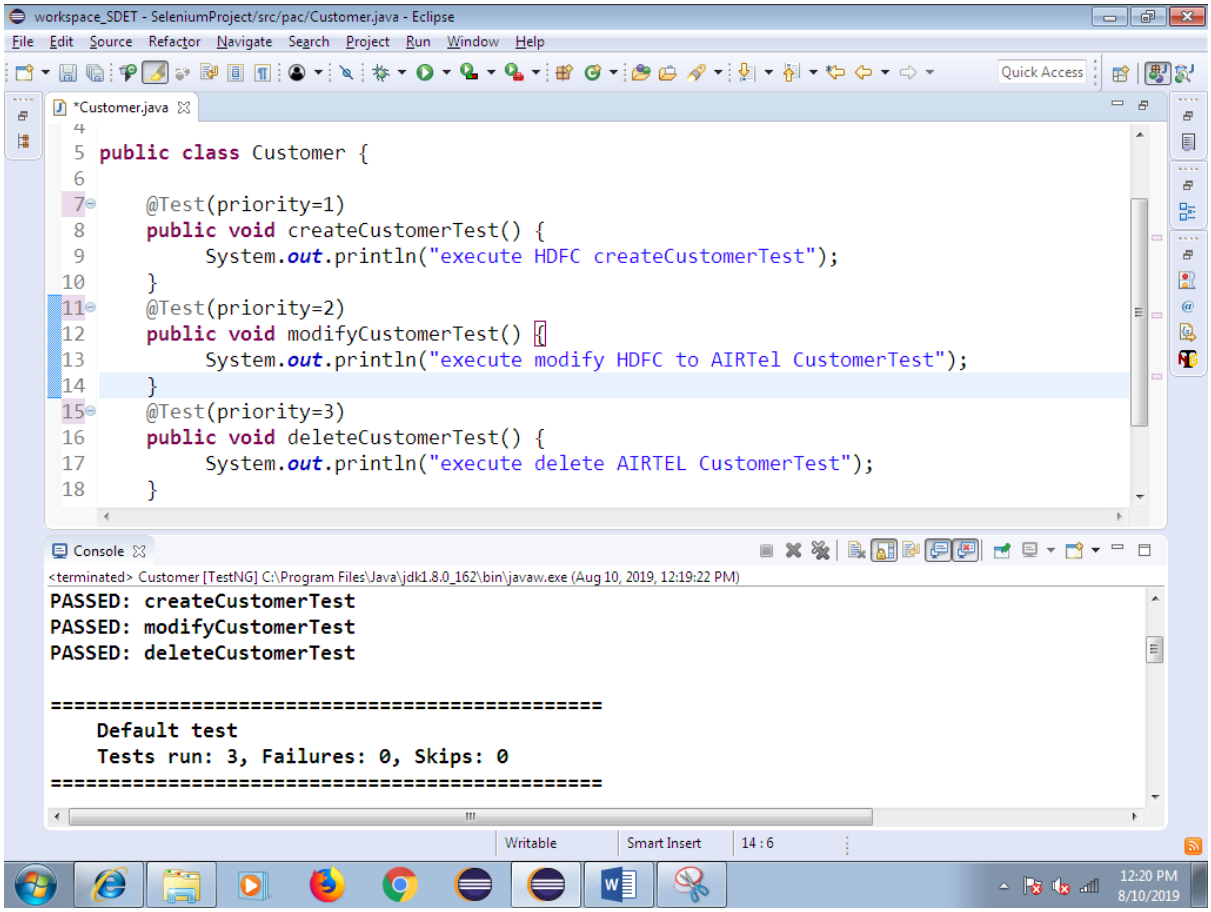
Refresh the project after execution—select project right click and click on refresh

Automatically we get **test-Output** folder within the same project.

Expand test-output folder → select emailable.html and right click → open with→ Open with browser

Priority

Whenever we execute testNG class , bydefault all the test method will be executed based on Alphabetical Order , in order the change the Order of Execution , we go for priority

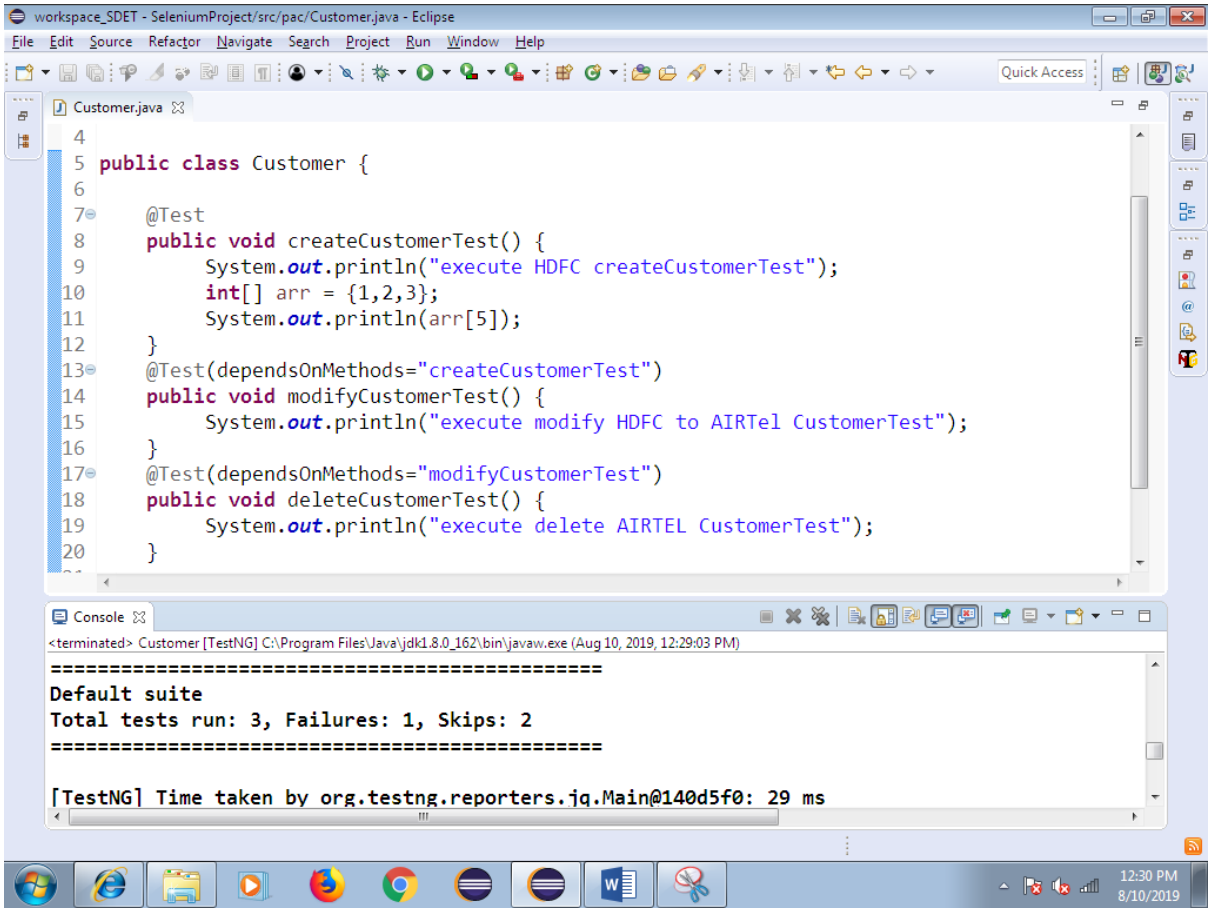


DependsOnMethod :

Its help us to check the dependent test case is pass or fail,

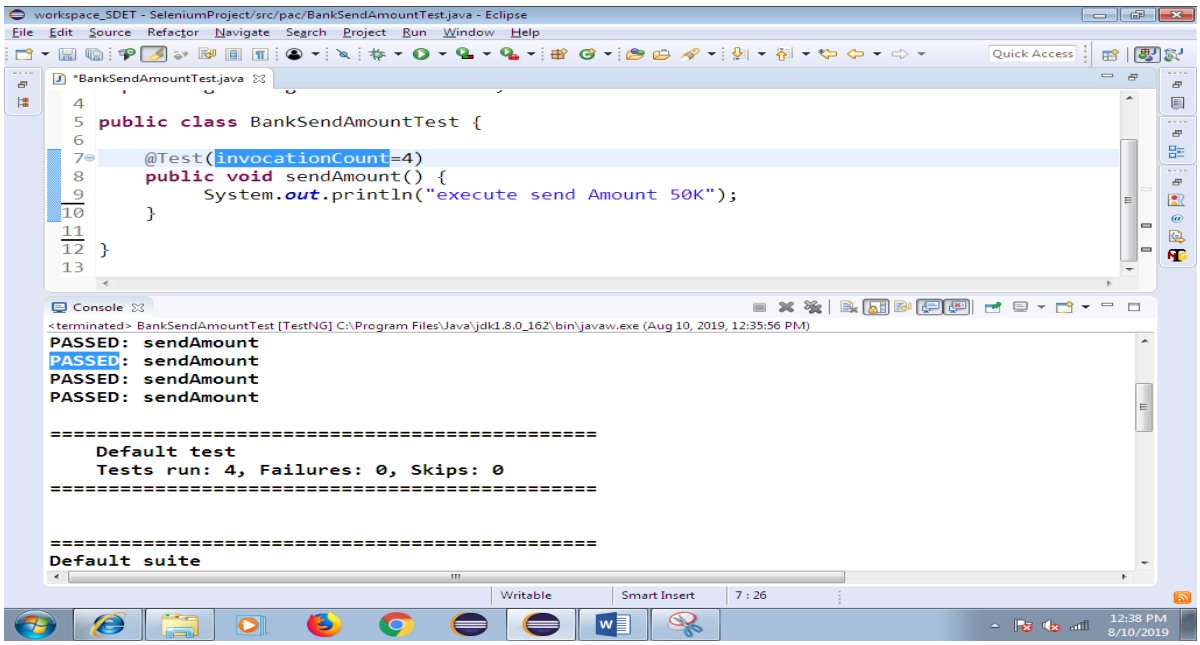
If dependent test-script get pass, execution will continue

If dependent test-script get fail, skip the all other test script which is dependent on first test



10. Invocation Count:

Same test-script executed with multiple Times with same test data



11 . Data Provider

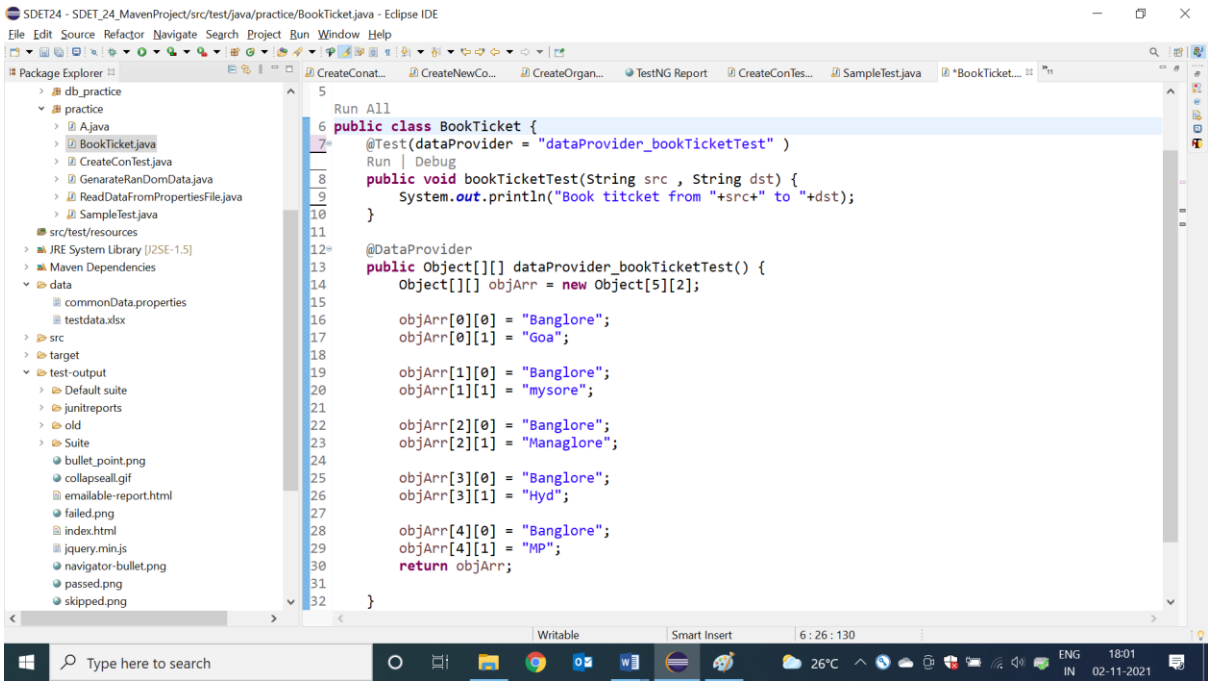
a. In Order to execute same test case multiple Times with different test Data , we go for @DataProvider annotataion
b. Data Provider annotation always return TWO –DIMENSTINAL Object array , because we can pass any type of datatype
c. Data Provider annotation help us to execute same test multiple times with the different set of data , each test-script should have dedicated @dataprovider annotation
d. DataProvider annotation play major role in Data driven framework , where we need to test the application with huge amount of data like Ecommerce , Booking ,banking application
e. In Below example , row count is 5 ➔ it indicates test needs to be executed 5 times

Column count is 2➔ it indicates every iteration 2 arguments will be passed

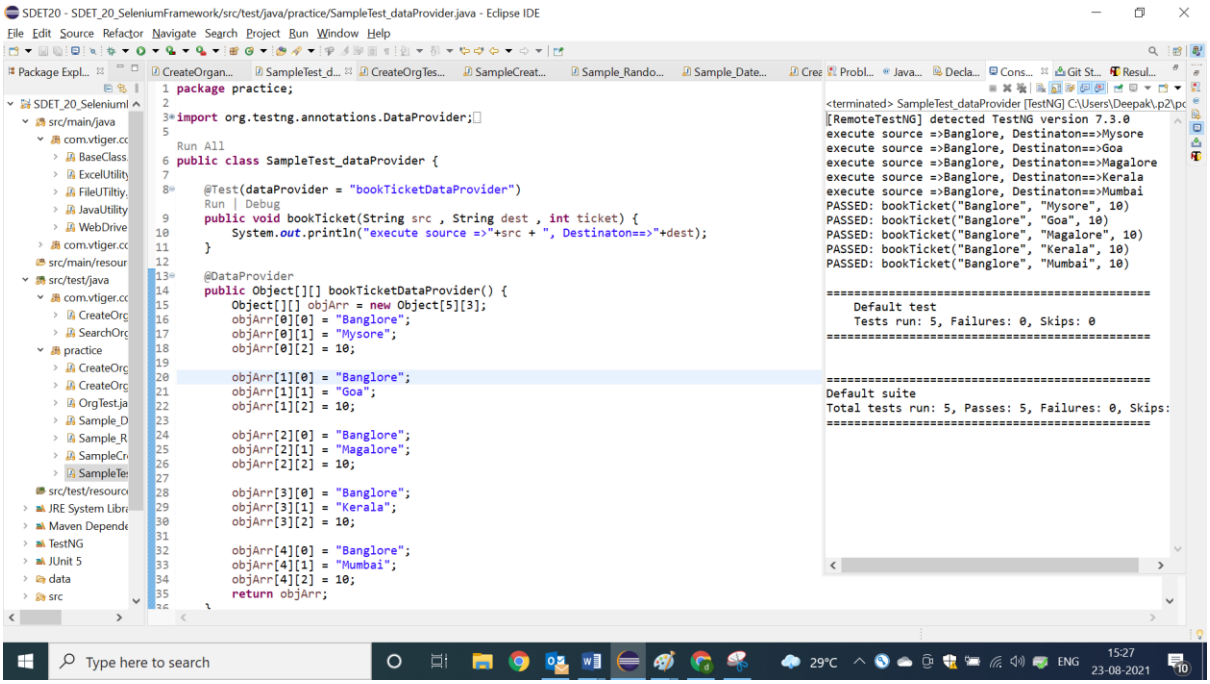
object[5][2]

Banglaore 0,0	Mysore 0,1
Banglore 1,0	Goa 1,1
Banglore 2,0	Mangalore 2,1
3,0 Banglore	3,1 Kerala
Banglore 4,0	Mumbai 4, 1

➔Sample program for data provider

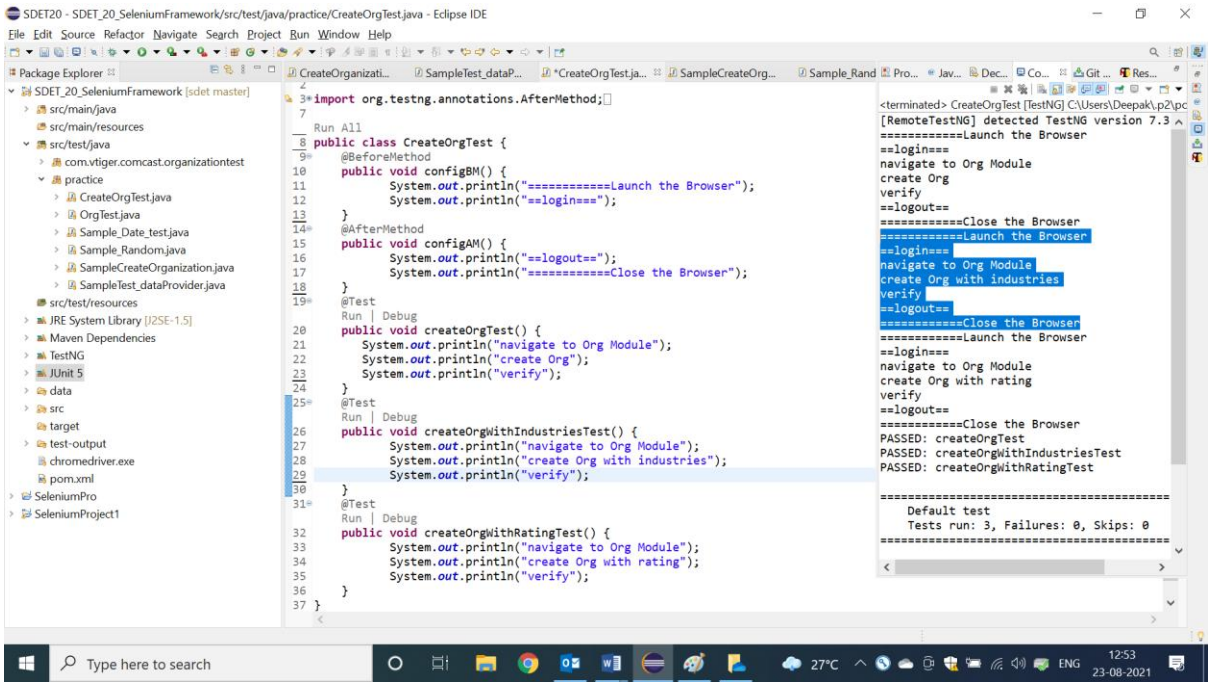


➔Data provider example with 3 arguments

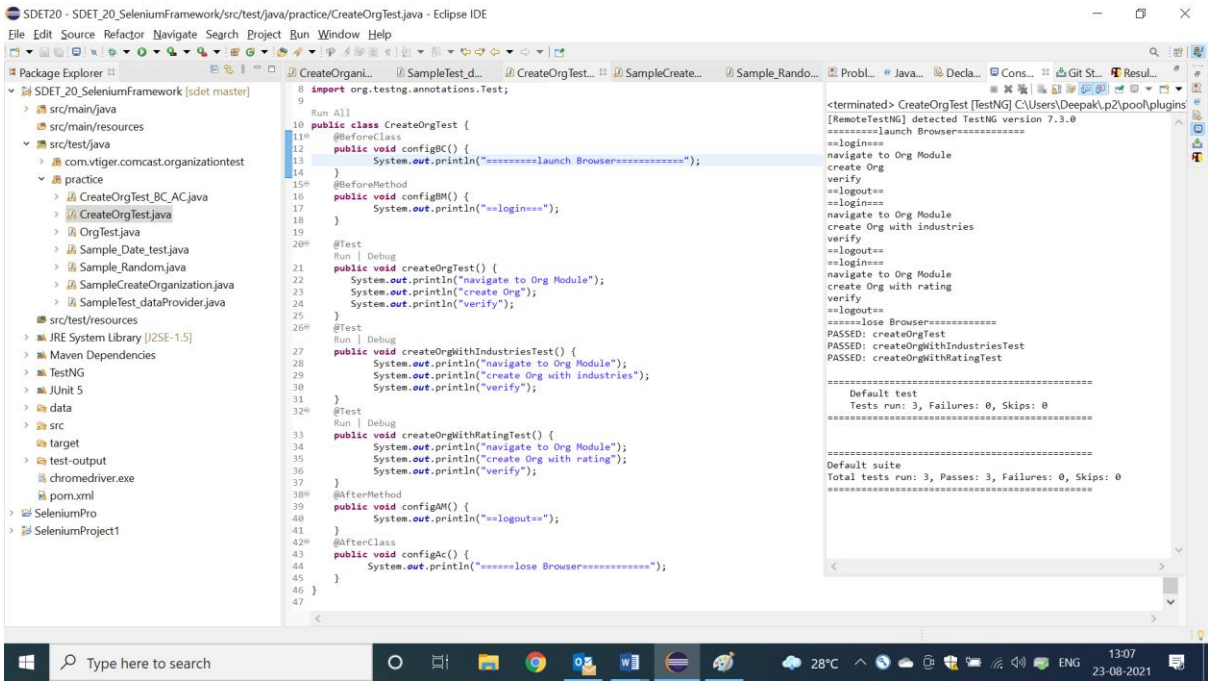


@BeforeMethod @AfterMethod

1. Before method annotations will be executed, before executing each @test method in a class
2. After method annotation will be executed, after executing each @test in a class
3. Beforemethod & aftermethod will not be executed, without @test annotation method, because they are configaration method
4. In order to implement similar pre- condition for all the testcase like “LOGIN code ” we go for @BM
5. In order to implement similar post-condition for all the testcase like “LOGOUT code ” we go for @AM



@BeforeClass & @Afterclass



1. BeforeClass Annotation method will be executed, before Executing first @test in a class
2. AfterClass Annotation method will be executed, after executing all/last test-case with in a class
3. BeforeClass & AfterClass annotations will be executed only once in a entire class execution.
4. It will be used to develop global configuration like launch browser, object initialization
   NOTE: As per the Rule of the Automation , test case should not be have dependency between one to another test , every testcase should be unique (it means every test should have fresh login & logout code)

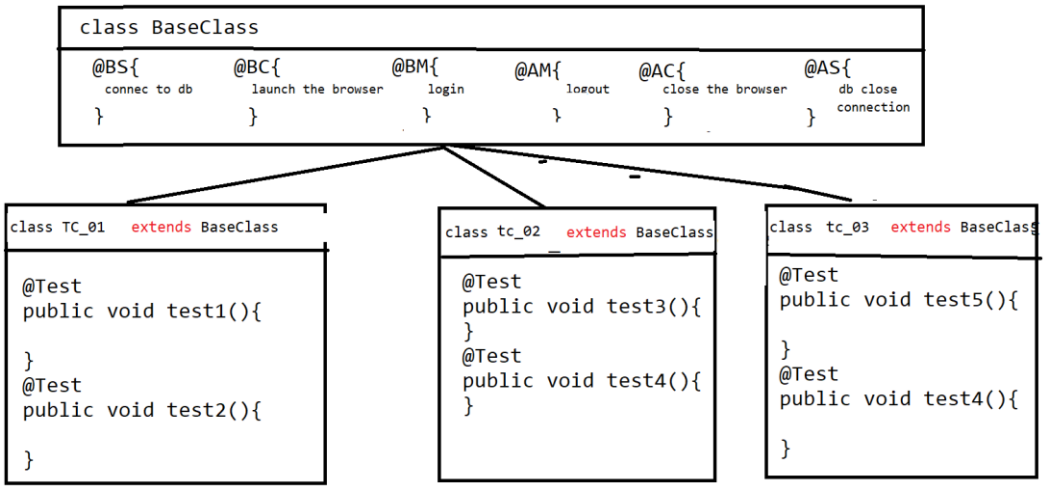
ConFig Annotation methods Usage in REAL Selenium Framework

1. In Real selenium FrameWork , all the configure annotation will be implemented Inside the BaseClass, that is being shared all the Automation engineers via GITGUB
2. BaseClass should be available in generic libraries package.
3. As per the Rule , Every testScripts class should extend BaseClass, so that all the configure annotation will be inherited & executed to the test scripts automatically

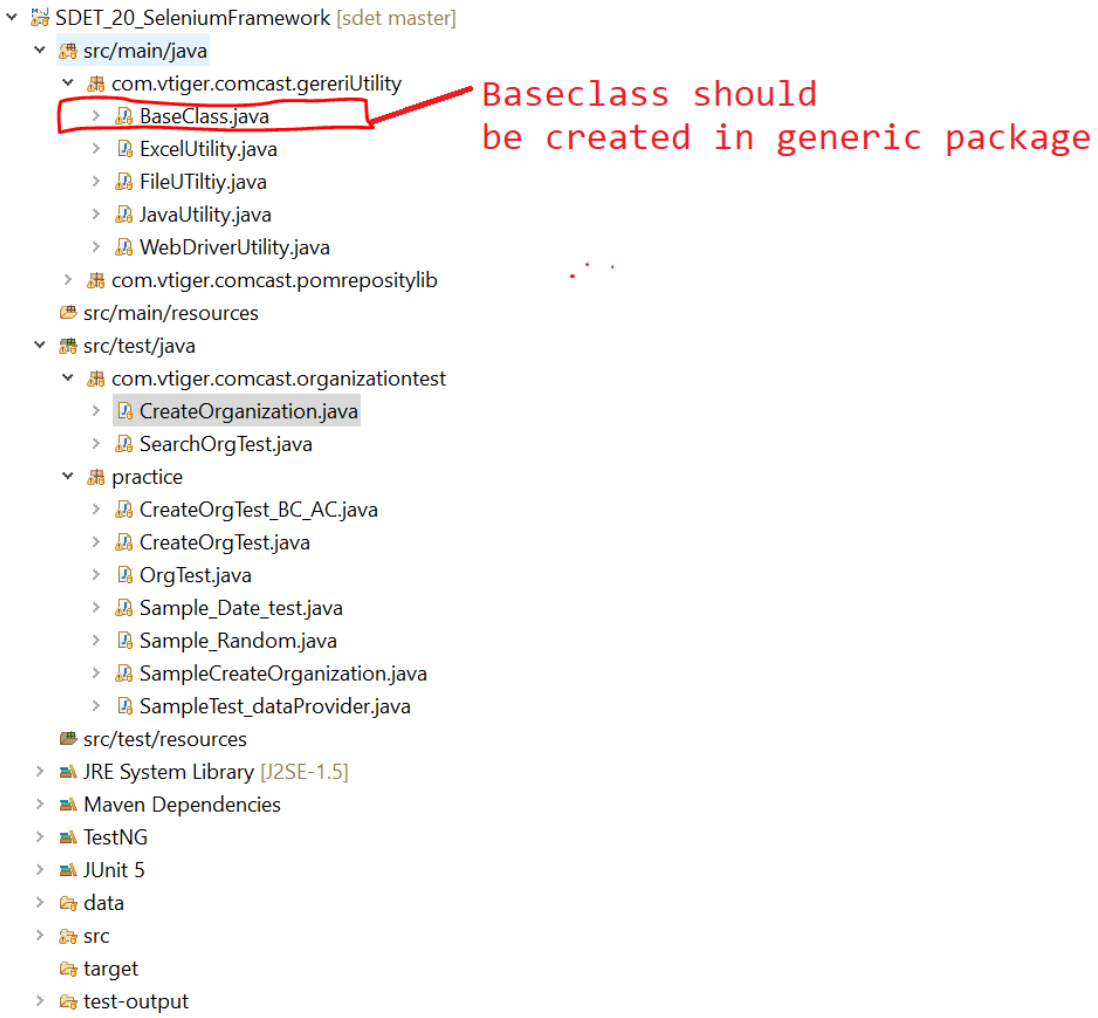
Advantages :

1. Code Reusability
2. Code Optimization
3. Modification is easy
4. Maintenance is easy
5. Test Development is faster
6. Slightly execution time is reduced

EG :



Project Structure in Eclipse



=========Base class Program==========

```
package com.vtiger.comcast.gereriUtility;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.AfterSuite;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.BeforeSuite;

import com.vtiger.comcast.pomrepositylib.Home;
import com.vtiger.comcast.pomrepositylib.Login;

public class BaseClass {
        public WebDriver driver;
          /*Object Creation for Lib*/
                public JavaUtility jLib = new JavaUtility();
                public WebDriverUtility wLib = new WebDriverUtility();
                public FileUTiltiy fLib = new FileUTiltiy();
                public ExcelUtility eLib = new ExcelUtility();

        @BeforeSuite
        public void configBS() {
```

```
                System.out.println("=======================connect to DB=======================");

        }

        @BeforeClass
        public void configBC() {
                System.out.println("=============Launch the Browser=======");
                driver = new ChromeDriver();
                wLib.waitUntilPageLoad(driver);
                driver.manage().window().maximize();
        }

        @BeforeMethod
        public void configBM() throws Throwable {
                /*common Data*/
                String USERNAME = fLib.getPropertyKeyValue("username");
                String PASSWORD = fLib.getPropertyKeyValue("password");
                String URL = fLib.getPropertyKeyValue("url");
                String BROWSER = fLib.getPropertyKeyValue("browser");
                /* Navigate to app*/
                   driver.get(URL);
                /* step 1 : login */
                Login loginPage = new Login(driver);
                loginPage.loginToApp(USERNAME, PASSWORD);
        }


        @AfterMethod
        public void configAM() {
              /*step 6 : logout*/
                Home homePage = new Home(driver);
        homePage.logout();
        }

        @AfterClass
        public void configAC() {
                System.out.println("=============Close the Browser=======");
                driver.quit();
        }

        @AfterSuite
        public void configAS() {
                System.out.println("=======================close DB=======================");
        }
}
```

==========Sample Test Script code using base class==========

```
package com.vtiger.comcast.organizationtest;

import org.testng.annotations.Test;

import com.vtiger.comcast.gereriUtility.BaseClass;
import com.vtiger.comcast.pomrepositylib.CreateNewOrganization;
import com.vtiger.comcast.pomrepositylib.Home;
import com.vtiger.comcast.pomrepositylib.OrganizationInfo;
import com.vtiger.comcast.pomrepositylib.Organizations;

public class CreateOrganization  extends BaseClass{

        @Test
        public void createOrgTest() throws Throwable {

                int randomInt = jLib.getRanDomNumber();
                /*test script Data*/
                String orgName = eLib.getDataFromExcel("Sheet1", 1, 2) + randomInt;

        /*step 2 : navigate to organization*/
        Home homePage = new Home(driver);
        homePage.getOrganizationLnk().click();
```

```java
/*step 3 : navigate to "create new organization"page by click on "+" image */
Organizations orgPage = new Organizations(driver);
orgPage.getCreateOrgImg().click();

/*step 4 : create organization*/
CreateNewOrganization cno = new CreateNewOrganization(driver);
cno.createOrg(orgName);

/*step 5 : verify the successful msg with org name*/
OrganizationInfo orginfoPage = new OrganizationInfo(driver);
String  actSuccesfullMg =  orginfoPage.getSuccesfullMsg().getText();
if(actSuccesfullMg.contains(orgName)) {
System.out.println(orgName + "==>created successfully");
}else {
System.out.println(orgName + "==> not created successfully");

}
}


@Test
public void createOrgWithIndutriesTest() throws Throwable {
        /*test script Data*/
        int randomInt = jLib.getRanDomNumber();
        String orgName = eLib.getDataFromExcel("Sheet1", 4, 2) + randomInt;
        String industriesType = eLib.getDataFromExcel("Sheet1", 4, 3);
         /*step 2 : navigate to organization*/
Home homePage = new Home(driver);
homePage.getOrganizationLnk().click();

/*step 3 : navigate to "create new organization"page by click on "+" image */
Organizations orgPage = new Organizations(driver);
orgPage.getCreateOrgImg().click();

/*step 4 : create organization*/
CreateNewOrganization cno = new CreateNewOrganization(driver);
cno.createOrg(orgName, industriesType);

/*verify orgname & industry */
OrganizationInfo orginfoPage = new OrganizationInfo(driver);
String  actSuccesfullMg =  orginfoPage.getSuccesfullMsg().getText();
if(actSuccesfullMg.contains(orgName)) {
System.out.println(orgName + "==>created successfully");
}else {
System.out.println(orgName + "==> not created successfully");

}

String actIndustryType = orginfoPage.getIndutryTypeInfo().getText();
if(actIndustryType.equals(industriesType)) {
System.out.println(industriesType + "==>industry is verified successfully");
}else {
System.out.println(industriesType + "==>industry is not verified successfully");

}
}

@Test
public void createOrgWithRatingTest() throws Throwable {
        /*test script Data*/
        int randomInt = jLib.getRanDomNumber();
        String orgName = eLib.getDataFromExcel("Sheet1", 7, 2) + randomInt;
        String rating = eLib.getDataFromExcel("Sheet1", 7, 3);
           /*step 2 : navigate to organization*/
Home homePage = new Home(driver);
homePage.getOrganizationLnk().click();

/*step 3 : navigate to "create new organization"page by click on "+" image */
Organizations orgPage = new Organizations(driver);
orgPage.getCreateOrgImg().click();

/*step 4 : create organization*/
CreateNewOrganization cno = new CreateNewOrganization(driver);
```

```
        cno.createOrg(orgName, rating, true);

        /*verify orgname & industry */
        OrganizationInfo orginfoPage = new OrganizationInfo(driver);
        String  actSuccesfullMg =  orginfoPage.getSuccesfullMsg().getText();
        if(actSuccesfullMg.contains(orgName)) {
        System.out.println(orgName + "==>created successfully");
        }else {
        System.out.println(orgName + "==> not created successfully");

        }
        String actRatingType = orginfoPage.getRatingTypeInfo().getText();

        if(actRatingType.equals(rating)) {
        System.out.println(rating + "==>industry is verified successfully");
        }else {
        System.out.println(rating + "==>industry is not verified successfully");

        }
        }
}
```

Batch Execution

- Collection of multiple test script is called batch, execute multiple test script through xml in a single click is called batch execution.
- In order to achieve batch execution, we go for Testng.xml configuration file
- TestNG xml file always start with suite xml tag fallowed by <test> and <classes.>
- In one xml file we can invoke N-number of testng classes, but all the classes should be present within a project.
- All the class should fallowed by packageNAme

  EG :

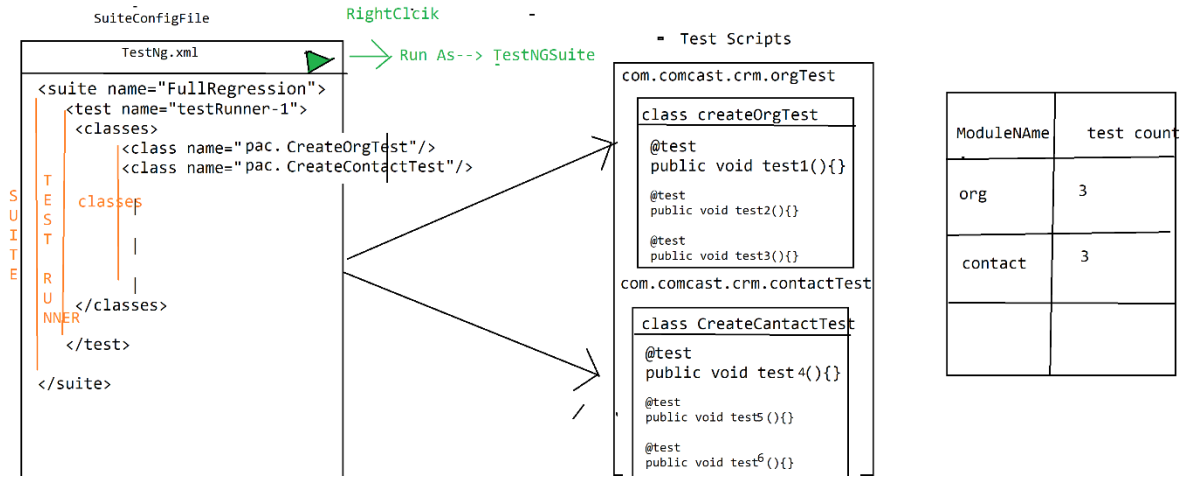
  <class name="com.comcast.orgtest.CreateOrg"></class>

  How to create testNG xml file automatically through eclipse?

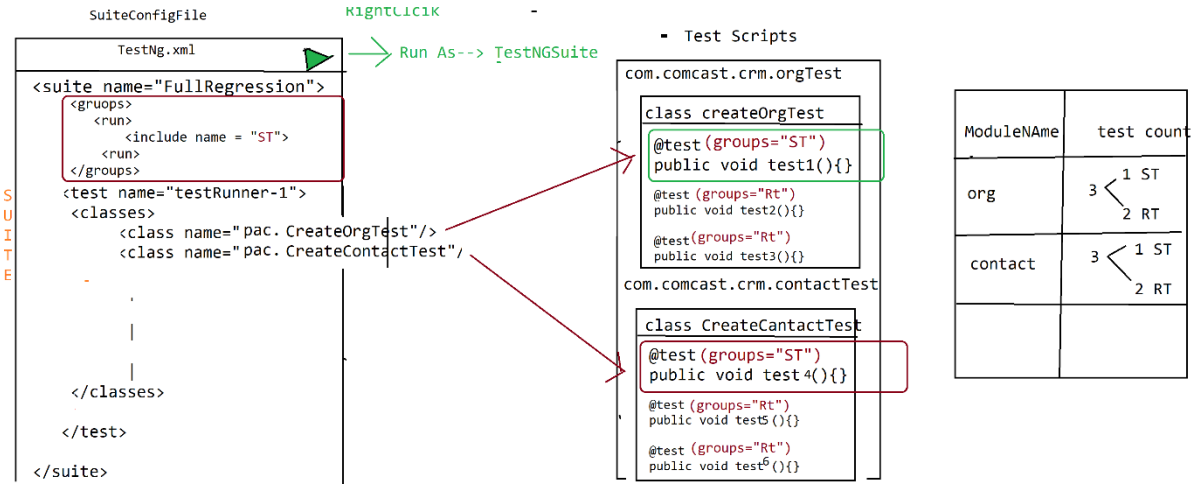
  Select all the testNG classes or packages -> right click-> select➔ testNG ➔and click on "Convert to testing" and➔ click on finish.

  - ⇨ Automatically you will get the testng.xml file with in the project
  - ⇨ In order to edit xml File➔ double click on testing.xml➔ click on "Source"

Batch Execution : [Full regression testing]



Grouping Execution:



- ➔ Collection of similar test scripts across the testing classes is called grouping Execution
- ➔ In order achieve grouping execution , each & every test script should have group name, group name will be written along with annotation
- ➔ In grouping execution, all configure annotation should have group name , other wise those annotation will not participate in grouping execution like @BeforeSuite @BeforeClass , @BeforeMethode etc
  EG :

```
@BeforeSuite(groups = {"smokeTest","regressionTest"})
public void configBS() {
        System.out.println("======Execute BeforeSuite=============");
}
```

Smoke Test:

```
@Test(groups={"smokeTest"})
public void createCustomerTest()
{
        System.out.println("execute createCustomerTest");
```

```
        }

    @Test(groups={"regressionTest")
            public void modifyCustomerTest()
      {
                    System.out.println("execute modifyCustomerTest");
      }
```

➔ In order to invoke grouping execution should declare Group Key in testing.xml file & group keep should be declared before <test>, after <suite> tag

Smoke Test:

```
<suite name="Suite">
        <groups>
                <run>
                   <include name="smokeTest"/>
                </run>
        </groups>
 <test name="Test">
  <classes>
   <class name="pac1.ProjectAndCustomerTest"/>
    <class name="pac2.ReportTest"/>

  </classes>
 </test>
        </suite>
```

➔We can invoke multiple groupkey in one XML File

```
<suite name="Suite">
        <groups>
                <run>
                   <include name="smokeTest"/>
                   <include name="regressionTest"/>
                </run>
        </groups>
 <test name="Test">
  <classes>
   <class name="pac1.ProjectAndCustomerTest"/>
    <class name="pac2.ReportTest"/>

  </classes>
 </test>
```

➔One test can have multiple Group name

```
@Test(groups={"regressionTest","smokeTest"})
    public void modifyCustomerTest()
{
            System.out.println("execute modifyCustomerTest");
}
```

Regional Regression Test:

➔To execute particular test cases across the Suite is called regional regression testing

➔in Real time , impact area is given by developer / test lead , based on that idea XML will be created

➔ Whenever we want execute particular @test method inside class , we go for <method> & <include>

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="com.comcast.crm.contacttest.CreateContactTest">
            <methods>
               <include name="createdContactTest"/>
            </methods>
      </class>
      <class name="com.comcast.crm.orgtest.CreateOrgTest">
            <methods>
               <include name="createdOrgWithRatingTest"/>
            </methods>
      </class>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Parallel Execution:

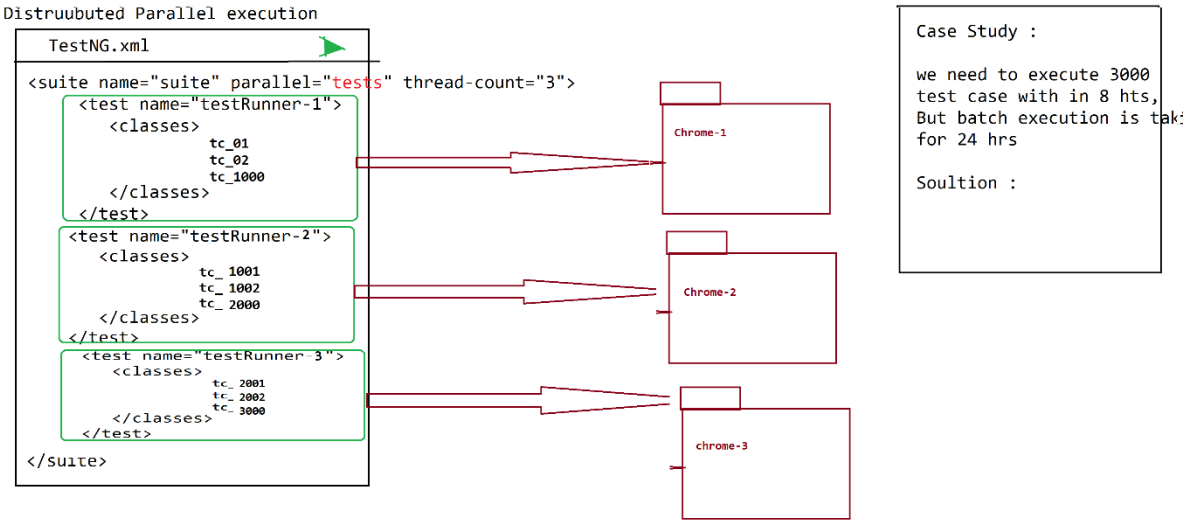
There are 3 types of Parallel Execution

1. **Distributed Parallel execution:**
2. **Cross browser parallel execution**
3. **Cross platform parallel execution**

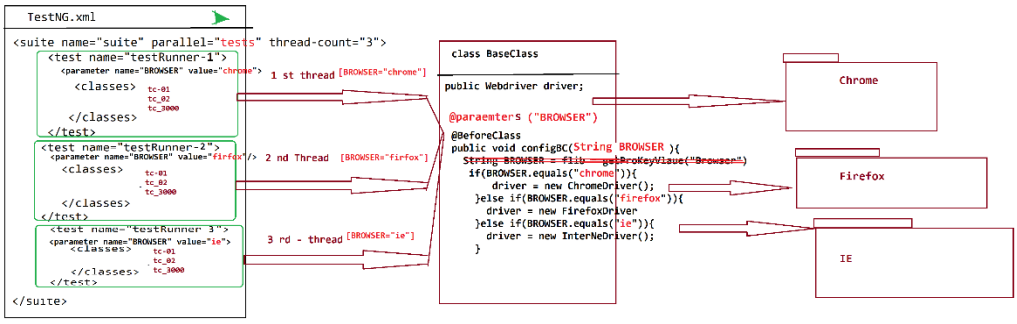
1. **Distributed Parallel execution:**

➔Distribute the test case across the multiple test runner & execute, each <test> Test runner in parallel is called Distributed Parallel execution

- ⇨ we reduce the suite execution time, so that we can get the result early
- ⇨ in order the archive parallel execution we should enable parallel=”tests” & thread-count=5 in <suite> , then create multiple test runner & distribute the testcase
- ⇨ maximum thread count is 5
- ⇨ Thread count should be same as number of <test> testRunner



2 Cross browser parallel execution / Browser Compatibility testing



1. Execute same set of testcase in different Browser parallel is called cross browser testing
2. To achieve cross browser testing we should we <parameter> in XML file & @parameters annotation inside the testScript

3. **<parameter > is used to specify the browser data for each <test>**

EG :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel= "tests" thread-count="2">
  <test name="Test-Runner-Chrome">
   <parameter name="BROWSER" value="chrome"/>
    <classes>
       <class name="com.vtiger.comcast.organizationtest.CreateOrganization"/>
      <class name="com.vtiger.comcast.organizationtest.SearchOrgTest"/>

    </classes>
  </test> <!-- Test -->

  <test name="Test-Runner-Firefox">
  <parameter name="BROWSER" value="firefox"/>
    <classes>
    <class name="com.vtiger.comcast.organizationtest.CreateOrganization"/>
      <class name="com.vtiger.comcast.organizationtest.SearchOrgTest"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

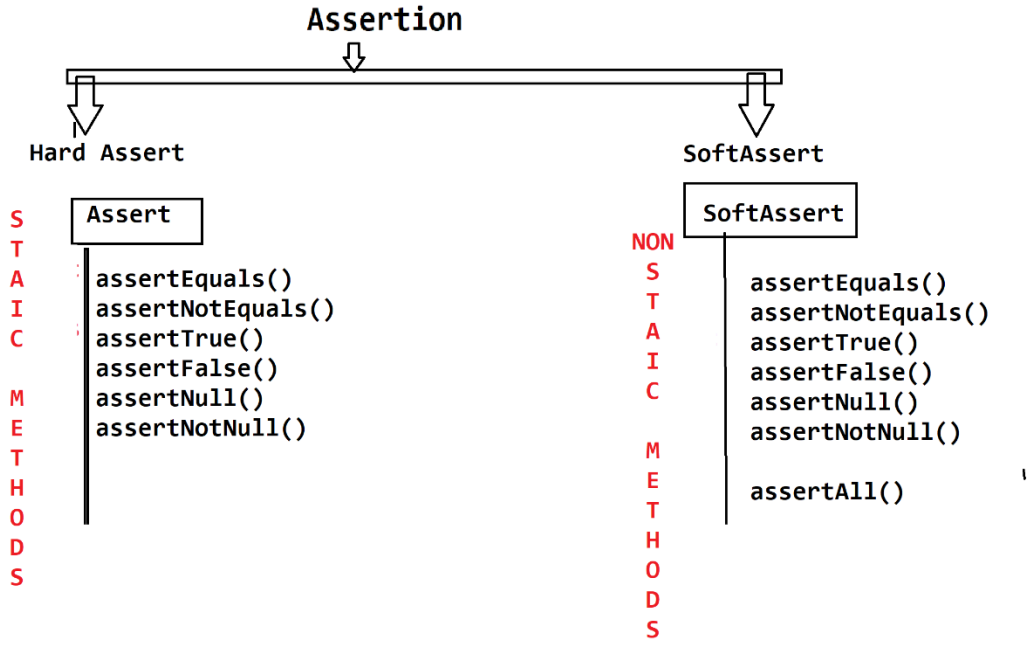
4. @parameters annotation will be used to receive the data (Eg : browser , platform etc) from the XML file to test Scripts [Like BaseClass]

Note : Replace below annotation inside the balseclass

```
@Parameters("BROWSER")
        @BeforeClass
        public void configBC(String BROWSER) {
                System.out.println("=============Launch the Browser=======");
                if(BROWSER.equals("chrome")) {
                    driver = new ChromeDriver();
                }else if(BROWSER.equals("firefox")) {
                        driver = new FirefoxDriver();
                }else if(BROWSER.equals("ie")) {
                        driver = new InternetExplorerDriver();
                }else {
                         driver = new ChromeDriver();
                }
                wLib.waitUntilPageLoad(driver);
                driver.manage().window().maximize();
        }
```

Assertion / CheckPoint

- **Assertion is a feature available in TestNG used to validate test scripts expected results**
- **As per the Rule of the automation every expected result should be verified with Assert statements, because java "if else "statement will not have capability to fail the testNG test Scripts**
- **There are 2 types of Assertions in TESTNG**



Hard Assertion	Soft Assertion
All methods are static in nature	All methods are non-static in nature
It does not allow further execution of test if the line containing hard assert gets failed.	Next steps would be executed even if the line containing soft assertion gets failed.
Whole test case gets failed if at least 1 hard assert fails.	AssertAll() extra lines of code are required to track the fail status.
To verify mandatory fields we go for hard assert	To verify non mandatory fields we go for soft assert

HardAssert

Whenever hardAssert method fails, testNG generate AssertError exception & stop the current test execution & continue execution with remaining test

```
@Test
public void createCustomerTest(){
```

```
		System.out.println("step_1");
		System.out.println("step_2");
		Assert.assertEquals("A", "B");
		System.out.println("step_3");
		System.out.println("step_4");
	}
	@Test
	public void modifyCustomerTest(){
		System.out.println("==============================");
		System.out.println("step_1");
		System.out.println("step_2");
		System.out.println("step_3");
	}
```

Out Put

```
step_1
step_2
==============================
step_1
step_2
step_3
PASSED: modifyCustomerTest
FAILED: createCustomerTest
java.lang.AssertionError: expected [B] but found [A]
```

Soft Assert

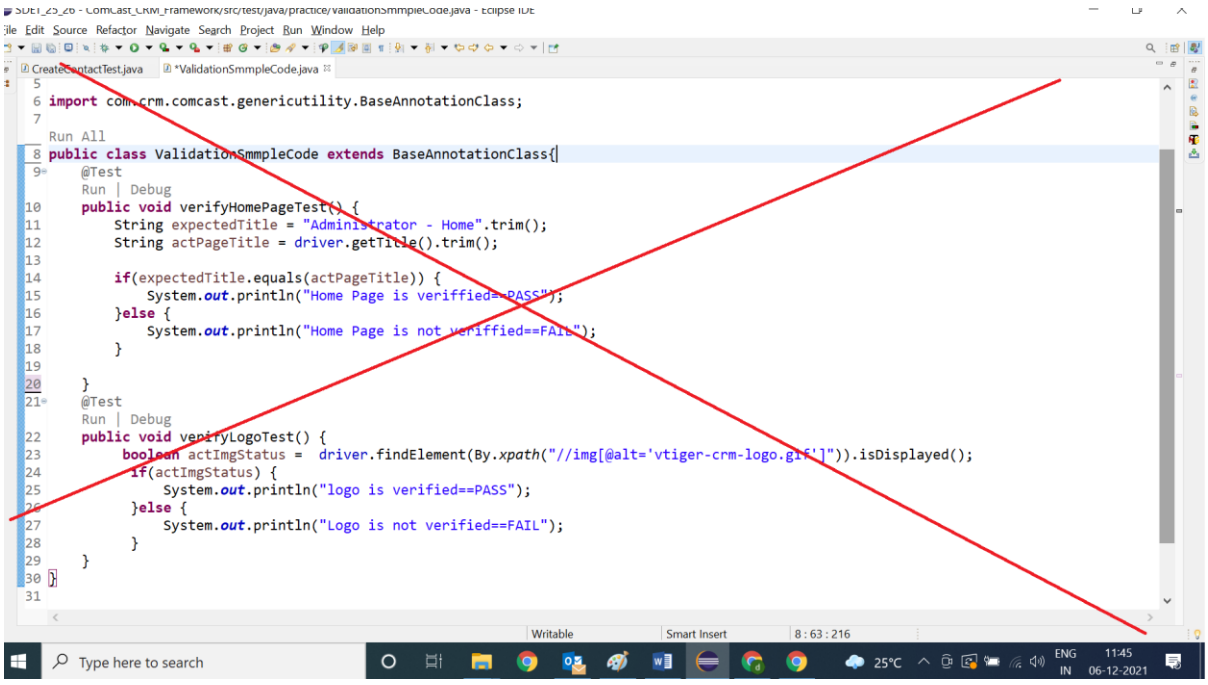
Whenever softAssert mtd fails , testNG Generate AssertError exception & continue execution with remaining steps of same testScript

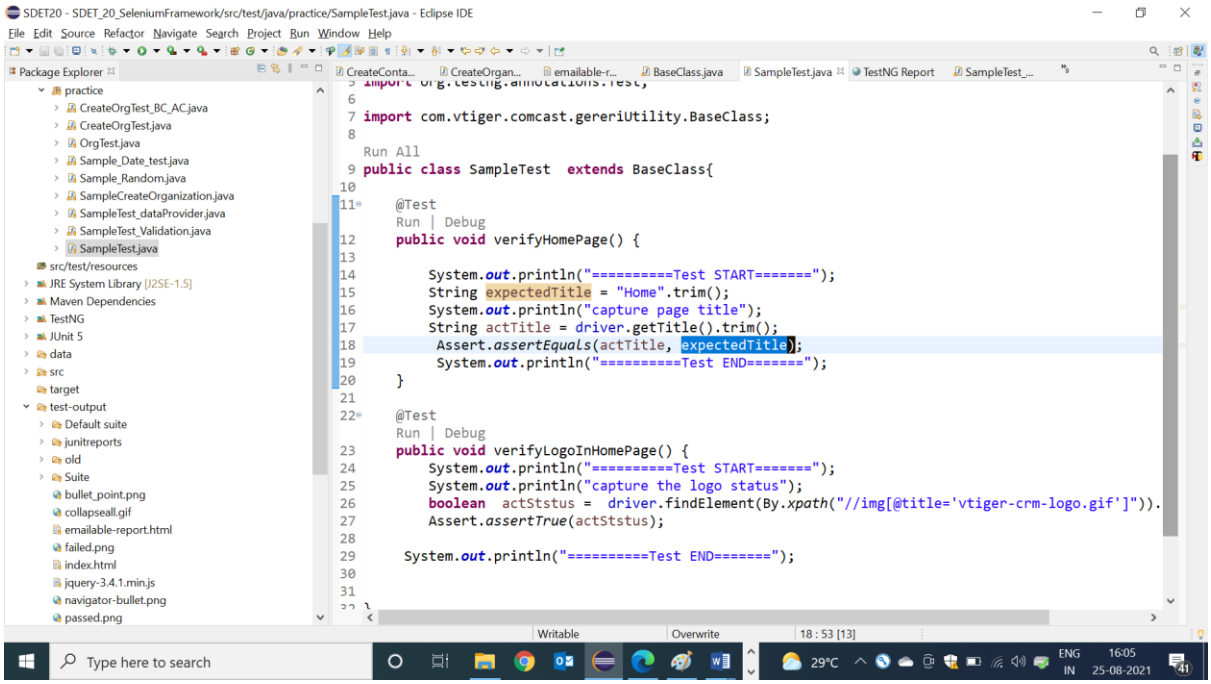
```
	public void createCustomerTest(){
		System.out.println("step_1");
		System.out.println("step_2");
		SoftAssert s = new SoftAssert();
		s.assertEquals("A", "B");
		System.out.println("step_3");
		s.assertEquals("X", "Y");
		System.out.println("step_4");
		s.assertAll();
	}
	@Test
	public void modifyCustomerTest(){
		System.out.println("==============================");
		System.out.println("step_1");
		System.out.println("step_2");
		System.out.println("step_3");
		System.out.println("step_4");
	}
```

Output

step_1
step_2
step_3
step_4
==========
step_1
step_2
step_3
step_4
PASSED: modifyCustomerTest
FAILED: createCustomerTest
java.lang.AssertionError: The following asserts failed:
expected [B] but found [A],
expected [Y] but found [X]

Assertion usage in Real Time

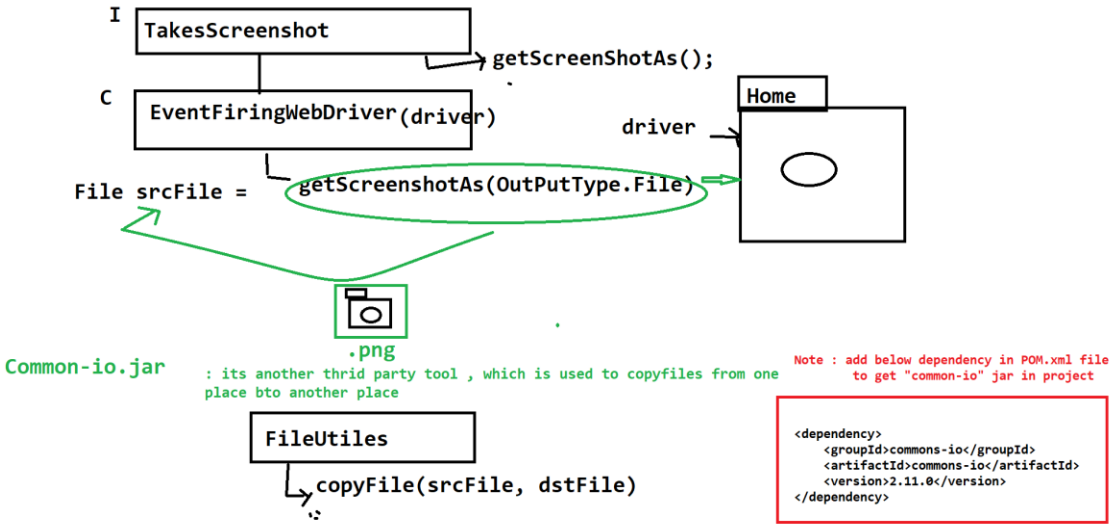




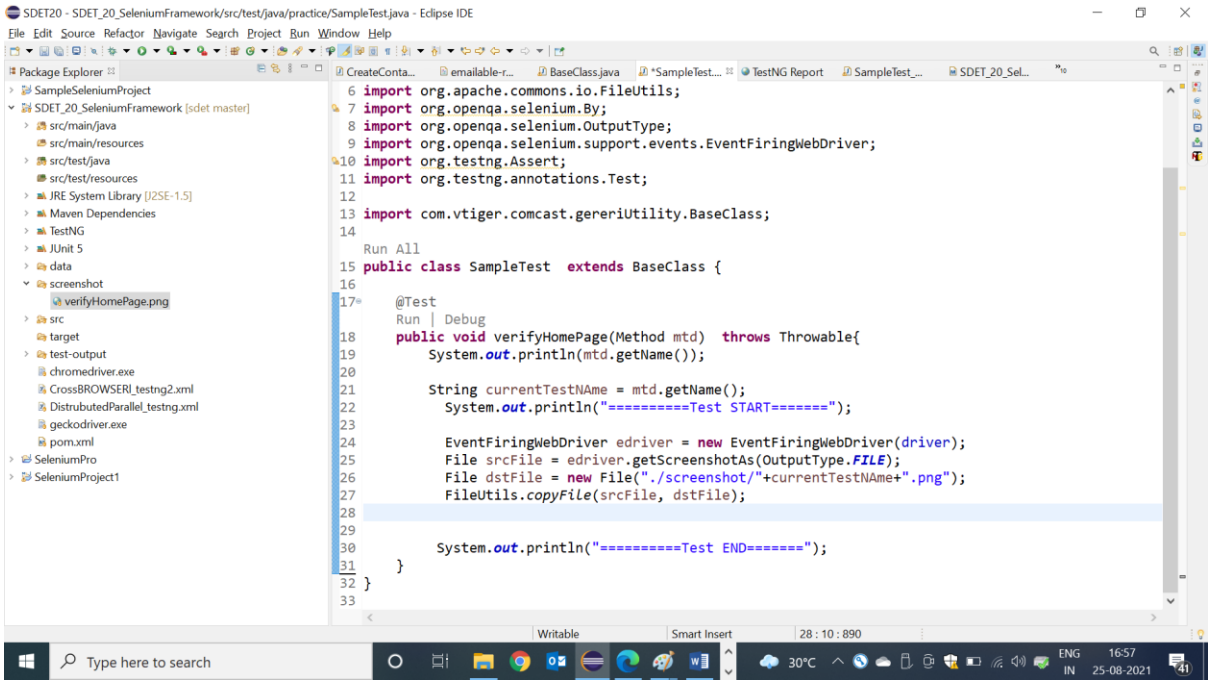
Advantages of Assertion:

- ⇨ It's used to fail the TESTNG test scripts
- ⇨ It's used for test scripts validation
- ⇨ It's generate "AssertErrorException" & reason of the failure + failed line number whenever test is failed
- ⇨ We can compare any 2 primitive variable or array or Collection or MAP in single line

Screen Shot

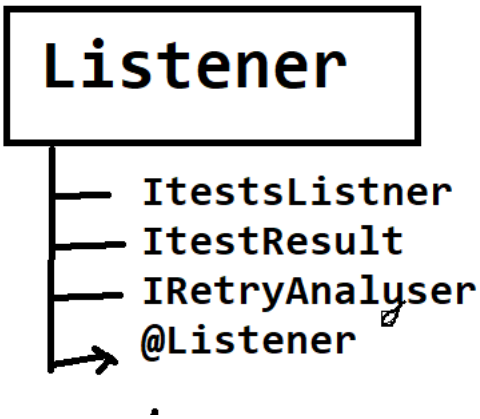


WebDriver Code to take a Screenshot



Listener

- Listen is feature available in TESTNG , which is used to capture runtime events during execution & perform appropriate action based on eventtype



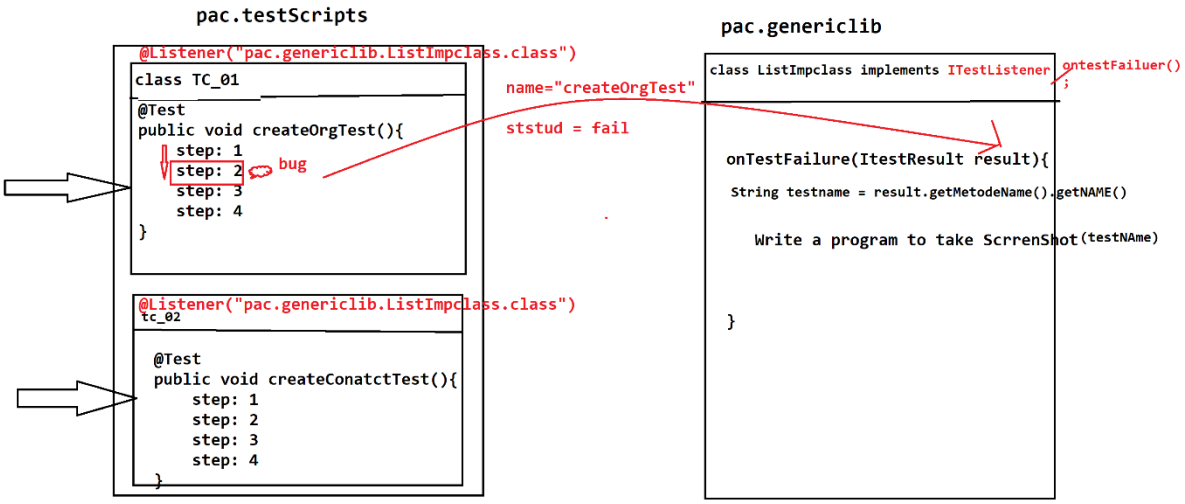
ITestListener:

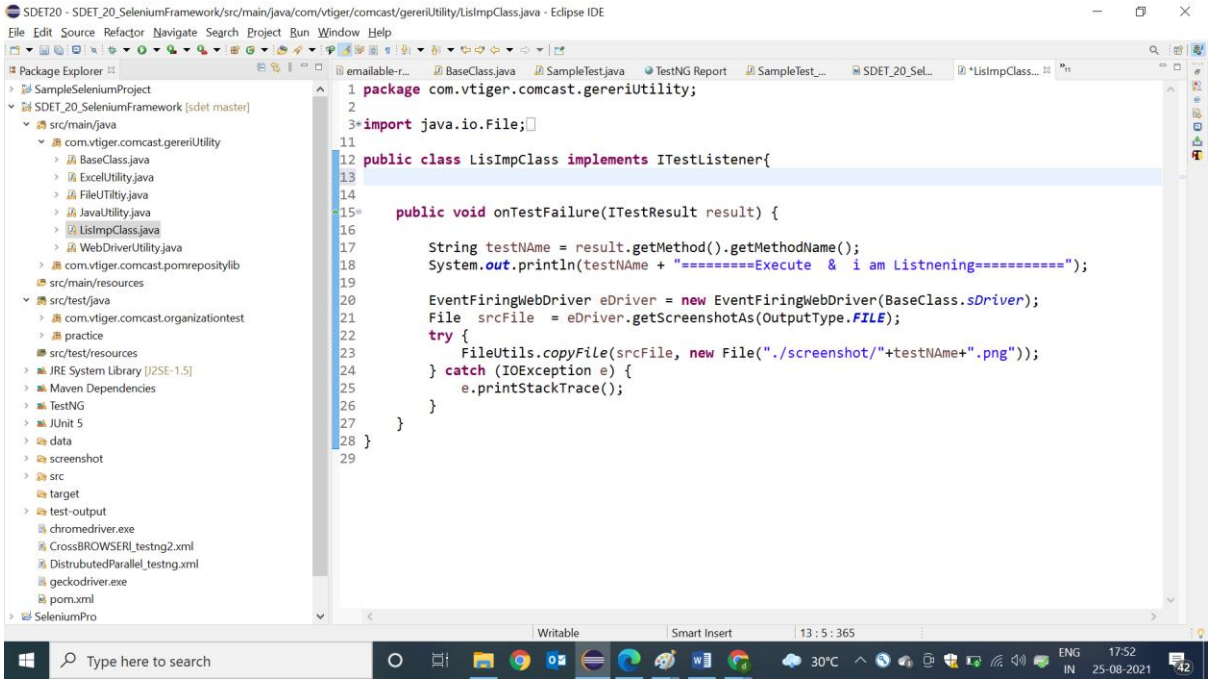
- This is a special feature in testNG which enables the user to take run time events whenever the test scripts fail/pass. Implementation class for ITestListener is mandatory to use Listener feature
- @Listener is testing annotation , which is used to monitor the test execution in the runtime & generate a runtime event to Listener Implementation class , if test is pass / fail
- @Listener annotation will be declared in every testScripts class before class definition block
- ListImplemenation class helps us to receive the failure events from the @Listener & perform appropriate actions

Advantages of Listener :

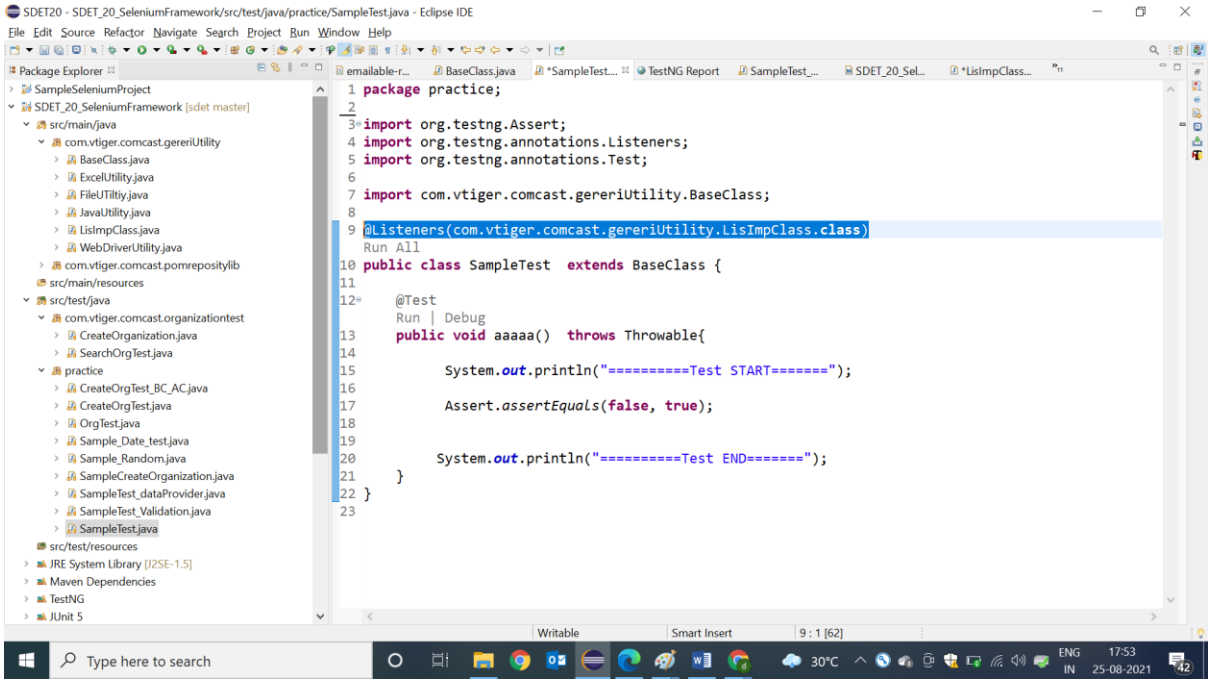
- We can use Listener to take a screenshot for the failed test case, when we execute in bulk
- We can also use Listener for connect to DB , Launch browser & login precondition program
- We can also use Listener for Extend Report Configuration

Listener Implementation class :

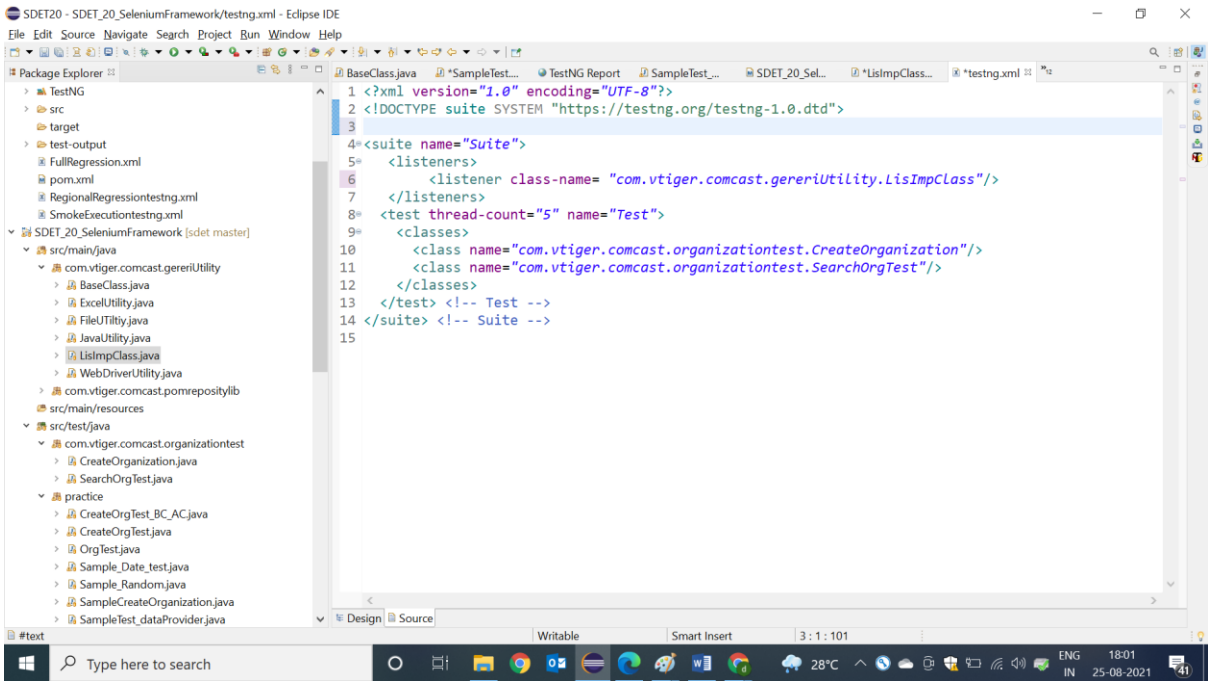




Sample test for Listener :

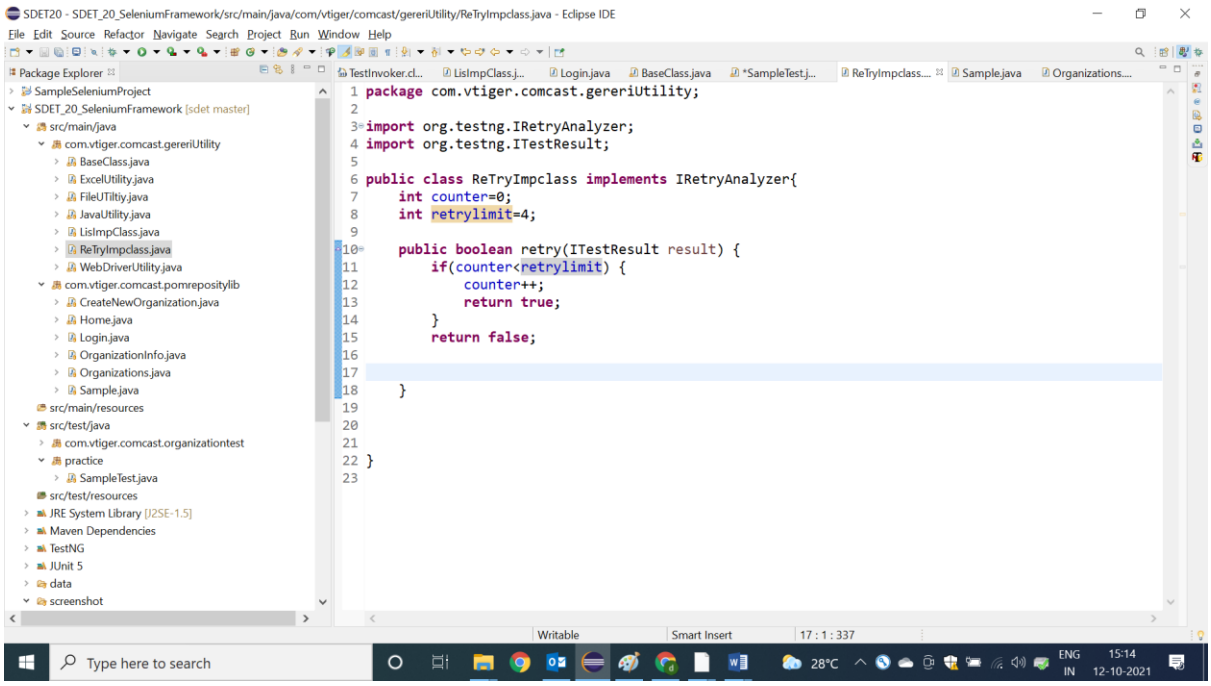


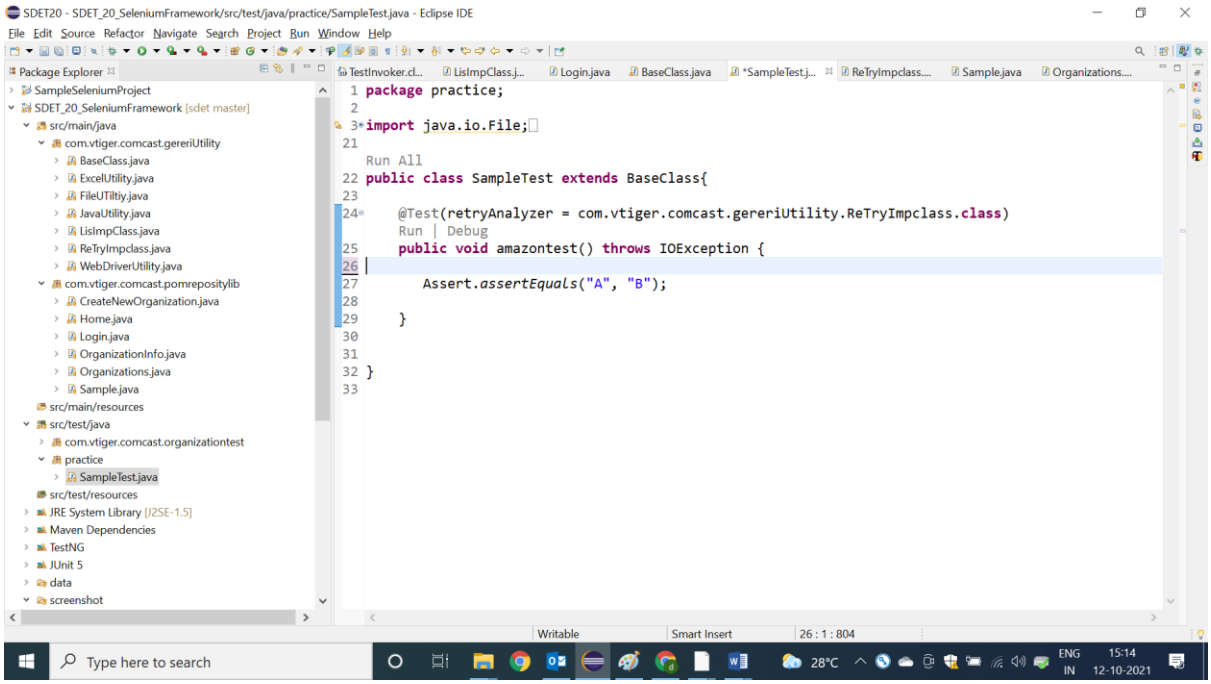
How to use Listener in TestNG.xml



RetryAnalyzer:

- ⇨ This feature of TestNG helps the user to rerun the test script when ever test is getting failed
- ⇨ In order to use this feature we have to implements RetryAnalyzer interface & override retry method
- ⇨ Inside the retry method, we should specify the upper limit so that test script will get executed specified limit when test is getting failed.





When ever we execute 1000 test scripts in batch, out of 1000 / 100 test scripts got failed , next time when a get new build , I wanted execute only failed test scripts , then is your approach ?

After the batach execution, refresh the project & go to test-output folder which is created inside the projectFolder , then execute “test-failed.xml” file

- ⇨ Testng-failed.xml file is created automatically by testing-tool itself for every failed execution reports .