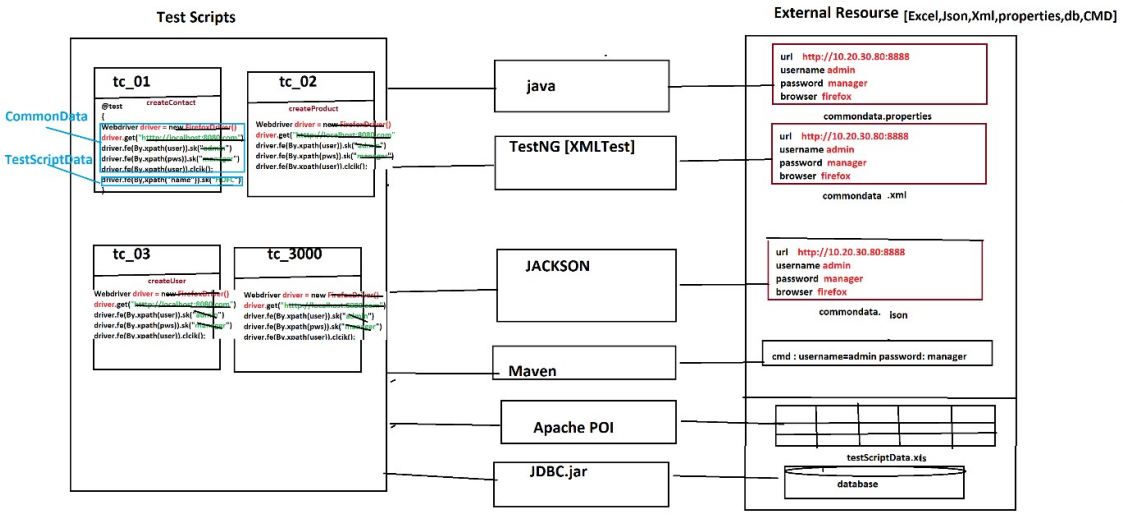
Data Driven testing

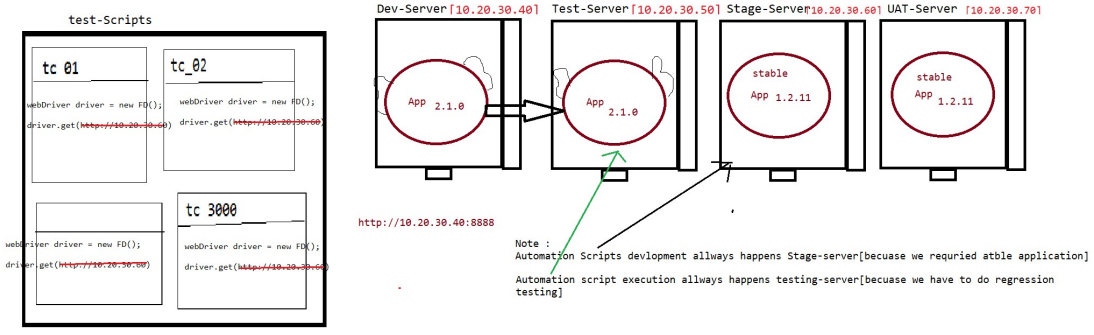
1. What is Data driven testing?

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 testDatateam)
    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

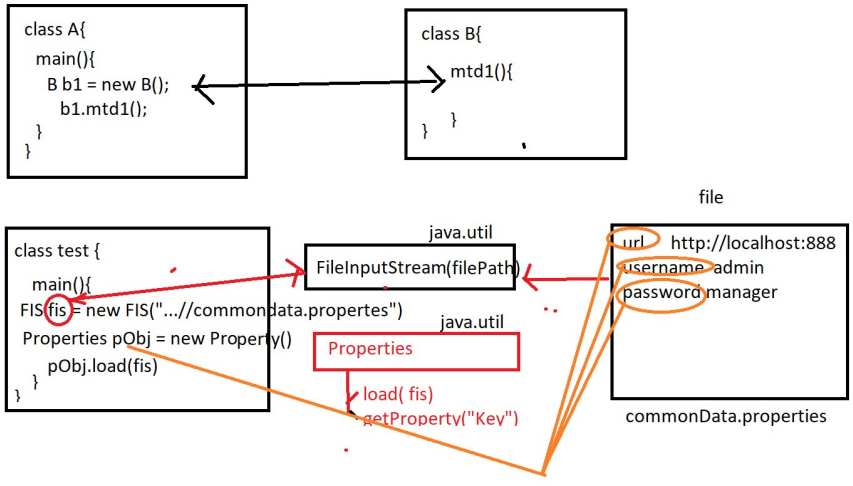
2. Why Property file ?

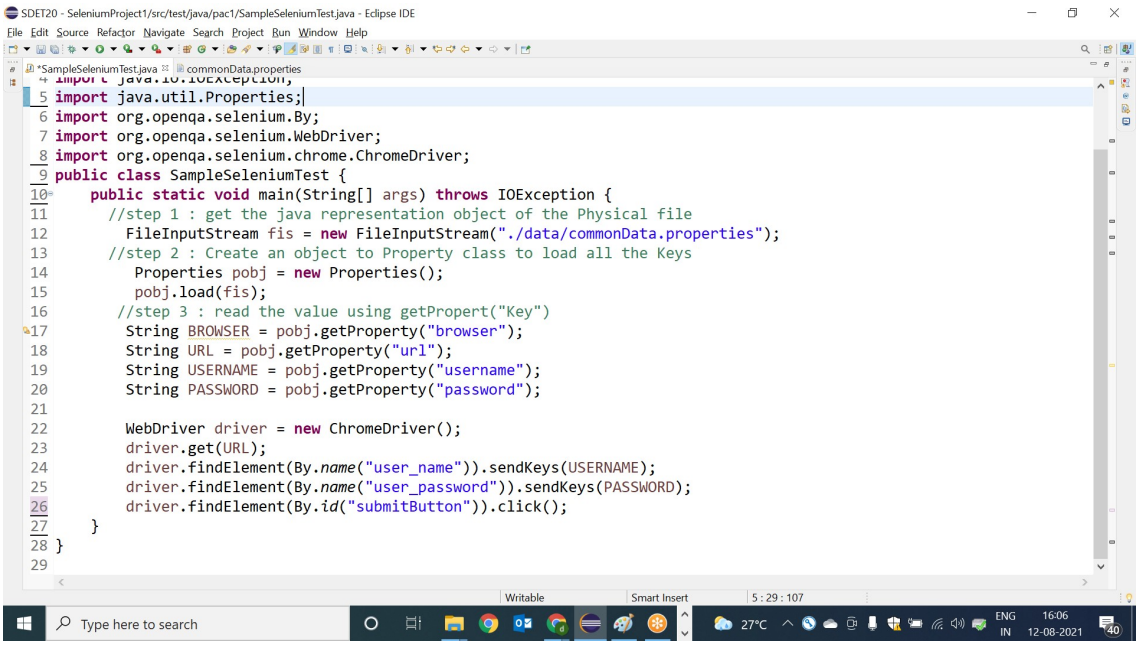
Property file is light weight & faster to read the data compare to any other file ,& java as own Class to read the data from property

3. How to read data from properties File?

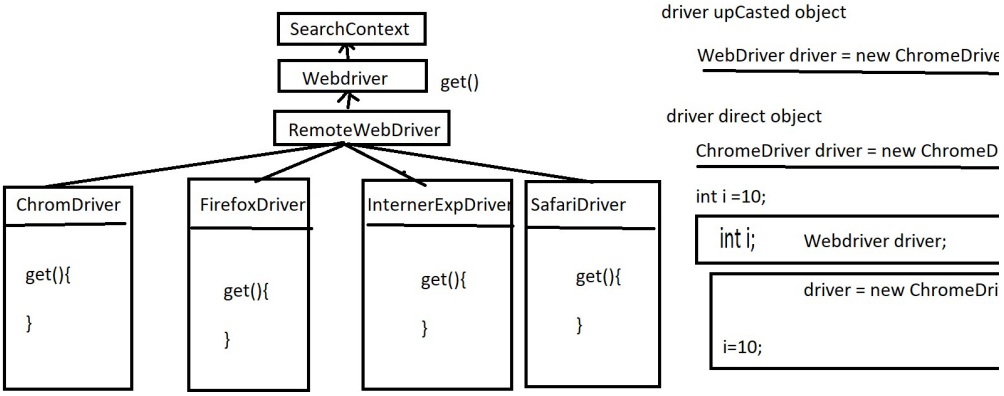
- Get the java representation Object of the Physical file using “FileInputSteam
- Create a Object of “Properties” class & load all the keys
- Read the data using getProperty(“Key”)

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

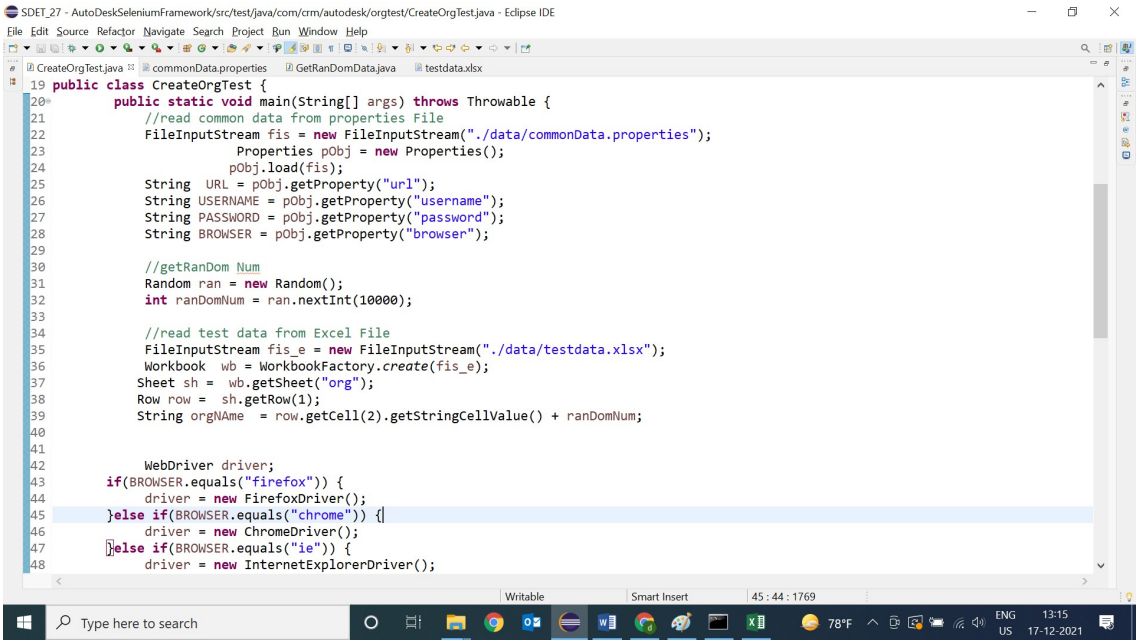
Assignment:

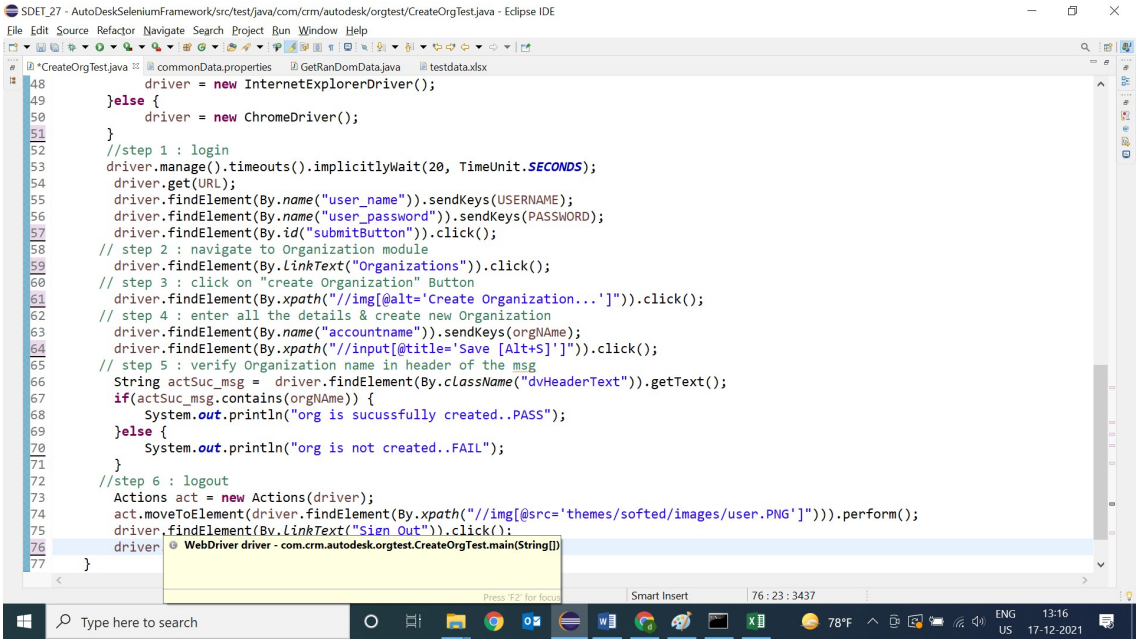
TestCase :CreateOrganization

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 , pptetc

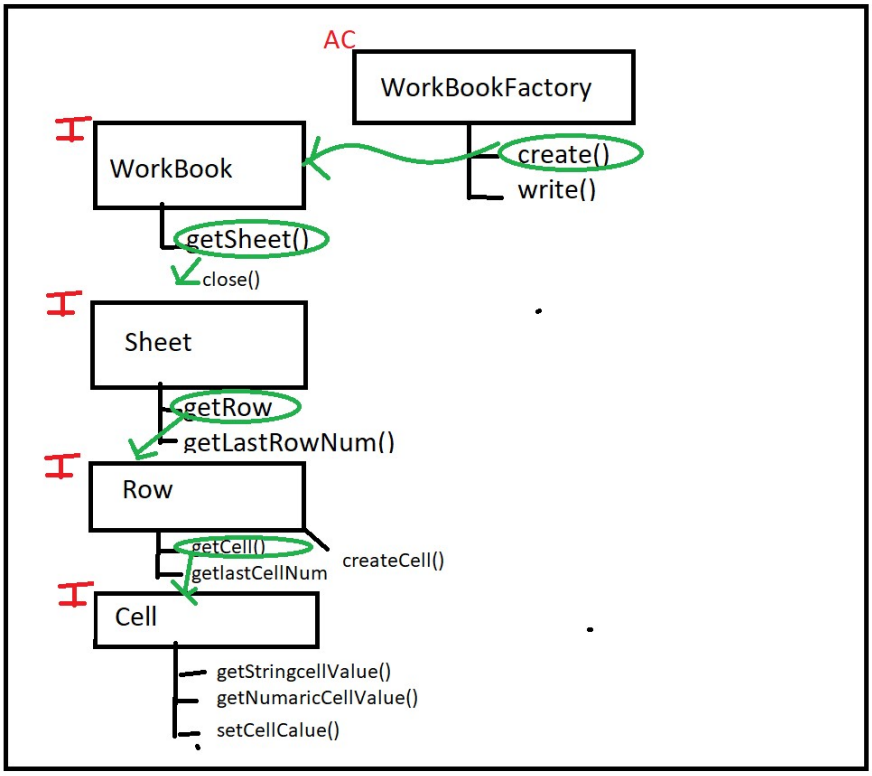
➔In real time most the company preferred the keep the test script data in Excel, because data will be in well-organized manner , so that modification & maintenance is easier.

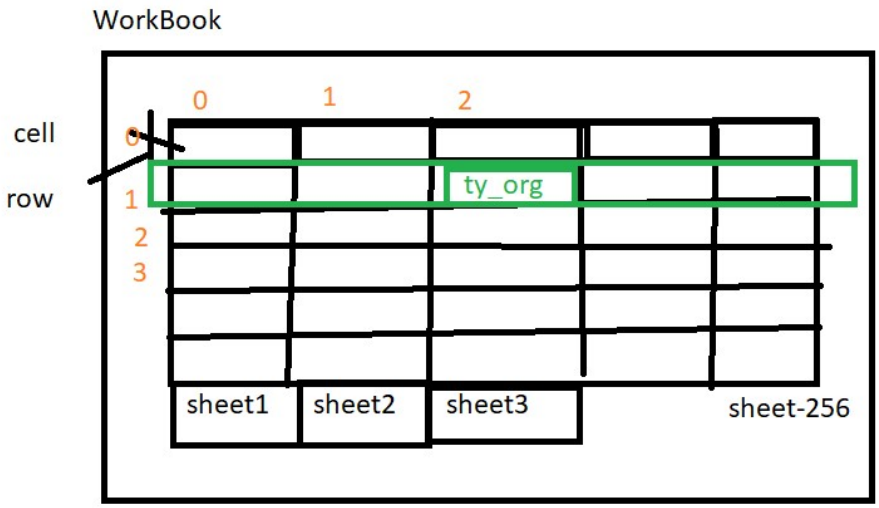
Installation steps:

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

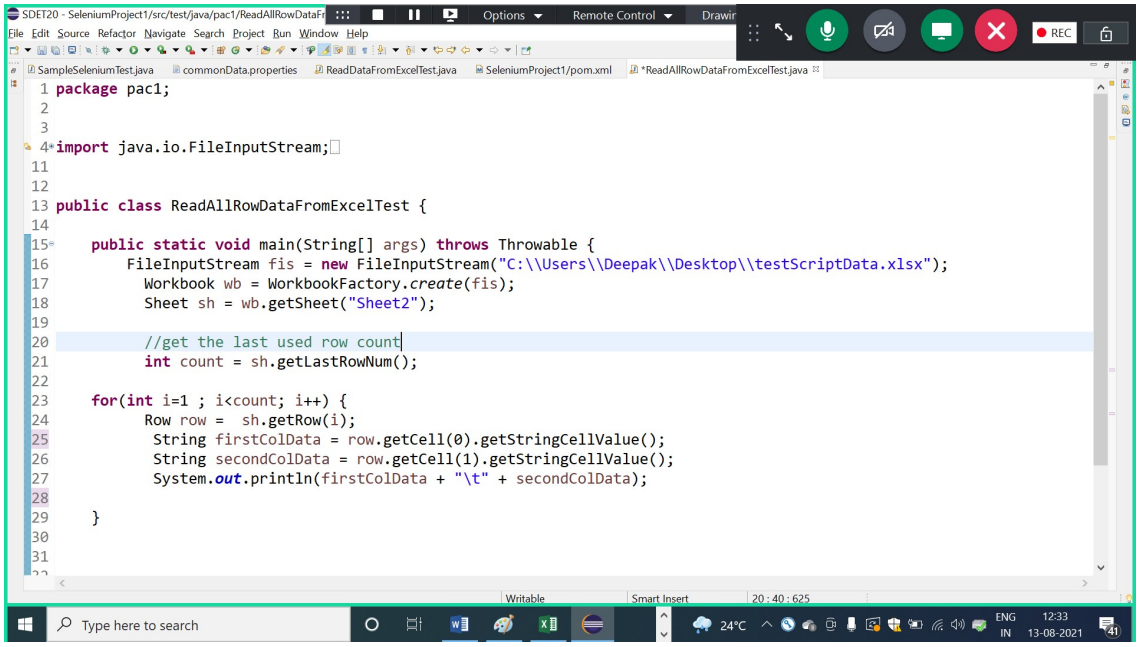




1. Program to read the data from WorkBook

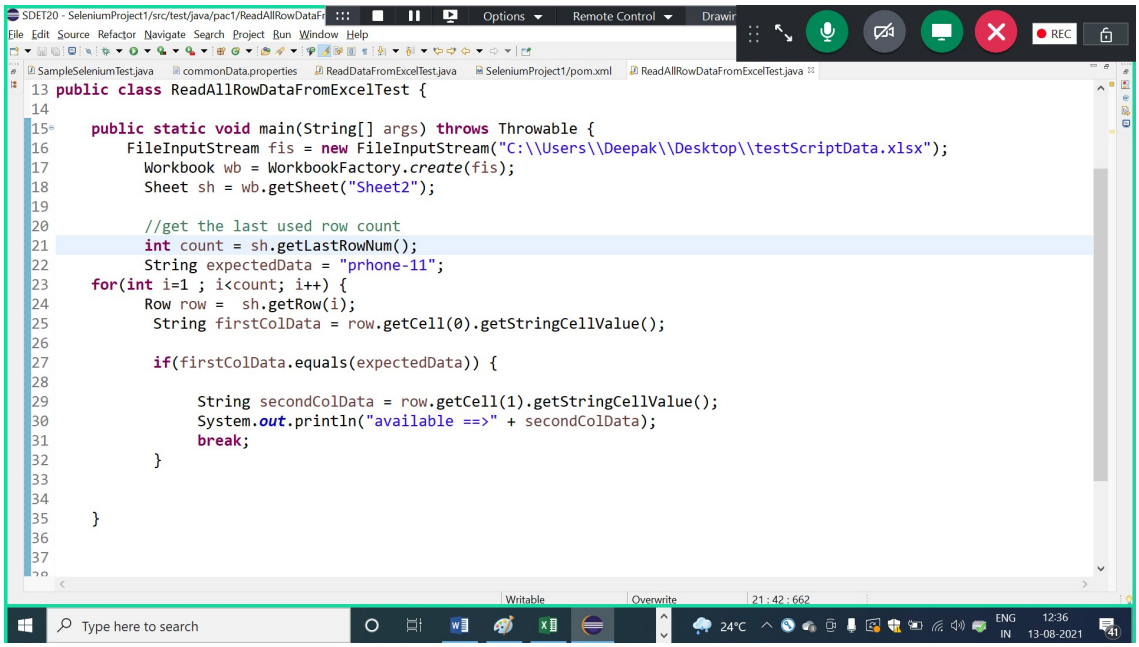
```java
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;


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 ndcoloum data from all the Rows



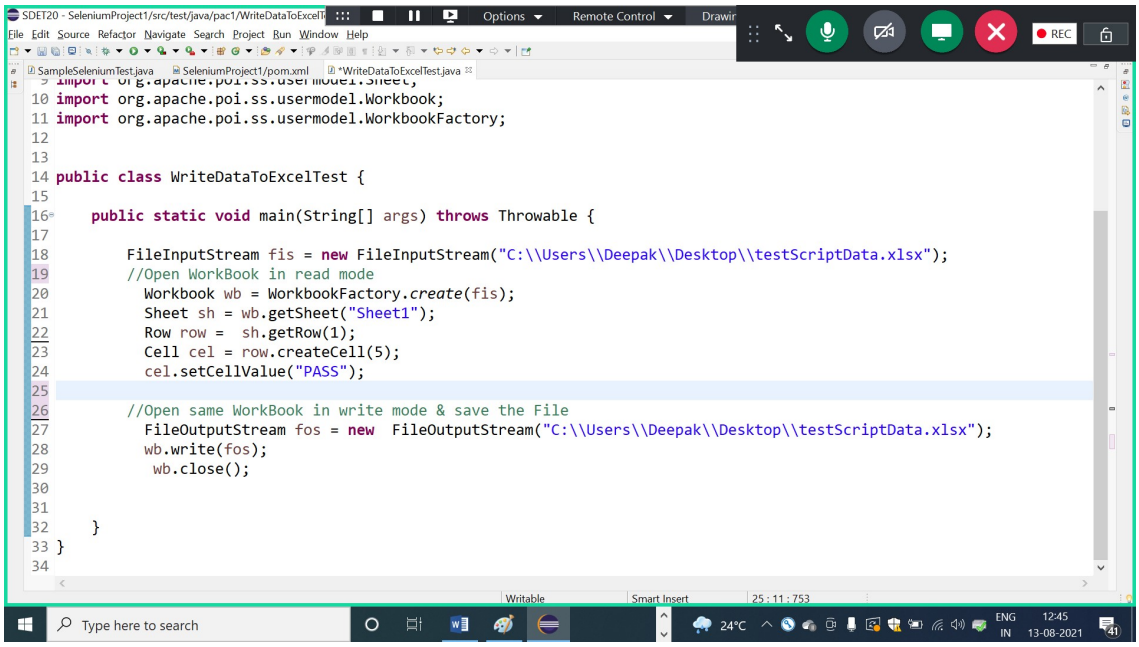
```
package pac1;


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 stcolumn , if data is available then read &dispaly next cell data

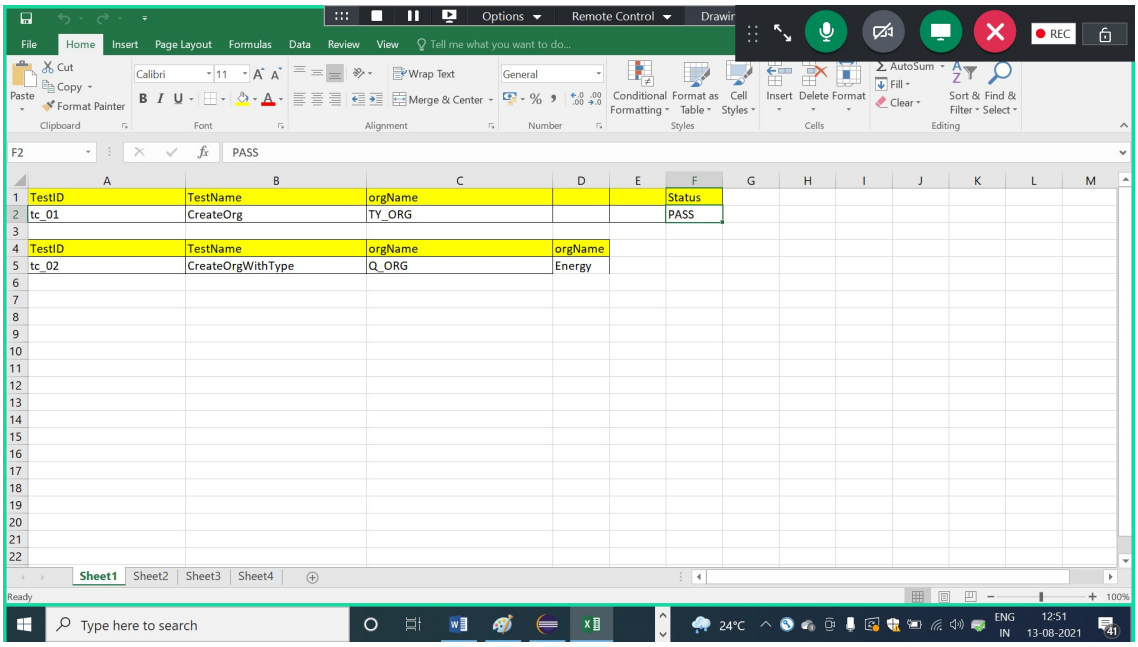


```
public class ReadAllRowDataFromExcelTest {

    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

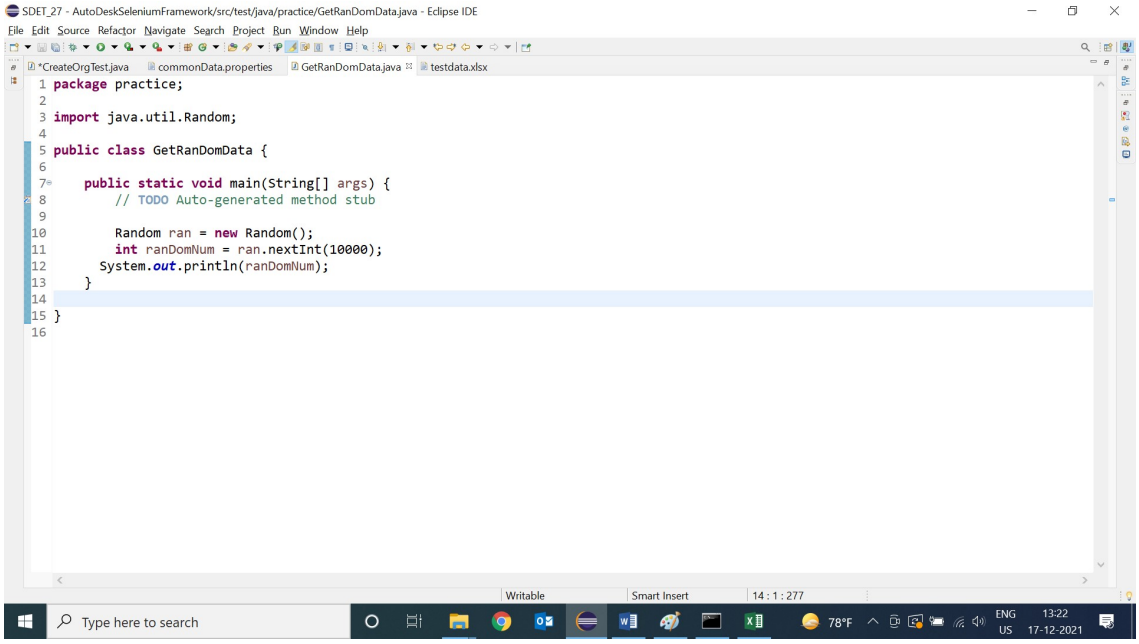


```java
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;


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

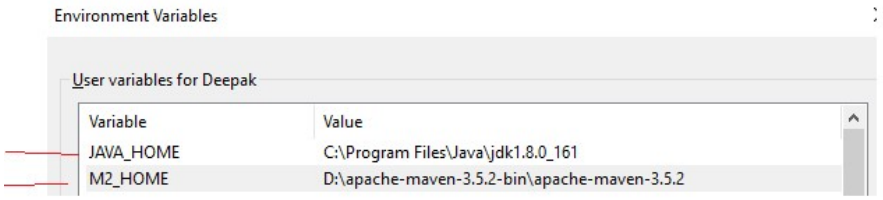
Why Java Main method argument is String array ?

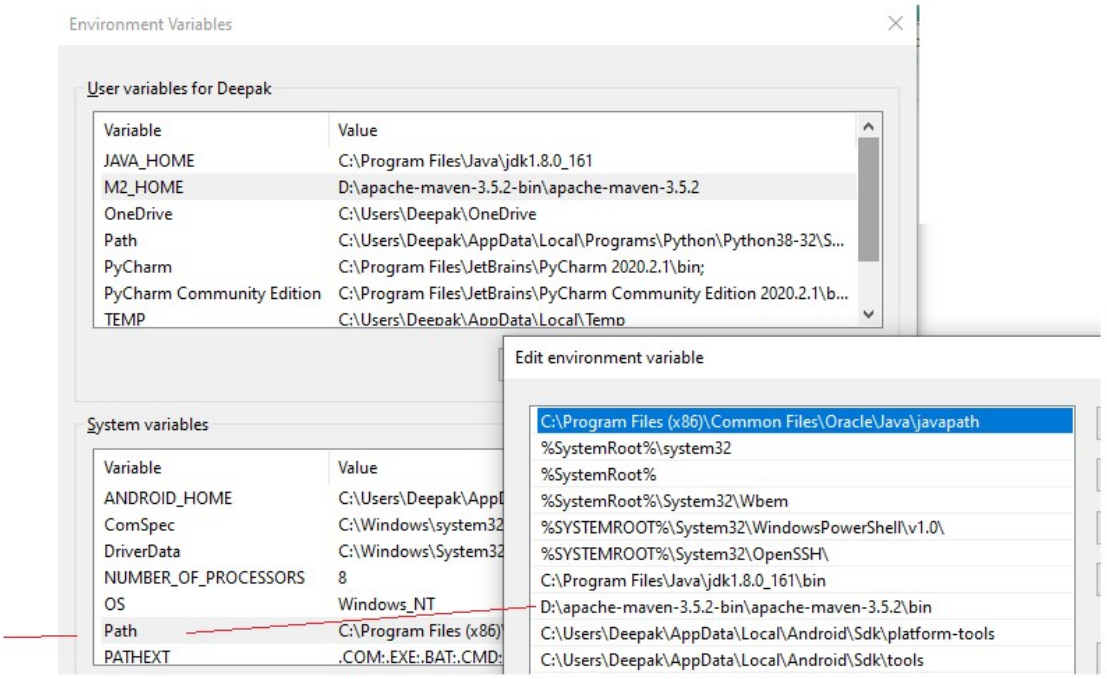
Because its always accept String array of data from the cammadline

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

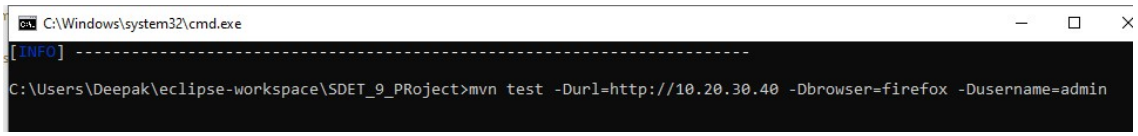
2. Create a Maven Project in Eclipse

3. Add testNG dependency in POM.xml

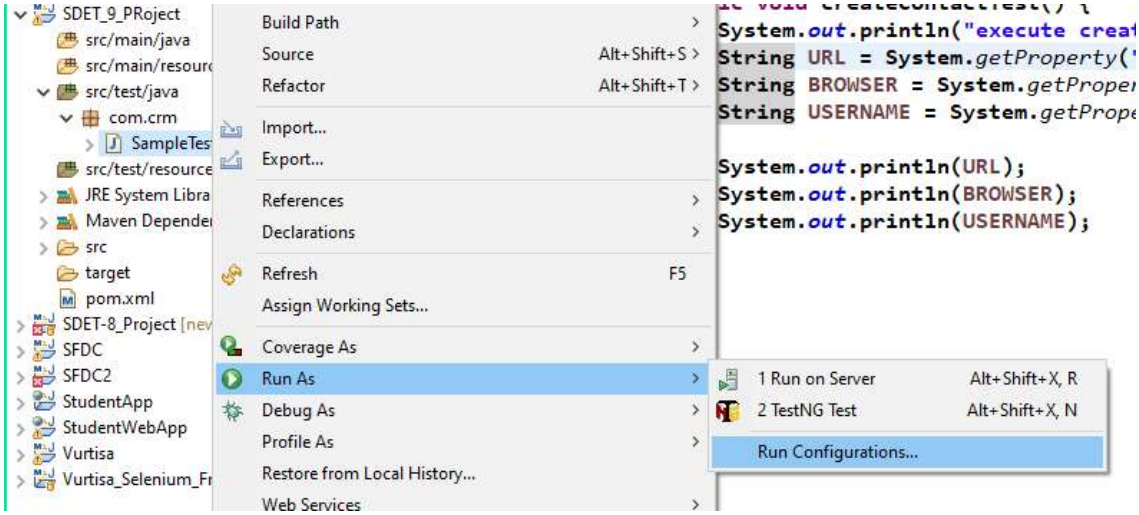
4. Create TestNg class

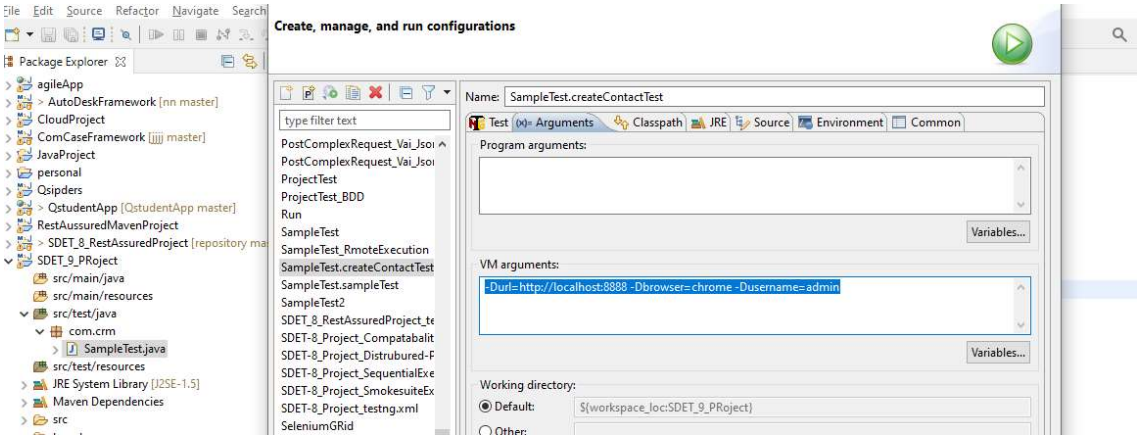
```
public class SampleTest {

    @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



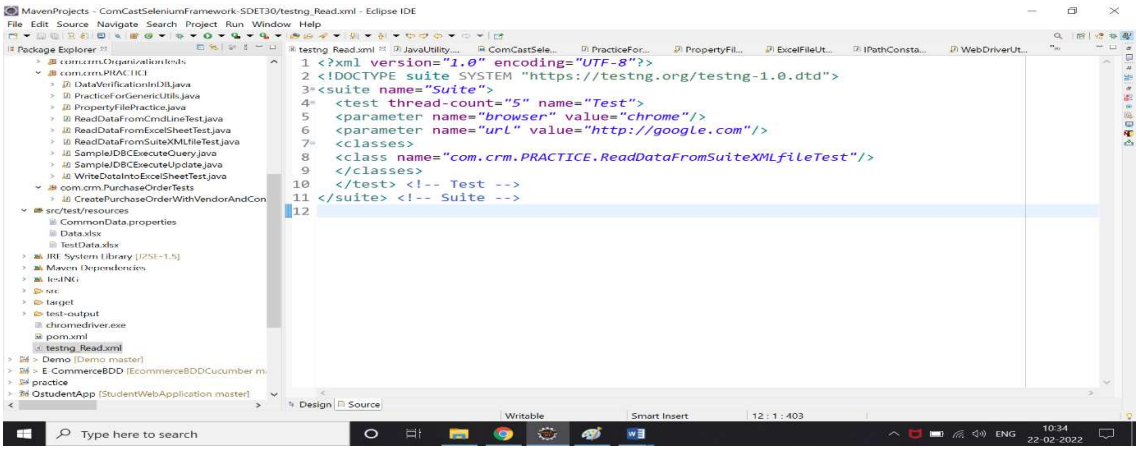
5. How to get Maven Parameters(data like url , username , password) from the Eclipse

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().