Generic Libraries

**What is Generic components in Automation Framework?**

🡺it’s one of the automation framework components which is common for all the application

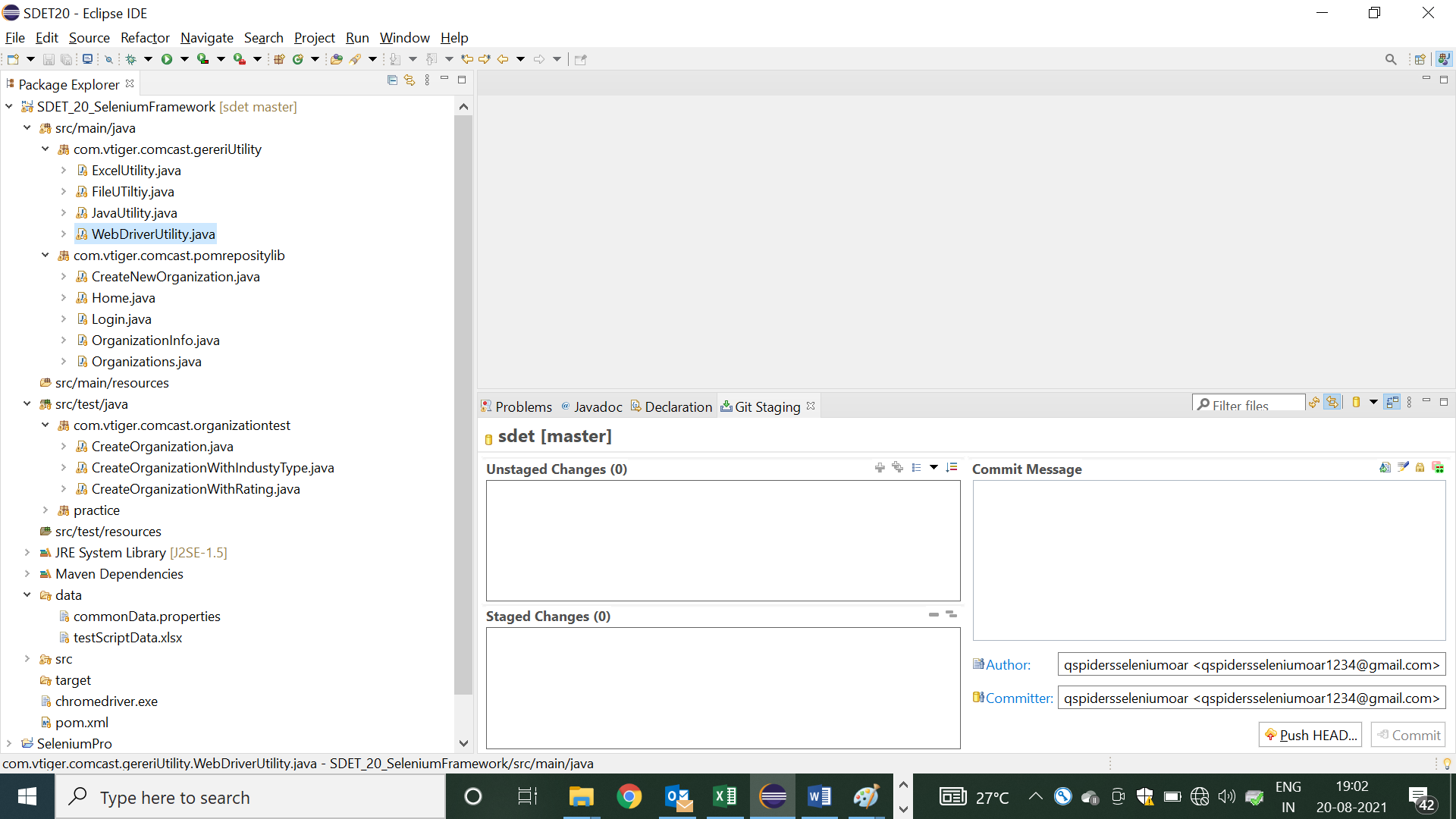
🡺its collection of generic class contains reusable methods / libraries

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

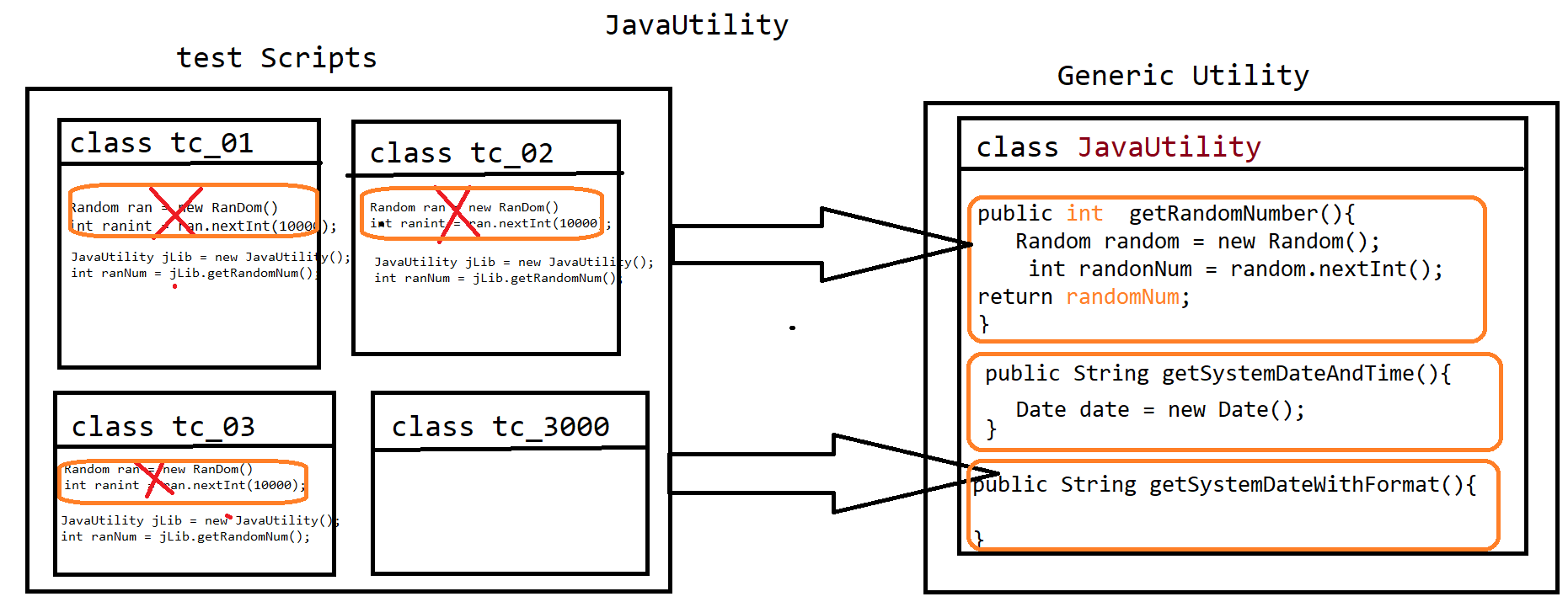
**What is the advantages of Generic components?**

* 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 {

/\*\*

\* it’s 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;

}

/\*\*

\* it’s used to get the current System date & time

\* @return

\*/

public String getSystmeDate() {

Date date = new Date();

String systemDateAndTime = date.toString();

return systemDateAndTime;

}

/\*\*

\* it’s 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();

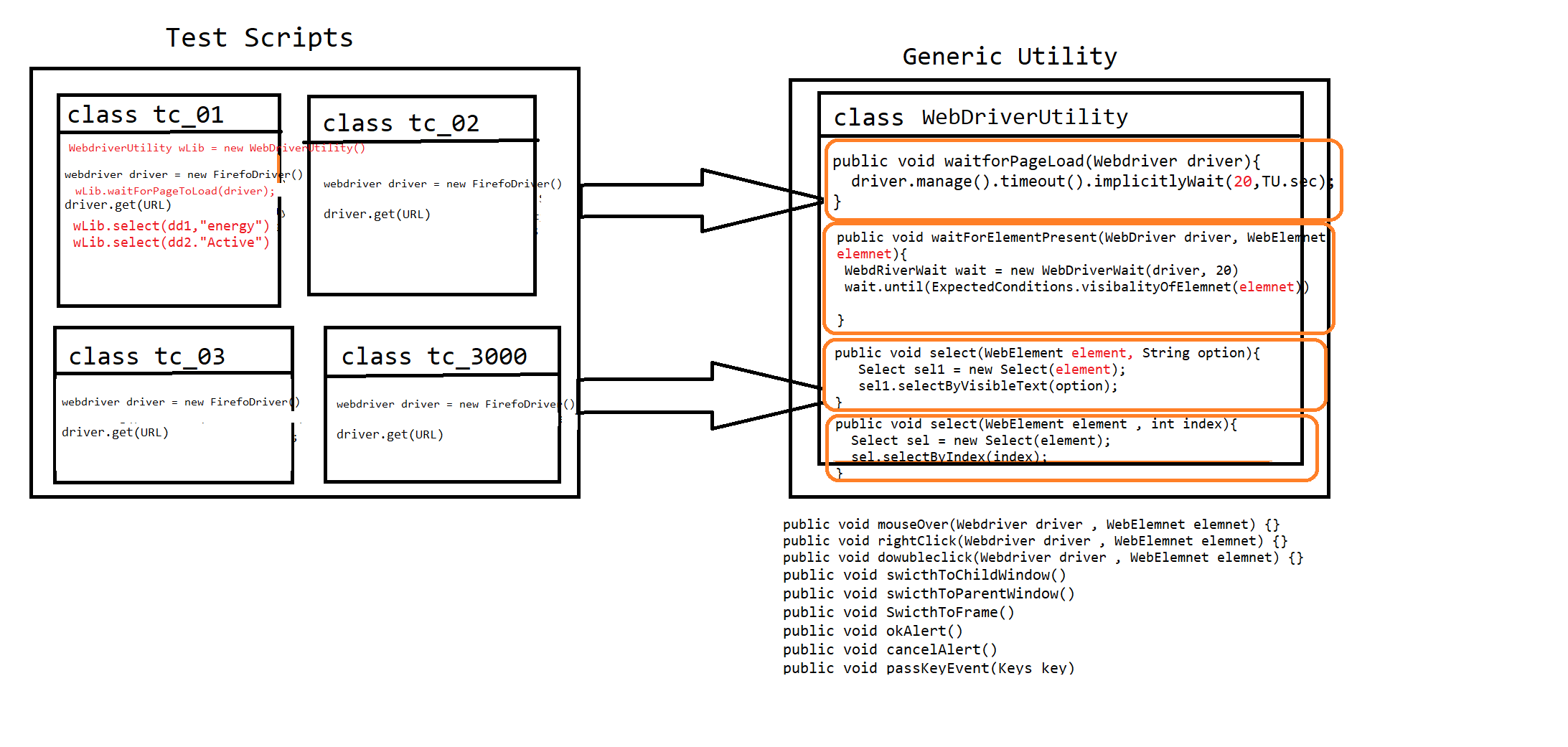
rc.keyPress(KeyEvent.VK\_ENTER);

rc.keyRelease(KeyEvent.VK\_ENTER);

}

}

1. WebDriver Utility Libraries



* Webdriver Utility 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 waits for 20 secs for page loading

\* @param driver

\*/

public void waitUntilPageLoad(WebDriver driver)

{

driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

}

/\*\*

\* This method waits 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 waits 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 method 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 method 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) {

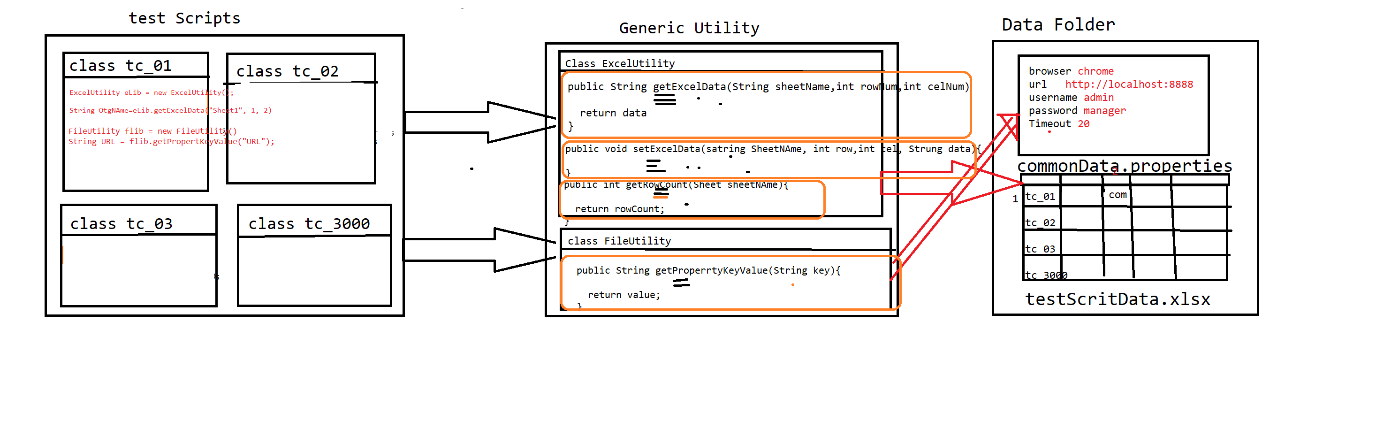
Actions act = new Actions(driver);

act.sendKeys(Keys.ENTER).perform();

}

}

1. 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 & .properties 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 to read the data from excel based on 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 {

/\*\*

\* it’s 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;

**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();

}

}