Table of Contents

[1. Create 1](#_Toc519160564)

[2. Copy below code under "Xls\_Reader" 1](#_Toc519160565)

[3. Under teste.java paste below code 6](#_Toc519160566)

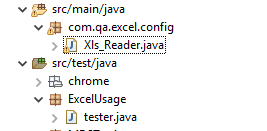
[4. Create an Excel file as below 7](#_Toc519160567)

[5. Output 7](#_Toc519160568)

Excel file Read and Write

# Create

In maven project create packages(com.qa.excel.config & ExcelUasge) and classes(Xls\_Reader & tester) shown below



## Copy below code under "Xls\_Reader"

package com.qa.excel.config;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.util.Calendar;

import org.apache.poi.hssf.usermodel.HSSFCellStyle;

import org.apache.poi.hssf.usermodel.HSSFDateUtil;

import org.apache.poi.hssf.util.HSSFColor;

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

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

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

import org.apache.poi.xssf.usermodel.XSSFCell;

import org.apache.poi.xssf.usermodel.XSSFCellStyle;

import org.apache.poi.xssf.usermodel.XSSFCreationHelper;

import org.apache.poi.xssf.usermodel.XSSFFont;

import org.apache.poi.xssf.usermodel.XSSFHyperlink;

import org.apache.poi.xssf.usermodel.XSSFRow;

import org.apache.poi.xssf.usermodel.XSSFSheet;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class Xls\_Reader {

//public static String filename = System.getProperty("user.dir")+"\\src\\com\\qtpselenium\\xlsx\\Suite.xlsx";

public String path;

public FileInputStream fis = null;

public FileOutputStream fileOut =null;

private XSSFWorkbook workbook = null;

private XSSFSheet sheet = null;

private XSSFRow row =null;

private XSSFCell cell = null;

public Xls\_Reader(String path) {

this.path=path;

try {

fis = new FileInputStream(path);

workbook = new XSSFWorkbook(fis);

sheet = workbook.getSheetAt(0);

fis.close();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

// returns the row count in a sheet

public int getRowCount(String sheetName){

int index = workbook.getSheetIndex(sheetName);

if(index==-1)

return 0;

else{

sheet = workbook.getSheetAt(index);

int number=sheet.getLastRowNum()+1;

return number;

}

}

// returns the data from a cell

public String getCellData(String sheetName,String colName,int rowNum){

try{

if(rowNum <=0)

return "";

int index = workbook.getSheetIndex(sheetName);

int col\_Num=-1;

if(index==-1)

return "";

sheet = workbook.getSheetAt(index);

row=sheet.getRow(0);

for(int i=0;i<row.getLastCellNum();i++){

//System.out.println(row.getCell(i).getStringCellValue().trim());

if(row.getCell(i).getStringCellValue().trim().equals(colName.trim()))

col\_Num=i;

}

if(col\_Num==-1)

return "";

sheet = workbook.getSheetAt(index);

row = sheet.getRow(rowNum-1);

if(row==null)

return "";

cell = row.getCell(col\_Num);

if(cell==null)

return "";

//System.out.println(cell.getCellType());

if(cell.getCellType()==Cell.CELL\_TYPE\_STRING)

return cell.getStringCellValue();

else if(cell.getCellType()==Cell.CELL\_TYPE\_NUMERIC || cell.getCellType()==Cell.CELL\_TYPE\_FORMULA ){

String cellText = String.valueOf(cell.getNumericCellValue());

if (HSSFDateUtil.isCellDateFormatted(cell)) {

// format in form of M/D/YY

double d = cell.getNumericCellValue();

Calendar cal =Calendar.getInstance();

cal.setTime(HSSFDateUtil.getJavaDate(d));

cellText =

(String.valueOf(cal.get(Calendar.YEAR))).substring(2);

cellText = cal.get(Calendar.DAY\_OF\_MONTH) + "/" +

cal.get(Calendar.MONTH)+1 + "/" +

cellText;

//System.out.println(cellText);

}

return cellText;

}else if(cell.getCellType()==Cell.CELL\_TYPE\_BLANK)

return "";

else

return String.valueOf(cell.getBooleanCellValue());

}

catch(Exception e){

e.printStackTrace();

return "row "+rowNum+" or column "+colName +" does not exist in xls";

}

}

// returns the data from a cell

public String getCellData(String sheetName,int colNum,int rowNum){

try{

if(rowNum <=0)

return "";

int index = workbook.getSheetIndex(sheetName);

if(index==-1)

return "";

sheet = workbook.getSheetAt(index);

row = sheet.getRow(rowNum-1);

if(row==null)

return "";

cell = row.getCell(colNum);

if(cell==null)

return "";

if(cell.getCellType()==Cell.CELL\_TYPE\_STRING)

return cell.getStringCellValue();

else if(cell.getCellType()==Cell.CELL\_TYPE\_NUMERIC || cell.getCellType()==Cell.CELL\_TYPE\_FORMULA ){

String cellText = String.valueOf(cell.getNumericCellValue());

if (HSSFDateUtil.isCellDateFormatted(cell)) {

// format in form of M/D/YY

double d = cell.getNumericCellValue();

Calendar cal =Calendar.getInstance();

cal.setTime(HSSFDateUtil.getJavaDate(d));

cellText =

(String.valueOf(cal.get(Calendar.YEAR))).substring(2);

cellText = cal.get(Calendar.MONTH)+1 + "/" +

cal.get(Calendar.DAY\_OF\_MONTH) + "/" +

cellText;

// System.out.println(cellText);

}

return cellText;

}else if(cell.getCellType()==Cell.CELL\_TYPE\_BLANK)

return "";

else

return String.valueOf(cell.getBooleanCellValue());

}

catch(Exception e){

e.printStackTrace();

return "row "+rowNum+" or column "+colNum +" does not exist in xls";

}

}

// returns true if data is set successfully else false

public boolean setCellData(String sheetName,String colName,int rowNum, String data){

try{

fis = new FileInputStream(path);

workbook = new XSSFWorkbook(fis);

if(rowNum<=0)

return false;

int index = workbook.getSheetIndex(sheetName);

int colNum=-1;

if(index==-1)

return false;

sheet = workbook.getSheetAt(index);

row=sheet.getRow(0);

for(int i=0;i<row.getLastCellNum();i++){

//System.out.println(row.getCell(i).getStringCellValue().trim());

if(row.getCell(i).getStringCellValue().trim().equals(colName))

colNum=i;

}

if(colNum==-1)

return false;

sheet.autoSizeColumn(colNum);

row = sheet.getRow(rowNum-1);

if (row == null)

row = sheet.createRow(rowNum-1);

cell = row.getCell(colNum);

if (cell == null)

cell = row.createCell(colNum);

// cell style

//CellStyle cs = workbook.createCellStyle();

//cs.setWrapText(true);

//cell.setCellStyle(cs);

cell.setCellValue(data);

fileOut = new FileOutputStream(path);

workbook.write(fileOut);

fileOut.close();

}

catch(Exception e){

e.printStackTrace();

return false;

}

return true;

}

// returns true if data is set successfully else false

public boolean setCellData(String sheetName,String colName,int rowNum, String data,String url){

//System.out.println("setCellData setCellData\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

try{

fis = new FileInputStream(path);

workbook = new XSSFWorkbook(fis);

if(rowNum<=0)

return false;

int index = workbook.getSheetIndex(sheetName);

int colNum=-1;

if(index==-1)

return false;

sheet = workbook.getSheetAt(index);

//System.out.println("A");

row=sheet.getRow(0);

for(int i=0;i<row.getLastCellNum();i++){

//System.out.println(row.getCell(i).getStringCellValue().trim());

if(row.getCell(i).getStringCellValue().trim().equalsIgnoreCase(colName))

colNum=i;

}

if(colNum==-1)

return false;

sheet.autoSizeColumn(colNum); //ashish

row = sheet.getRow(rowNum-1);

if (row == null)

row = sheet.createRow(rowNum-1);

cell = row.getCell(colNum);

if (cell == null)

cell = row.createCell(colNum);

cell.setCellValue(data);

XSSFCreationHelper createHelper = workbook.getCreationHelper();

//cell style for hyperlinks

//by default hypelrinks are blue and underlined

CellStyle hlink\_style = workbook.createCellStyle();

XSSFFont hlink\_font = workbook.createFont();

hlink\_font.setUnderline(XSSFFont.U\_SINGLE);

hlink\_font.setColor(IndexedColors.BLUE.getIndex());

hlink\_style.setFont(hlink\_font);

//hlink\_style.setWrapText(true);

XSSFHyperlink link = createHelper.createHyperlink(XSSFHyperlink.LINK\_FILE);

link.setAddress(url);

cell.setHyperlink(link);

cell.setCellStyle(hlink\_style);

fileOut = new FileOutputStream(path);

workbook.write(fileOut);

fileOut.close();

}

catch(Exception e){

e.printStackTrace();

return false;

}

return true;

}

// returns true if sheet is created successfully else false

public boolean addSheet(String sheetname){

FileOutputStream fileOut;

try {

workbook.createSheet(sheetname);

fileOut = new FileOutputStream(path);

workbook.write(fileOut);

fileOut.close();

} catch (Exception e) {

e.printStackTrace();

return false;

}

return true;

}

// returns true if sheet is removed successfully else false if sheet does not exist

public boolean removeSheet(String sheetName){

int index = workbook.getSheetIndex(sheetName);

if(index==-1)

return false;

FileOutputStream fileOut;

try {

workbook.removeSheetAt(index);

fileOut = new FileOutputStream(path);

workbook.write(fileOut);

fileOut.close();

} catch (Exception e) {

e.printStackTrace();

return false;

}

return true;

}

// returns true if column is created successfully

public boolean addColumn(String sheetName,String colName){

//System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*addColumn\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

try{

fis = new FileInputStream(path);

workbook = new XSSFWorkbook(fis);

int index = workbook.getSheetIndex(sheetName);

if(index==-1)

return false;

XSSFCellStyle style = workbook.createCellStyle();

style.setFillForegroundColor(HSSFColor.GREY\_40\_PERCENT.index);

style.setFillPattern(HSSFCellStyle.SOLID\_FOREGROUND);

sheet=workbook.getSheetAt(index);

row = sheet.getRow(0);

if (row == null)

row = sheet.createRow(0);

//cell = row.getCell();

//if (cell == null)

//System.out.println(row.getLastCellNum());

if(row.getLastCellNum() == -1)

cell = row.createCell(0);

else

cell = row.createCell(row.getLastCellNum());

cell.setCellValue(colName);

cell.setCellStyle(style);

fileOut = new FileOutputStream(path);

workbook.write(fileOut);

fileOut.close();

}catch(Exception e){

e.printStackTrace();

return false;

}

return true;

}

// removes a column and all the contents

public boolean removeColumn(String sheetName, int colNum) {

try{

if(!isSheetExist(sheetName))

return false;

fis = new FileInputStream(path);

workbook = new XSSFWorkbook(fis);

sheet=workbook.getSheet(sheetName);

XSSFCellStyle style = workbook.createCellStyle();

style.setFillForegroundColor(HSSFColor.GREY\_40\_PERCENT.index);

XSSFCreationHelper createHelper = workbook.getCreationHelper();

style.setFillPattern(HSSFCellStyle.NO\_FILL);

for(int i =0;i<getRowCount(sheetName);i++){

row=sheet.getRow(i);

if(row!=null){

cell=row.getCell(colNum);

if(cell!=null){

cell.setCellStyle(style);

row.removeCell(cell);

}

}

}

fileOut = new FileOutputStream(path);

workbook.write(fileOut);

fileOut.close();

}

catch(Exception e){

e.printStackTrace();

return false;

}

return true;

}

// find whether sheets exists

public boolean isSheetExist(String sheetName){

int index = workbook.getSheetIndex(sheetName);

if(index==-1){

index=workbook.getSheetIndex(sheetName.toUpperCase());

if(index==-1)

return false;

else

return true;

}

else

return true;

}

// returns number of columns in a sheet

public int getColumnCount(String sheetName){

// check if sheet exists

if(!isSheetExist(sheetName))

return -1;

sheet = workbook.getSheet(sheetName);

row = sheet.getRow(0);

if(row==null)

return -1;

return row.getLastCellNum();

}

//String sheetName, String testCaseName,String keyword ,String URL,String message

public boolean addHyperLink(String sheetName,String screenShotColName,String testCaseName,int index,String url,String message){

//System.out.println("ADDING addHyperLink\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

url=url.replace('\\', '/');

if(!isSheetExist(sheetName))

return false;

sheet = workbook.getSheet(sheetName);

for(int i=2;i<=getRowCount(sheetName);i++){

if(getCellData(sheetName, 0, i).equalsIgnoreCase(testCaseName)){

//System.out.println("\*\*caught "+(i+index));

setCellData(sheetName, screenShotColName, i+index, message,url);

break;

}

}

return true;

}

public int getCellRowNum(String sheetName,String colName,String cellValue){

for(int i=2;i<=getRowCount(sheetName);i++){

if(getCellData(sheetName,colName , i).equalsIgnoreCase(cellValue)){

return i;

}

}

return -1;

}

// to run this on stand alone

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

//System.out.println(filename);

Xls\_Reader datatable = null;

/\* datatable = new Xls\_Reader(System.getProperty("user.dir")+"\\src\\com\\qtpselenium\\xls\\Controller.xlsx");

for(int col=0 ;col< datatable.getColumnCount("TC5"); col++){

System.out.println(datatable.getCellData("TC5", col, 1));

} \*/

}

}

# Under teste.java paste below code

package ExcelUsage;

import com.qa.excel.config.Xls\_Reader;

public class tester {

public static void main(String[] args) {

Xls\_Reader reader = new Xls\_Reader("P:\\Selenium\\POMLearning.xlsx");

//Enter Sheet Name

int rowCount = reader.getRowCount("Sheet1");

System.*out*.println(rowCount);

for(int rowNum=2; rowNum<=rowCount; rowNum++){

String FirstName = reader.getCellData("Sheet1", "FirstName", rowNum);

System.*out*.println(FirstName);

String LastName = reader.getCellData("Sheet1", "LastName", rowNum);

System.*out*.println(LastName);

String EmailAdd = reader.getCellData("Sheet1", "EmailAdd", rowNum);

System.*out*.println(EmailAdd);

String Password1 = reader.getCellData("Sheet1", "Password1", rowNum);

System.*out*.println(Password1);

String Password2 = reader.getCellData("Sheet1", "Password2", rowNum);

System.*out*.println(Password2);

reader.setCellData("Sheet1","Status",rowNum,"pass");

}

}

}

# Create an Excel file as below

**Copy the path (**P:\\Selenium\\POMLearning.xlsx) **of excel file and paste in above code**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FirstName | LastName | EmailAdd | Password1 | Password2 | Status |
| FN01 | LN01 | EA01 | Pass01 | Pass10 | pass |
| FN02 | LN02 | EA02 | Pass02 | Pass11 | pass |
| FN03 | LN03 | EA03 | Pass03 | Pass12 | pass |
| FN04 | LN04 | EA04 | Pass04 | Pass13 | pass |
| FN05 | LN05 | EA05 | Pass05 | Pass14 | pass |

# Output

6

FN01

LN01

EA01

Pass01

Pass10

FN02

LN02

EA02

Pass02

Pass11

FN03

LN03

EA03

Pass03

Pass12

FN04

LN04

EA04

Pass04

Pass13

FN05

LN05

EA05

Pass05

Pass14