What is Apache POI?

**Apache POI** (Poor Obfuscation Implementation) is a Java API for reading and writing Microsoft Documents in both formats **.xls** and **.xlsx**. It contains classes and interfaces. The Apache POI library provides two implementations for reading excel files:

* **HSSF (Horrible SpreadSheet Format) Implementation:** It denotes an API that is working with Excel 2003 or earlier versions.
* **XSSF (XML SpreadSheet Format) Implementation:** It denotes an API that is working with Excel 2007 or later versions.

## Interfaces and Classes in Apache POI

### **Interfaces**

* **Workbook:** It represents an **Excel Workbook**. It is an interface implement by **HSSFWorkbook** and **XSSFWorkbook**.
* **Sheet:** It is an interface that represents an **Excel worksheet**. A sheet is a central structure of a workbook, which represents a grid of cells. The Sheet interface extends **java.lang.Iterable**.
* **Row:** It is also an interface that represents the **row** of the spreadsheet. The Row interface extends **java.lang.Iterable**. There are two concrete classes: **HSSFRow** and **XSSFRow**.
* **Cell:** It is an interface. It is a high-level representation of a **cell** in a row of the spreadsheet. **HSSFCell** and **XSSFCell** implement Cell interface.

### **Classes**

**XLS Classes**

* **HSSFWorkbook:** It is a class representing the XLS file.
* **HSSFSheet:** It is a class representing the sheet in an XLS file.
* **HSSFRow:** It is a class representing a row in the sheet of XLS file.
* **HSSFCell:** It is a class representing a cell in a row of XLS file.

**XLSX Classes**

* **XSSFWorkbook:** It is a class representing the XLSX file.
* **XSSFSheet:** It is a class representing the sheet in an XLSX file.
* **XSSFRow:** It is a class representing a row in the sheet of XLSX file.
* **XSSFCell:** It is a class representing a cell in a row of XLSX file.

## Steps to read data from XLS file

**Step 1:** Create a simple Java project in eclipse.

**Step 2:** Now, create a lib folder in the project.

**Step 3:** Download and add the following jar files in the lib folder:

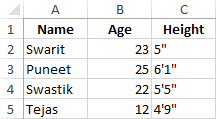
* commons-collections4-4.1.jar [**Click Here**](https://bit.ly/2SG4r3Y)
* poi-3.17.jar [**Click Here**](https://bit.ly/2Y6HRY9)
* poi-ooxml-3.17.jar [**Click Here**](https://bit.ly/2LJ1leO)
* poi-ooxml-schemas-3.17.jar [**Click Here**](https://bit.ly/2LHjeL9)
* xmlbeans-2.6.0.jar [**Click Here**](https://bit.ly/2ybqxBR)

**Step 4:** Set the Class Path:

Right-click on the project ->Build Path ->Add External JARs -> select all the above jar files -> Apply and close.

**Step 5:** Now create a class file with the name **ReadExcelFileDemo** and write the following code in the file.

**Step 6:** Create an excel file with the name "student.xls" and write some data into it.



**Step 7:** Save and run the program.

**Example of reading excel file (.xls) file**

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

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

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

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

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

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

public class ReadExcelFileDemo

{

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

{

//obtaining input bytes from a file

FileInputStream fis=new FileInputStream(new File("C:\\demo\\student.xls"));

//creating workbook instance that refers to .xls file

HSSFWorkbook wb=new HSSFWorkbook(fis);

//creating a Sheet object to retrieve the object

HSSFSheet sheet=wb.getSheetAt(0);

//evaluating cell type

FormulaEvaluator formulaEvaluator=wb.getCreationHelper().createFormulaEvaluator();

for(Row row: sheet) //iteration over row using for each loop

{

for(Cell cell: row) //iteration over cell using for each loop

{

switch(formulaEvaluator.evaluateInCell(cell).getCellType())

{

case Cell.CELL\_TYPE\_NUMERIC: //field that represents numeric cell type

//getting the value of the cell as a number

System.out.print(cell.getNumericCellValue()+ "\t\t");

break;

case Cell.CELL\_TYPE\_STRING: //field that represents string cell type

//getting the value of the cell as a string

System.out.print(cell.getStringCellValue()+ "\t\t");

break;

}

}

System.out.println();

}

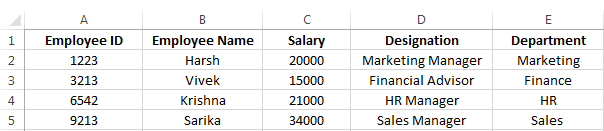
}

}

Reading XLSX File

All steps will remain same except file format.

**Table:** employee.xslx



**Example of read excel file (.xlsx)**

In this example we use XSSFWorkbook class.

import java.io.File;

import java.io.FileInputStream;

import java.util.Iterator;

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

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

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

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

public class XLSXReaderExample

{

public static void main(String[] args)

{

try

{

File file = new File("C:\\demo\\employee.xlsx"); //creating a new file instance

FileInputStream fis = new FileInputStream(file); //obtaining bytes from the file

//creating Workbook instance that refers to .xlsx file

XSSFWorkbook wb = new XSSFWorkbook(fis);

XSSFSheet sheet = wb.getSheetAt(0); //creating a Sheet object to retrieve object

Iterator<Row> itr = sheet.iterator(); //iterating over excel file

while (itr.hasNext())

{

Row row = itr.next();

Iterator<Cell> cellIterator = row.cellIterator(); //iterating over each column

while (cellIterator.hasNext())

{

Cell cell = cellIterator.next();

switch (cell.getCellType())

{

case Cell.CELL\_TYPE\_STRING: //field that represents string cell type

System.out.print(cell.getStringCellValue() + "\t\t\t");

break;

case Cell.CELL\_TYPE\_NUMERIC: //field that represents number cell type

System.out.print(cell.getNumericCellValue() + "\t\t\t");

break;

default:

}

}

System.out.println("");

}

}

catch(Exception e)

{

e.printStackTrace();

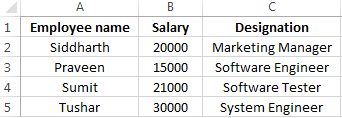
}

}

}

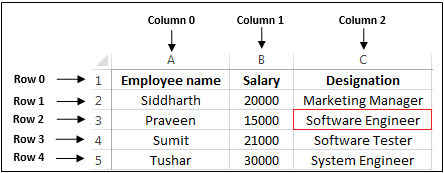
Reading a particular cell value from a excel file (.xlsx)

**Table:** EmployeeData.xlsx



**Example**

In the following example, we read the value of the 2nd row and the 2nd column. The row and column counting start from 0. So the program returns "Software Engineer."



//reading value of a particular cell

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

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

import org.apache.poi.ss.usermodel.\*;

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

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

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

public class ReadCellExample

{

public static void main(String[] args)

{

ReadCellExample rc=new ReadCellExample(); //object of the class

//reading the value of 2nd row and 2nd column

String vOutput=rc.ReadCellData(2, 2);

System.out.println(vOutput);

}

//method defined for reading a cell

public String ReadCellData(int vRow, int vColumn)

{

String value=null; //variable for storing the cell value

Workbook wb=null; //initialize Workbook null

try

{

//reading data from a file in the form of bytes

FileInputStream fis=new FileInputStream("C:\\demo\\EmployeeData.xlsx");

//constructs an XSSFWorkbook object, by buffering the whole stream into the memory

wb=new XSSFWorkbook(fis);

}

catch(FileNotFoundException e)

{

e.printStackTrace();

}

catch(IOException e1)

{

e1.printStackTrace();

}

Sheet sheet=wb.getSheetAt(0); //getting the XSSFSheet object at given index

Row row=sheet.getRow(vRow); //returns the logical row

Cell cell=row.getCell(vColumn); //getting the cell representing the given column

value=cell.getStringCellValue(); //getting cell value

return value; //returns the cell value

}

}

### **1. Read Excel file in Java**

In order to use Apache POI we need to add its dependencies, if you are using ANT you can download JARs from [here](https://poi.apache.org/download.html) and add it to your classpath.  
If you are using MAVEN you can add these dependencies into your POM.XML file :

pom.xml:

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10 | <dependency>      <groupId>org.apache.poi</groupId>      <artifactId>poi</artifactId>      <version>3.10-FINAL</version>  </dependency>      <dependency>      <groupId>org.apache.poi</groupId>      <artifactId>poi-ooxml</artifactId>      <version>3.10-FINAL</version>  </dependency> |

Once the dependencies are satisfied and project is build, lets create a test xlsx file which we will read from our Java code.

Below is the excel file :

This Excel file contains data for Students and their marks. We will read this data and store it in a list of Java Objects.

This Java class used here is Student :

package com.jcg.example;

/\*\*

\* Created by anirudh on 20/10/14.

\*/

public class Student {

private String name;

private String maths;

private String science;

private String english;

public Student(){}

public Student(String name, String maths, String science, String english) {

this.name = name;

this.maths = maths;

this.science = science;

this.english = english;

}

//getters and setter..

@Override

public String toString() {

return name+ ": Maths "+maths+ " Science "+science+" English "+english;

}

Now, lets see the Java Code to read this Excel file:

ReadExcelFileExample.java

package com.jcg.example;

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

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

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

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

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

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.List;

/\*\*

\* Created by anirudh on 20/10/14.

\*/

public class ReadExcelFileExample {

private static final String FILE\_PATH = "/Users/anirudh/Projects/JCGExamples/JavaWriteReadExcelFileExample/testReadStudents.xlsx";

public static void main(String args[]) {

List studentList = getStudentsListFromExcel();

System.out.println(studentList);

}

private static List getStudentsListFromExcel() {

List studentList = new ArrayList();

FileInputStream fis = null;

try {

fis = new FileInputStream(FILE\_PATH);

// Using XSSF for xlsx format, for xls use HSSF

Workbook workbook = new XSSFWorkbook(fis);

int numberOfSheets = workbook.getNumberOfSheets();

//looping over each workbook sheet

for (int i = 0; i < numberOfSheets; i++) {

Sheet sheet = workbook.getSheetAt(i);

Iterator rowIterator = sheet.iterator();

//iterating over each row

while (rowIterator.hasNext()) {

Student student = new Student();

Row row = rowIterator.next();

Iterator cellIterator = row.cellIterator();

//Iterating over each cell (column wise) in a particular row.

while (cellIterator.hasNext()) {

Cell cell = cellIterator.next();

//The Cell Containing String will is name.

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

student.setName(cell.getStringCellValue());

//The Cell Containing numeric value will contain marks

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

//Cell with index 1 contains marks in Maths

if (cell.getColumnIndex() == 1) {

student.setMaths(String.valueOf(cell.getNumericCellValue()));

}

//Cell with index 2 contains marks in Science

else if (cell.getColumnIndex() == 2) {

student.setScience(String.valueOf(cell.getNumericCellValue()));

}

//Cell with index 3 contains marks in English

else if (cell.getColumnIndex() == 3) {

student.setEnglish(String.valueOf(cell.getNumericCellValue()));

}

}

}

//end iterating a row, add all the elements of a row in list

studentList.add(student);

}

}

fis.close();

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

return studentList;

}

}

### **2. Write Excel File In Java**

Now, lets see how to write an excel file in Java, we will use the same Student Class for this.

WriteExcelFileExample.java

package com.jcg.example;

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

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

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

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

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by anirudh on 23/10/14.

\*/

public class WriteExcelFileExample {

private static final String FILE\_PATH = "/Users/anirudh/Projects/JCGExamples/JavaWriteReadExcelFileExample/testWriteStudents.xlsx";

//We are making use of a single instance to prevent multiple write access to same file.

private static final WriteExcelFileExample INSTANCE = new WriteExcelFileExample();

public static WriteExcelFileExample getInstance() {

return INSTANCE;

}

private WriteExcelFileExample() {

}

public static void main(String args[]){

List studentList = new ArrayList();

studentList.add(new Student("Magneto","90","100","80"));

studentList.add(new Student("Wolverine","60","60","90"));

studentList.add(new Student("ProfX","100","100","100"));

writeStudentsListToExcel(studentList);

}

public static void writeStudentsListToExcel(List studentList){

// Using XSSF for xlsx format, for xls use HSSF

Workbook workbook = new XSSFWorkbook();

Sheet studentsSheet = workbook.createSheet("Students");

int rowIndex = 0;

for(Student student : studentList){

Row row = studentsSheet.createRow(rowIndex++);

int cellIndex = 0;

//first place in row is name

row.createCell(cellIndex++).setCellValue(student.getName());

//second place in row is marks in maths

row.createCell(cellIndex++).setCellValue(student.getMaths());

//third place in row is marks in Science

row.createCell(cellIndex++).setCellValue(student.getScience());

//fourth place in row is marks in English

row.createCell(cellIndex++).setCellValue(student.getEnglish());

}

//write this workbook in excel file.

try {

FileOutputStream fos = new FileOutputStream(FILE\_PATH);

workbook.write(fos);

fos.close();

System.out.println(FILE\_PATH + " is successfully written");

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

}