Apache poi:

* In Java, reading an Excel file is not similar to reading a Word file because of cells(rows & columns) in an Excel file.
* JDK does not provide a direct API to read data from Excel files for which we have to use a third-party library that is Apache POI.
* Apache POI is an open-source java library designed for reading and writing Microsoft documents in order to create and manipulate various file formats based on Microsoft Office.
* Create, modify and display/read operations can be done using POI.

**Apache POI, JExcel, and Fastexcel APIs are used among this Apache POI is used widely.**

The name **POI**, or "**Poor Obfuscation Implementation**", refers to the fact that Microsoft file formats are deliberately obfuscated(to make something unclear or hard to understand, especially deliberately), but poorly, since they can be successfully reverse-engineered. This was discovered in the early 2000s when the Apache Foundation began reverse-engineering Microsoft file formats so that Java developers could work with them without getting into the details of file formats.

Apache POI is an API that provides a collection of Java libraries that allow users to read, write, and manipulate different Microsoft files, such as Excel, PowerPoint, and Word files.

POI can be used with any JVM based language that can import Java jar files, such as Scala, Groovy, and Kotlin.

The Apache POI library **supports both *.xls* and *.xlsx* files** and is a more complex library than other Java libraries for working with Excel files.

It provides the *Workbook* interface for modeling an *Excel* file, and the *Sheet*, *Row,* and *Cell* interfaces that model the elements of an Excel file, as well as implementations of each interface for both file formats.

When working with the newer *.xlsx* file format(after 2007), we’d use the *XSSFWorkbook*, *XSSFSheet*, *XSSFRow,* and *XSSFCell* classes*.*

*XSSF - XML SpreadSheet Format*

To work with the older *.xls* format, we use the *HSSFWorkbook*, *HSSFSheet*, *HSSFRow,* and *HSSFCell* classes*. HSSF - Horrible SpreadSheet Format*

When the cell type value is *STRING*, the content will be read using the *getRichStringCellValue()* method of the *Cell* interface

For *BOOLEAN* values, we have the *getBooleanCellValue()* method

And when the cell type is *FORMULA*, we can use the *getCellFormula()* method

## Workbook

This is the super-interface of all classes that create or maintain Excel workbooks. It belongs to the **org.apache.poi.ss.usermodel** package. The two classes that implement this interface are,

* **HSSFWorkbook** − This class has methods to read and write Microsoft Excel files in .xls format. It is compatible with MS-Office versions 97-2003.
* **XSSFWorkbook** − This class has methods to read and write Microsoft Excel and OpenOffice xml files in .xls or .xlsx format. It is compatible with MS-Office versions 2007 or later.

## XSSFWorkbook

It is a class that is used to represent both high and low level Excel file formats. It belongs to the **org.apache.xssf.usemodel** package and implements the **Workbook** interface. Listed below are the methods and constructors under this class.

### Class Constructors

| **XSSFWorkbook()**  Creates a new XSSFworkbook object from scratch. |
| --- |
| **XSSFWorkbook(java.io.File file)**  Constructs an XSSFWorkbook object from a given file. |
| **XSSFWorkbook(java.io.InputStream is)**  Constructs an XSSFWorkbook object, by buffering the whole input stream into memory and then opening an OPCPackage object for it. |
| **XSSFWorkbook(java.lang.String path)**  Constructs an XSSFWorkbook object given the full path of a file. |

### Class Methods

| **createSheet()**  Creates an XSSFSheet for this workbook, adds it to the sheets, and returns the high level representation. |
| --- |
| **createSheet(java.lang.String sheetname)**  Creates a new sheet for this Workbook and returns the high level representation. |
| **createFont()**  Creates a new font and adds it to the workbook's font table. |
| **createCellStyle()**  Creates a new XSSFCellStyle and adds it to the workbook's style table. |
| **createFont()**  Creates a new font and adds it to the workbook's font table. |
| **setPrintArea(int sheetIndex, int startColumn, int endColumn, int startRow,int endRow)**  Sets the print area of a given sheet as per the specified parameters.  For the remaining methods of this class, refer the complete API document at − [https://poi.apache.org/apidocs/org/apache/poi/xssf/usermodel/XSSFWorkbook.html.](https://poi.apache.org/apidocs/org/apache/poi/xssf/usermodel/XSSFWorkbook.html) |

## Sheet

Sheet is an interface under the **org.apache.poi.ss.usermodel** package and it is a super-interface of all classes that create high or low level spreadsheets with specific names. The most common type of spreadsheet is worksheet, which is represented as a grid of cells.

## HSSFSheet

This is a class under the **org.apache.poi.hssf.usermodel** package. It can create excel spreadsheets and it allows to format the sheet style and sheet data.

## XSSFSheet

This is a class which represents high level representation of excel spreadsheets. It is under **org.apache.poi.hssf.usermodel** package.

### Class Constructors

| **XSSFSheet()**  Creates new XSSFSheet − called by XSSFWorkbook to create a sheet from scratch. |
| --- |
| **XSSFSheet(PackagePart part, PackageRelationship rel)**  Creates an XSSFSheet representing the given package part and relationship. |

### Class Methods

| **addMergedRegion(CellRangeAddress region)**  Adds a merged region of cells (hence those cells form one). |
| --- |
| **autoSizeColumn(int column)**  Adjusts the column width to fit the contents. |
| **iterator()**  This method is an alias for rowIterator() to allow foreach loops |
| **addHyperlink(XSSFHyperlink hyperlink)**  Registers a hyperlink in the collection of hyperlinks on this sheet |

For the remaining methods of this class, refer the complete API at − <https://poi.apache.org/apidocs/org/apache/poi/xssf/usermodel/XSSFSheet.html>

## Row

This is an interface under the **org.apache.poi.ss.usermodel** package. It is used for high-level representation of a row of a spreadsheet. It is a super-interface of all classes that represent rows in POI library.

## XSSFRow

This is a class under the **org.apache.poi.xssf.usermodel** package. It implements the Row interface, therefore it can create rows in a spreadsheet. Listed below are the methods and constructors under this class.

### Class Methods

| **createCell(int columnIndex)**  Creates new cells within the row and returns it. |
| --- |
| **setHeight(short height)**  Sets the height in short units. |

## Cell

This is an interface under the **org.apache.poi.ss.usermodel** package. It is a super-interface of all classes that represent cells in the rows of a spreadsheet.

Cells can take various attributes such as blank, numeric, date, error, etc. Cells should have their own numbers (0 based) before being added to a row.

## XSSFCell

This is a class under the **org.apache.poi.xssf.usermodel** package. It implements the Cell interface. It is a high-level representation of cells in the rows of a spreadsheet.

### Class Methods

| **setCellStyle(CellStyle style)**  Sets the style for the cell. |
| --- |
| **setCellType(int cellType)**  Sets the type of cells (numeric, formula, or string). |
| **setCellValue(boolean value)**  Sets a boolean value for the cell. |
| **setCellValue(java.util.Calendar value)**  Sets a date value for the cell. |
| **setCellValue(double value)**  Sets a numeric value for the cell. |
| **setCellValue(java.lang.String str)**  Sets a string value for the cell. |
| **setHyperlink(Hyperlink hyperlink)**  Assigns a hyperlink to this cell. |

puisg-col puisg-col-4-of-12 puisg-col-4-of-16 puisg-col-4-of-20 puisg-col-4-of-24 puis-list-col-left

puisg-col puisg-col-4-of-12 puisg-col-8-of-16 puisg-col-12-of-20 puisg-col-12-of-24 puis-list-col-right"