==========

Topic: **Apache POI**

==========

1. Document prepared by: **Rajat Verma**
    a. https://www.linkedin.com/in/rajat-v-3b0685128/
    b. https://github.com/rajatt95
    c. https://rajatt95.github.io/

========== Testing - Apache POI ==========

Apache POI:

1. Links:
    a. https://poi.apache.org/
    b. https://mvnrepository.com/artifact/org.apache.poi/poi
2. Apache POI is a Java library that is used to handle Microsoft Office Documents. Apache POI is open source and can be used by JVM-based programming languages.
3. Apache POI is an API provided by Apache foundation which is a collection of different java libraries.
4. This library gives the facility to read, write and manipulate different Microsoft files such as excel sheets, power-point, and word files.
5. The Apache POI in Selenium is a widely used API for selenium data-driven testing.
6. It is a POI library written in Java that gives users an API for manipulating Microsoft documents like .xls and .xlsx.
7. The Apache POI in Selenium is a widely used API for selenium data-driven testing. It is a POI library written in Java that gives users an API for manipulating Microsoft documents like .xls and .xlsx.
8. Apache POI is an open-source java library to create and manipulate various file formats based on Microsoft Office. Using POI, one should be able to perform create, modify and display/read operations on the following file formats. For Example, Java doesn't provide built-in support for working with excel files, so we need to look for open-source APIs for the job.

Important features:

1. Apache POI provides stream-based processing, that is suitable for large files and requires less memory.
2. Apache POI is able to handle both XLS and XLSX formats of spreadsheets.
3. Apache POI contains HSSF implementation for Excel '97(-2007) file format i.e XLS.
4. Apache POI XSSF implementation should be used for Excel 2007 OOXML (.xlsx) file format.
5. Apache POI HSSF and XSSF API provide mechanisms to read, write or modify excel spreadsheets.
6. Apache POI also provides SXSSF API which is an extension of XSSF to work with very large excel sheets.
7. SXSSF API requires less memory and is suitable when working with very large spreadsheets and heap memory is limited.
8. There are two models to choose from – the event model and the user model. The event model requires less memory because the excel file is read in tokens and requires processing. The user model is more object-oriented and easy to use.
9. Apache POI provides excellent support for additional excel features such as working with Formulas, creating cell styles by filling colors and borders, fonts, headers and footers, data validations, images, hyperlinks, etc.

Commonly used components of Apache POI:

1. **HSSF (Horrible Spreadsheet Format):** It is used to read and write .xls format of MS-Excel files.
2. **XSSF (XML Spreadsheet Format):** It is used for the .xlsx file format of MS-Excel.
3. **POIFS (Poor Obfuscation Implementation File System):** This component is the basic factor of all other POI elements. It is used to read different files explicitly.
4. **HWPF (Horrible Word Processor Format):** It is used to read and write doc extension files of MS Word.
5. **HSLF (Horrible Slide Layout Format):** It is used for reading, creating, and editing PowerPoint presentations.

Test scenario: Facebook Login

Test cases:

1. Blank value
2. Invalid data
3. Invalid username and valid password
4. A valid username and invalid password
5. A valid username and Valid password

Read Excel file:

1. **Xls** vs **xlsx:**
2. Selenium supports only web applications.
3. So, for reading/write Excel files in Selenium, we have to take help from a third-party API like JXL and Apache POI.
4. **Apache POI** is an open-source API,
    a. Written in Java.
    b. Gives so much flexibility to read/write excel files.
    c. It has so many predefined methods
    d. It supports both Excel 2003 (.xls) and Excel 2007 (.xlsx) file formats
5. **JXL** does not support Excel 2007 (.xlsx) file format.
    a. It only supports Excel 2003 (.xls) file format

Working with Apache POI:

Classes and Interfaces used in POI

XLS classes (2003)	**Interface**	**XLSX classes (2007)**
HSSFWorkbook	Workbook	XSSFWorkbook
HSSFSheet	Sheet	XSSFSheet
HSSFRow	Row	XSSFRow
HSSFCell	Cell	XSSFCell

Methods used to deal with Excel file data are as below:

1. **getRow(rowNum).getCell(cellNum).getStringCellValue()** -> Print the data of any cell
2. **getLastRowNum()** -> Return Last row number of Table present in Excel sheet, it works on the basis of index
3. **getFirstRowNum()** -> Return first row number of Table present in Excel sheet, it works on the basis of index
4. **getRow(rowNum).getPhysicalNumOfCells()** -> Return Total number of columns in a row
5. **getPhysicalNumOfRows()** -> Return the total number of rows present in Excel sheet
6. **getRow(0).getLastCellNum()** -> Return last cell number in a 0th row, it does not work on the basis of indexing

---

---

1. **Code Snippets:**
    a. Read Rows
    b. Read Columns
    c. Print all Cells
    d. Write Data
    e. Read Data
    f. CopyData_XLS_To_NewSheet_AllData
    g. CopyData_XLSX_To_NewSheet_AllData

CODE - START

---

1. Read Rows:

```java
package _01_apachePOI;

import java.io.File;..

public class _01_Excel_Read_Rows {

    private static final String SubSheet_Name = "DemoSheet";
    private static final String Sheet_Location = ".//src//test//resources//apache_POI_config//Demo.xlsx";

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

        XSSFSheet sh = initializeExcelSheet();

        // Print the name of loaded sheet
        System.out.println("sh.getSheetName(): " + sh.getSheetName());
        System.out.println("----------------------------------");
        // Print Username from Excel sheet
        System.out.println(
                "sh.getRow(0).getCell(0).getStringCellValue(): " + sh.getRow(0).getCell(0).getStringCellValue());

        // Print P2 from Excel sheet
        System.out.println(
                "sh.getRow(2).getCell(1).getStringCellValue(): " + sh.getRow(2).getCell(1).getStringCellValue());
        System.out.println("----------------------------------");
        // Print the Total no. of Rows - 1st Way
        System.out.println("Total Rows - 1st way: " + sh.getPhysicalNumberOfRows());
        System.out.println("----------------------------------");
        // Print the Total no. of Rows - 2nd Way
        //// Here, index will work
        System.out.println("2nd way: ");
        System.out.println("  " + sh.getLastRowNum());// 5
        System.out.println("  " + sh.getFirstRowNum());// 0
        int rows = (sh.getLastRowNum() - sh.getFirstRowNum() + 1);
        System.out.println("Total rows: " + rows);
        System.out.println("----------------------------------");
        // Print the Total no. of Rows - 3rd Way
        System.out.println("3rd way: ");
        System.out.println(sh.getLastRowNum() + 1);

    }

    private static XSSFSheet initializeExcelSheet() throws FileNotFoundException, IOException {
        // Specify the location of Excel File
        File src = new File(Sheet_Location);

        // Load File
        FileInputStream fis = new FileInputStream(src);

        // Load Workbook
        XSSFWorkbook wb = new XSSFWorkbook(fis);

        // Load Worksheet
        XSSFSheet sh = wb.getSheet(SubSheet_Name);
        return sh;
    }

}
```

---

2. Read Column:

```java
package _01_apachePOI;

import java.io.File;

public class _02_Excel_Read_Column {

    private static final String SubSheet_Name = "DemoSheet";
    private static final String Sheet_Location = ".//src//test//resources//apache_POI_config//Demo.xlsx";

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

        XSSFSheet sh = initializeExcelSheet();

        // Print the name of loaded sheet
        System.out.println("sh.getSheetName(): " + sh.getSheetName());

        // Print the Total no. of Column - 1st Way
        System.out.println("1st way: ");
        System.out.println(sh.getRow(0).getPhysicalNumberOfCells());// 2

        // Print the Total no. of Column - 2nd Way
        System.out.println("2nd way: ");
        System.out.println(sh.getRow(0).getLastCellNum());// 2

        // Print the Total no. of Column - 3rd Way
        System.out.println("3rd way: ");
        int columns = sh.getRow(0).getLastCellNum();
        System.out.println("Total Columns: " + columns);// 2

    }

    private static XSSFSheet initializeExcelSheet() throws FileNotFoundException, IOException {
        // Specify the location of Excel File
        File src = new File(Sheet_Location);

        // Load File
        FileInputStream fis = new FileInputStream(src);

        // Load Workbook
        XSSFWorkbook wb = new XSSFWorkbook(fis);

        // Load Worksheet
        XSSFSheet sh = wb.getSheet(SubSheet_Name);
        return sh;
    }

}
```

---

3. Print all cells

```java
package _01_apachePOI;

import java.io.File;

public class _03_Excel_Read_Print_All_Cells {

    private static final String SubSheet_Name = "DemoSheet";
    private static final String Sheet_Location = ".//src//test//resources//apache_POI_config//Demo.xlsx";

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

        XSSFSheet sh = initializeExcelSheet();

        // Print the name of loaded sheet
        System.out.println("sh.getSheetName(): " + sh.getSheetName());

//      int rows = (sh.getLastRowNum() - sh.getFirstRowNum() + 1);
//      int columns = sh.getRow(0).getLastCellNum();

        int rows = sh.getLastRowNum();
        int columns = sh.getRow(0).getLastCellNum();

        // Print all Cells of Excel sheet
        for (int i = 0; i <= rows; i++) {
            for (int j = 0; j < columns; j++) {
                System.out.print(sh.getRow(i).getCell(j).getStringCellValue()+", ");
            }
            System.out.println();
            System.out.println("------------");
        }

    }

    private static XSSFSheet initializeExcelSheet() throws FileNotFoundException, IOException {
        // Specify the location of Excel File
        File src = new File(Sheet_Location);

        // Load File
        FileInputStream fis = new FileInputStream(src);

        // Load Workbook
        XSSFWorkbook wb = new XSSFWorkbook(fis);

        // Load Worksheet
        XSSFSheet sh = wb.getSheet(SubSheet_Name);
        return sh;
    }
}
```

---

4. Write Data

```java
package _01_apachePOI;

import java.io.File;

Run All
public class _11_HW_WriteData {

    private static final String Sheet_Location = ".//src//test//resources//apache_POI_config//HW_Write.xlsx";
    private static final String SubSheet_Name = "Employee Data";

    // any exceptions need to be caught
    @Test
    Run | Debug
    public void TC_Write_Data_In_Excel() throws Exception {
        // workbook object
        XSSFWorkbook workbook = new XSSFWorkbook();

        // spreadsheet object
        XSSFSheet spreadsheet = workbook.createSheet(SubSheet_Name);

        // creating a row object
        XSSFRow row;
        // This data needs to be written (Object[])
        Map<String, Object[]> studentData = new TreeMap<String, Object[]>();

        studentData.put("1", new Object[] { "Employee Id", "NAME", "POST" });
        studentData.put("2", new Object[] { "1301", "Prajjawal", "Intern" });
        studentData.put("3", new Object[] { "1302", "Nitish", "Software Consultant" });
        studentData.put("4", new Object[] { "1303", "Aditi", "QA Engineer" });
        studentData.put("5", new Object[] { "1303", "Ayush", "System Engineer" });
        studentData.put("6", new Object[] { "1304", "Abhishek", "Intern" });

        Set<String> keyid = studentData.keySet();

        int rowid = 0;
        // writing the data into the sheets...
        for (String key : keyid) {
            row = spreadsheet.createRow(rowid++);
            Object[] objectArr = studentData.get(key);
            int cellid = 0;

            for (Object obj : objectArr) {
                Cell cell = row.createCell(cellid++);
                cell.setCellValue((String) obj);
                System.out.println(cellid + " " + cell);
            }
        }
        // .xlsx is the format for Excel Sheets...
        // writing the workbook into the file...
        FileOutputStream out = new FileOutputStream(new File(Sheet_Location));
        workbook.write(out);

        out.close();
    }
}
```

---

5. Read Data

```java
package _01_apachePOI;

import java.io.FileInputStream;

Run All
public class _12_HW_ReadData {

    // https://blog.knoldus.com/read-and-write-data-from-excel-sheet-using-apache-poi/

    private static final String SubSheet_Name = "Employee Data";
    private static final String Sheet_Location = ".//src//test//resources//apache_POI_config//HW_Write.xlsx";

    // Any exceptions need to be caught
    @Test
    Run | Debug
    public void TC_Read_Data_In_Excel() throws EncryptedDocumentException, IOException {

        // Get the excel sheet file location
        FileInputStream fis = new FileInputStream(Sheet_Location);

        XSSFWorkbook workbook = new XSSFWorkbook(fis);

        // XSSFSheet sheet = workbook.getSheetAt(0);
        XSSFSheet sheet = workbook.getSheet(SubSheet_Name);

        // Get the total row count in the excel sheet
        Iterator<Row> iterator = sheet.rowIterator();

        // int cellIndex = 0;
        // String description = null;
        while (iterator.hasNext()) {
            Row row = iterator.next();
            for (int i = 0; i < row.getPhysicalNumberOfCells(); i++) {
                Cell cell = row.getCell(i);
                // print the cell value
                // System.out.print(i + " " + cell);
                System.out.print(cell + "    -     ");
            }
            System.out.println();
        }
    }

}
```

---

6. CopyData_XLS_To_NewSheet_AllData

```java
package _01_apachePOI;

import java.io.File;

public class _14_HW_CopyData_XLS_To_NewSheet_AllData {

    private static final String SubSheet_Name = "MyLinks";
    private static final String SubSheet_New_Name = "MyLinks_2";

    private static final String XLS_FILE_PATH = ".//src//test//resources//apache_POI_config//Sample_Sheet.xls";

    // https://www.quora.com/How-can-I-copy-several-rows-from-sheet-1-to-another-sheet-2-of-the-same-workbook-using-JAVA-in-Selenium

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

        FileInputStream file = new FileInputStream(XLS_FILE_PATH);

        HSSFWorkbook workbook = new HSSFWorkbook(file);

        // Getting sheet1
        HSSFSheet sheet = workbook.getSheet(SubSheet_Name);

        // Creating sheet2
        Sheet sheetTwo = workbook.createSheet(SubSheet_New_Name);

        Row row;
        int rowCount_Sheet1 = (sheet.getLastRowNum() + 1);

        for (int j = 0; j < rowCount_Sheet1; j++) {
            // Getting row at index 0 in sheet1
            row = sheet.getRow(j);
            int rowLength = row.getPhysicalNumberOfCells();

            // Creating row at index 0 in sheet2
            Row sheetTwoRow = sheetTwo.createRow(j);

            // Setting value in row of sheet2 from sheet1
            for (int i = 0; i < rowLength; i++) {
                Cell cell = sheetTwoRow.createCell(i);
                Cell firstSheetCell = row.getCell(i);
                cell.setCellValue(firstSheetCell.getStringCellValue());
            }

        }

        writeChangesInExcelSheet(file, workbook);

        System.out.println("Success");
    }

    private static void writeChangesInExcelSheet(FileInputStream file, HSSFWorkbook workbook)
            throws IOException, FileNotFoundException {
        // Writing changes in Excel file
        file.close();
        FileOutputStream outFile = new FileOutputStream(new File(XLS_FILE_PATH));
        workbook.write(outFile);
        outFile.close();
    }

}
```

---

7. CopyData_XLSX_To_NewSheet_AllData

```java
package _01_apachePOI;

import java.io.File;

public class _14_HW_CopyData_XLSX_To_NewSheet_AllData {

    private static final String SubSheet_Name = "Employee Data";
    private static final String SubSheet_New_Name = "Employee Data_4";
    private static final String XLS_FILE_PATH = ".//src//test//resources//apache_POI_config//HW_Write.xlsx";

    // https://www.quora.com/How-can-I-copy-several-rows-from-sheet-1-to-another-sheet-2-of-the-same-workbook-using-JAVA-in-Selenium

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

        FileInputStream file = new FileInputStream(XLS_FILE_PATH);

        XSSFWorkbook workbook = new XSSFWorkbook(file);

        // Getting sheet1
        XSSFSheet sheet = workbook.getSheet(SubSheet_Name);

        // Creating sheet2
        Sheet sheetTwo = workbook.createSheet(SubSheet_New_Name);

        Row row;
        int rowCount_Sheet1 = (sheet.getLastRowNum() + 1);

        for (int j = 0; j < rowCount_Sheet1; j++) {
            // Getting row at index 0 in sheet1
            row = sheet.getRow(j);
            int rowLength = row.getPhysicalNumberOfCells();

            // Creating row at index 0 in sheet2
            Row sheetTwoRow = sheetTwo.createRow(j);

            // Setting value in row of sheet2 from sheet1
            for (int i = 0; i < rowLength; i++) {
                Cell cell = sheetTwoRow.createCell(i);
                Cell firstSheetCell = row.getCell(i);
                cell.setCellValue(firstSheetCell.getStringCellValue());
            }

        }

        writeChangesInExcelSheet(file, workbook);

        System.out.println("Success");
    }

    private static void writeChangesInExcelSheet(FileInputStream file, XSSFWorkbook workbook)
            throws IOException, FileNotFoundException {
        // Writing changes in Excel file
        file.close();
        FileOutputStream outFile = new FileOutputStream(new File(XLS_FILE_PATH));
        workbook.write(outFile);
        outFile.close();
    }

}
```

CODE - END

==========

1. **To connect:**
    a. https://www.linkedin.com/in/rajat-v-3b0685128/
    b. https://github.com/rajatt95
    c. https://rajatt95.github.io/

==========