Apache Poi - Demo Codes

1. Install Apache Poi:

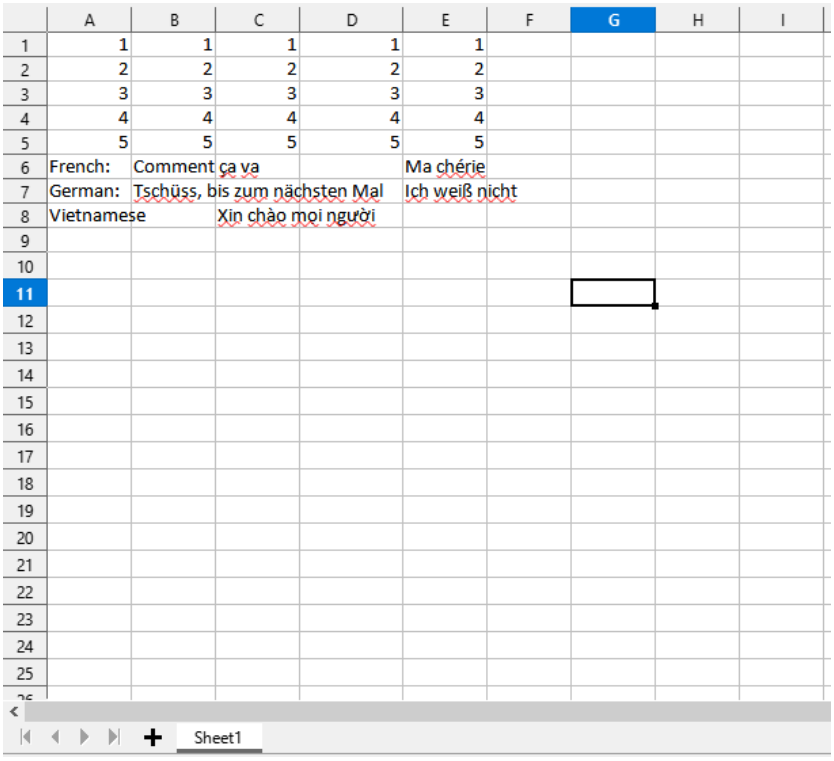
After creating a new Maven project in IntelliJ, we add these lines of code into file **pom.xml**, to use Apache Poi version 5.0.0:

```
<dependencies>
    <!-- https://mvnrepository.com/artifact/org.apache.poi/poi -->
    <dependency>
        <groupId>org.apache.poi</groupId>
        <artifactId>poi</artifactId>
        <version>5.0.0</version>
    </dependency>

    <dependency>
        <groupId>org.apache.poi</groupId>
        <artifactId>poi-ooxml</artifactId>
        <version>5.0.0</version>
    </dependency>
</dependencies>
```

2. Read and Write Excel file:

- **Read from .xlsx file:**

There's a file called **input_xlsx.xlsx** (inside folder **inputFiles**):



Then running this code in IntelliJ (**ReadXLSX.java**):

```java
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.Iterator;

import org.apache.poi.poifs.crypt.EncryptionInfo;
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

/**
 * A dirty simple program that reads an Excel file.
 * @author www.codejava.net
 *
 */

public class ReadXLSX {
    public static void main(String[] args) throws IOException {
        String excelFilePath = "C:\\Users\\ntmanh\\Desktop\\input_xlsx.xlsx";
        FileInputStream inputStream = new FileInputStream(new File(excelFilePath));
        Workbook workbook = new XSSFWorkbook(inputStream);
        Sheet firstSheet = workbook.getSheetAt( i: 0);
        Iterator<Row> iterator = firstSheet.iterator();

        while (iterator.hasNext()) {
            Row nextRow = iterator.next();
            Iterator<Cell> cellIterator = nextRow.cellIterator();

            while (cellIterator.hasNext()) {
                Cell cell = cellIterator.next();

                if (cell.getCellType() == CellType.STRING) {
                    System.out.print(cell.getStringCellValue());
                }
                if (cell.getCellType() == CellType.BOOLEAN) {
                    System.out.print(cell.getBooleanCellValue());
                }
                if (cell.getCellType() == CellType.NUMERIC) {
                    System.out.print(cell.getNumericCellValue());
                }

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

        workbook.close();
        inputStream.close();
    }
}
```

This is the result we will get in console window:

```
"C:\Program Files\Eclipse Adoptium\jdk-11.0.13.8-hotspot\bin\java.exe" ...
1.0   1.0   1.0   1.0   1.0
2.0   2.0   2.0   2.0   2.0
3.0   3.0   3.0   3.0   3.0
4.0   4.0   4.0   4.0   4.0
5.0   5.0   5.0   5.0   5.0
French:   Comment ça va   Ma chérie
German:   Tschüss, bis zum nächsten Mal    Ich weiß nicht
Vietnamese   Xin chào mọi người


Process finished with exit code 0
```

- **Write to .xlsx file:**

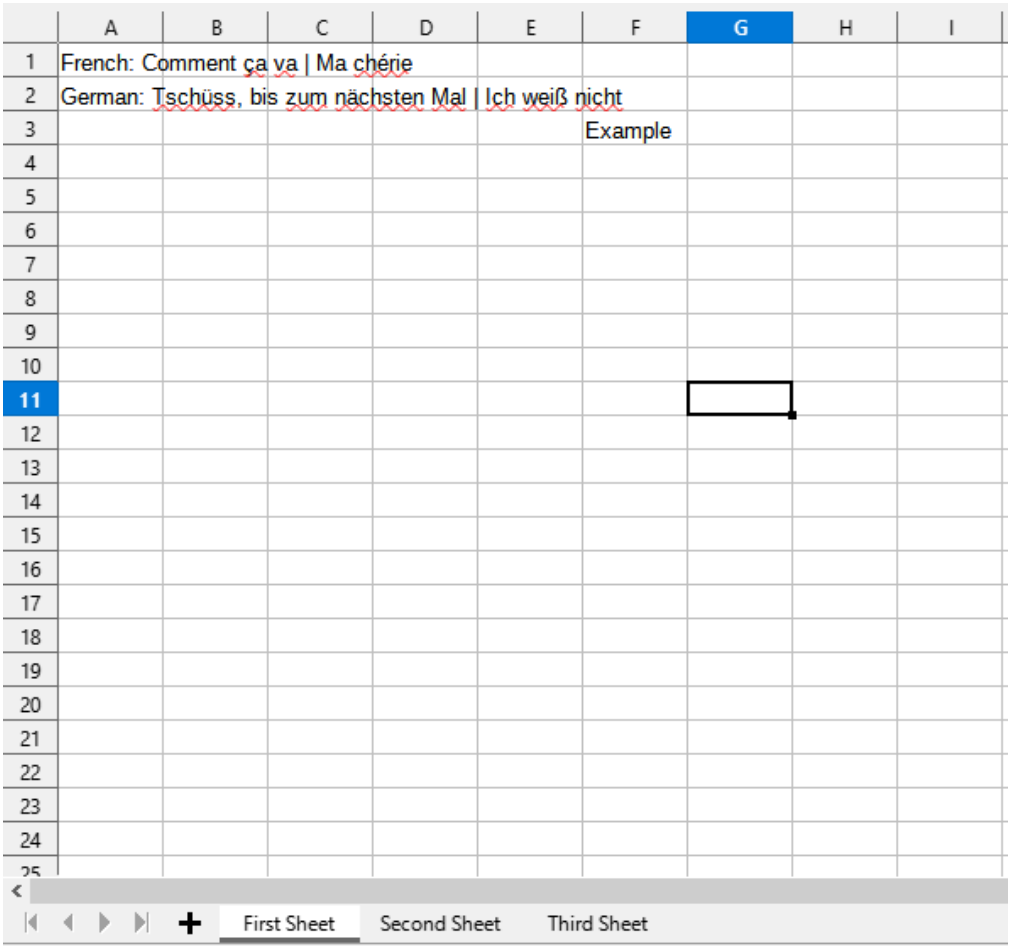
Run this code in IntelliJ (**WriteXLSX.java**):

```java
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
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.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;

public class WriteXLSX {
    public static void main(String[] args) throws FileNotFoundException, IOException {
        //Write .xlsx file
        Workbook wookbook = new XSSFWorkbook();
        try(OutputStream fileOut = new FileOutputStream( name: "C:\\Users\\ntmanh\\Desktop\\output_xlsx.xlsx")) {
            Sheet sheet1 = wookbook.createSheet( s: "First Sheet");
            Sheet sheet2 = wookbook.createSheet( s: "Second Sheet");
            Sheet sheet3 = wookbook.createSheet( s: "Third Sheet");
            Row row     = sheet1.createRow( i: 2);
            Cell cell   = row.createCell( i: 5);
            cell.setCellValue("Example");

            row = sheet1.createRow( i: 0);
            cell   = row.createCell( i: 0);
```

```
            cell.setCellValue("French: Comment ça va | Ma chérie");

            row = sheet1.createRow( i: 1);
            cell   = row.createCell( i: 0);
            cell.setCellValue("German: Tschüss, bis zum nächsten Mal | Ich weiß nicht");
            wookbook.write(fileOut);
        } catch(Exception e) {
            System.out.println(e.getMessage());
        }

    }
}
```

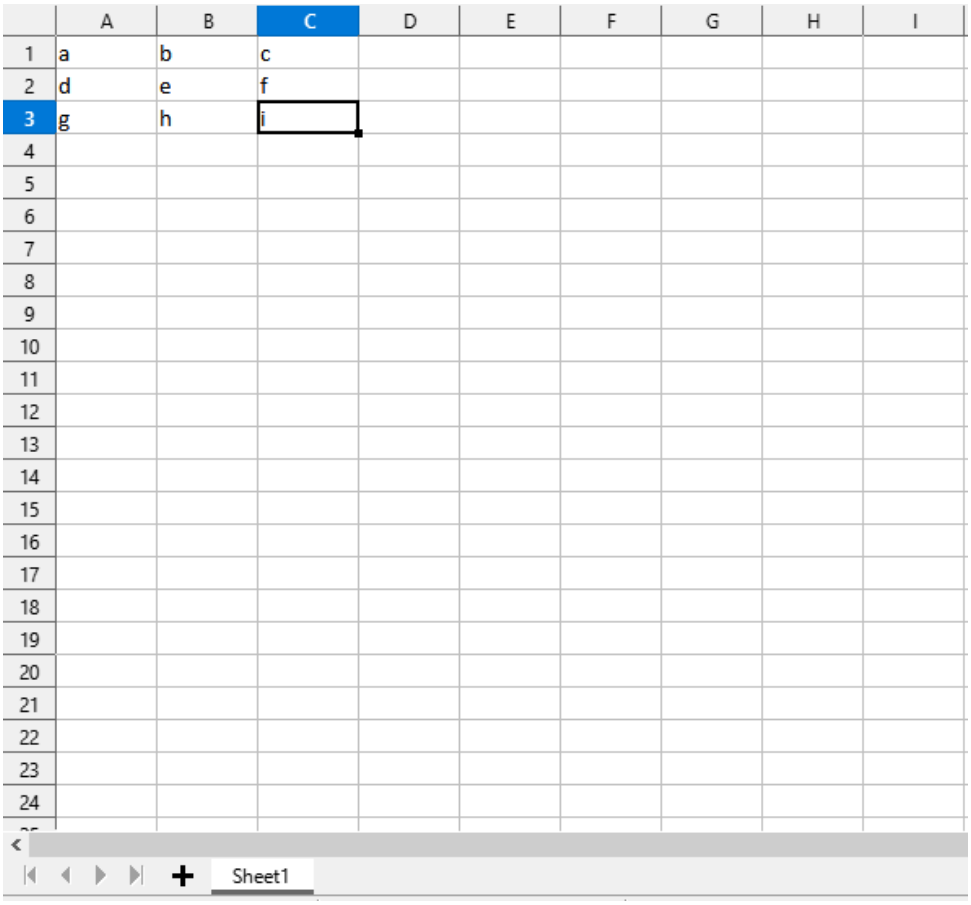
```
"C:\Program Files\Eclipse Adoptium\jdk-11.0.13.8-hotspot\bin\java.exe" ...

Process finished with exit code 0
```

Open **output_xls.xls**:



3. Read .xlsx file protected with password:

There's a file called **check_password.xlsx** (inside folder **inputFiles**), with password to open is **abc123**



Then running this code in IntelliJ (**ReadXLSXPassword.java**):

```java
import java.io.File;
import java.io.IOException;
import java.io.InputStream;
import java.security.GeneralSecurityException;
import java.util.Iterator;

import org.apache.poi.poifs.crypt.Decryptor;
import org.apache.poi.poifs.crypt.EncryptionInfo;
import org.apache.poi.poifs.filesystem.POIFSFileSystem;
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.xssf.usermodel.XSSFWorkbook;

public class ReadXLSXPassword {
    public static void main(String[] args) {

        // Creating a xlsx file object with specific file path to read
        File xlsxFile = new File( pathname: "C:\\Users\\ntmanh\\Desktop\\check_password.xlsx");

        try {
            POIFSFileSystem fs = new POIFSFileSystem(xlsxFile);
            EncryptionInfo info = new EncryptionInfo(fs);
            Decryptor decryptor = Decryptor.getInstance(info);

            //Verifying the password
            if (!decryptor.verifyPassword( s: "abc123")) {
                throw new RuntimeException("Incorrect password: Unable to process");
            }

            InputStream dataStream = decryptor.getDataStream(fs);

            // Now parse dataStream

            XSSFWorkbook workbook = new XSSFWorkbook(dataStream);
            // Reading the first sheet of the excel file
            Sheet sheet = workbook.getSheetAt( index: 0);

            Iterator<Row> iterator = sheet.iterator();

            // Iterating all the rows
            while (iterator.hasNext()) {
                Row nextRow = iterator.next();
                Iterator<Cell> cellIterator = nextRow.cellIterator();

                // Iterating all the columns in a row
                while (cellIterator.hasNext()) {
                    Cell cell = cellIterator.next();
                    switch (cell.getCellType()) {
                        case STRING:
                            System.out.print(cell.getStringCellValue());
                            break;
```

```
                        case BOOLEAN:
                            System.out.print(cell.getBooleanCellValue());
                            break;
                        case NUMERIC:
                            System.out.print(cell.getNumericCellValue());
                            break;
                        default:
                            break;
                    }
                    System.out.print("  ");
                }
                System.out.println();
            }
            workbook.close();
        } catch (GeneralSecurityException | IOException ex) {
            throw new RuntimeException("Unable to process encrypted document", ex);
        }
    }
}
```

This is the result we will get in console window:

```
"C:\Program Files\Eclipse Adoptium\jdk-11.0.13.8-hotspot\bin\java.exe" ...
a  b  c
d  e  f
g  h  i

Process finished with exit code 0
```

If we enter an incorrect password, such as **abcd1234**:

```
            if (!decryptor.verifyPassword( s: "abc124")) {
```

The console window will throw an **Incorrect password** exception:

```
"C:\Program Files\Eclipse Adoptium\jdk-11.0.13.8-hotspot\bin\java.exe" ...
Exception in thread "main" java.lang.RuntimeException Create breakpoint : Incorrect password: Unable to process
    at ReadXLSXPassword.main(ReadXLSXPassword.java:28)

Process finished with exit code 1
```

Reference:
https://howtodoinjava.com/java/library/readingwriting-excel-files-in-java-poi-tutorial/