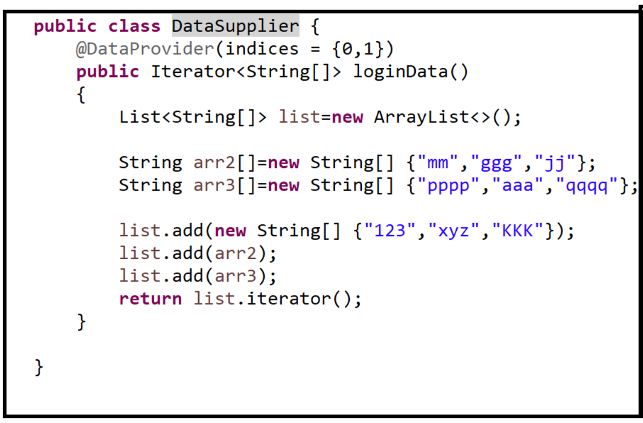
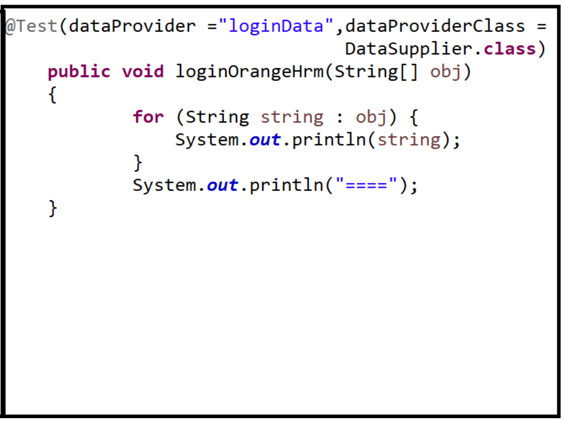
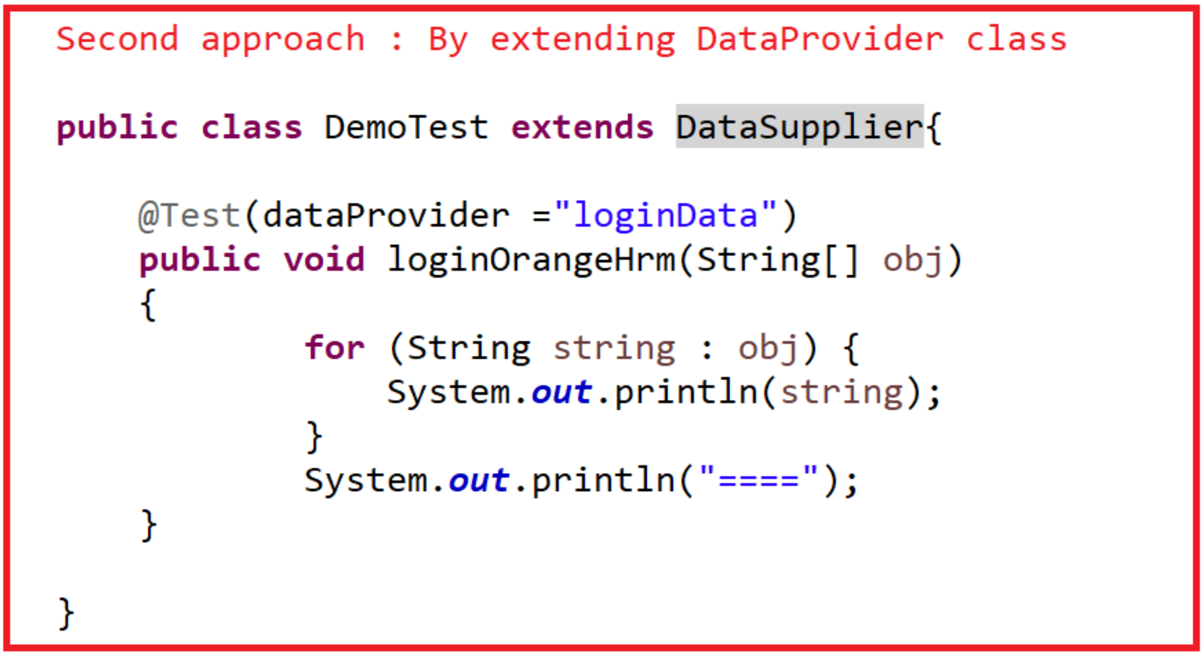
```
How to Create DataProvider methods in Seperate Class?
-> Create a new class
   Define a method and annotate this methods with @DataProvider


-> To use dataprovider method in Test class we have two approach
   1) By extending DataProvider class in Test Class
   2) By using dataProviderClass attribute of @Test annotatio as below
```



```java
public class DataSupplier {
    @DataProvider(indices = {0,1})
    public Iterator<String[]> loginData()
    {
        List<String[]> list=new ArrayList<>();

        String arr2[]=new String[] {"mm","ggg","jj"};
        String arr3[]=new String[] {"pppp","aaa","qqqq"};

        list.add(new String[] {"123","xyz","KKK"});
        list.add(arr2);
        list.add(arr3);
        return list.iterator();
    }

}
```



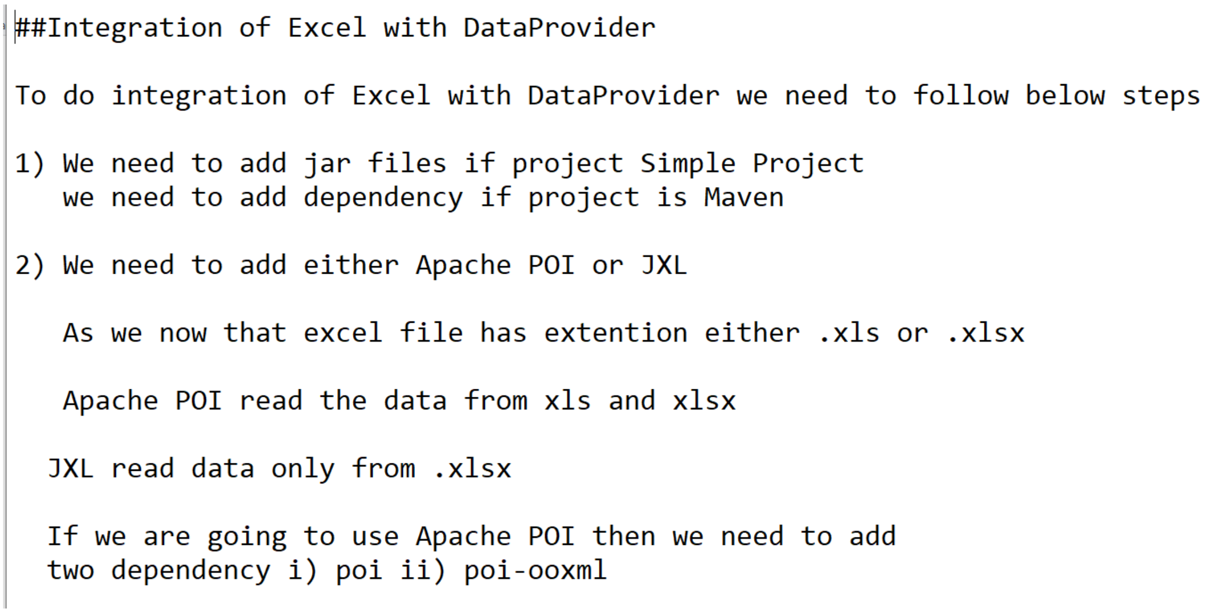
```java
@Test(dataProvider ="loginData",dataProviderClass =
                                        DataSupplier.class)
    public void loginOrangeHrm(String[] obj)
    {
            for (String string : obj) {
                System.out.println(string);
            }
            System.out.println("====");
    }
```



Second approach : By extending DataProvider class

```java
public class DemoTest extends DataSupplier{

    @Test(dataProvider ="loginData")
    public void loginOrangeHrm(String[] obj)
    {
            for (String string : obj) {
                System.out.println(string);
            }
            System.out.println("====");
    }

}
```

```
Can we seperate dataProvider from Test class?
Ans:-> yes

Q. Can we store multiple dataprovider in dataProvider class?
Ans -> yes

Q. Can we have multiple dataprovider classes in the Project?
Ans:-> Yes

-----------------------------------------------------------------
```



##Integration of Excel with DataProvider

To do integration of Excel with DataProvider we need to follow below steps

1) We need to add jar files if project Simple Project
   we need to add dependency if project is Maven

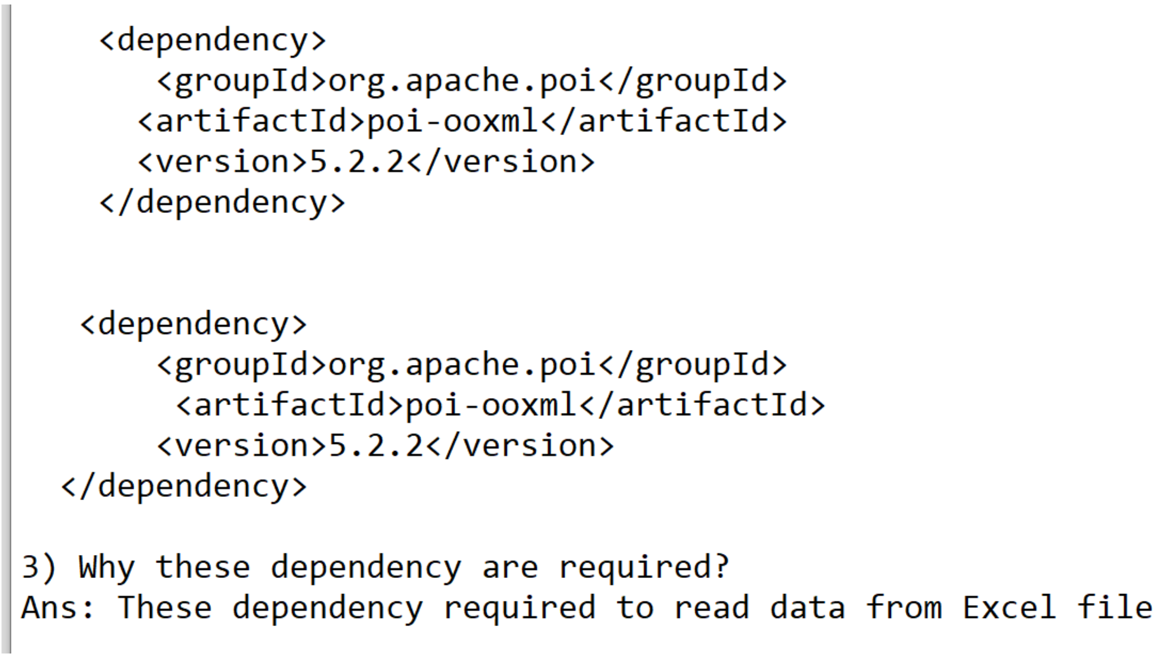
2) We need to add either Apache POI or JXL

   As we now that excel file has extention either .xls or .xlsx

   Apache POI read the data from xls and xlsx

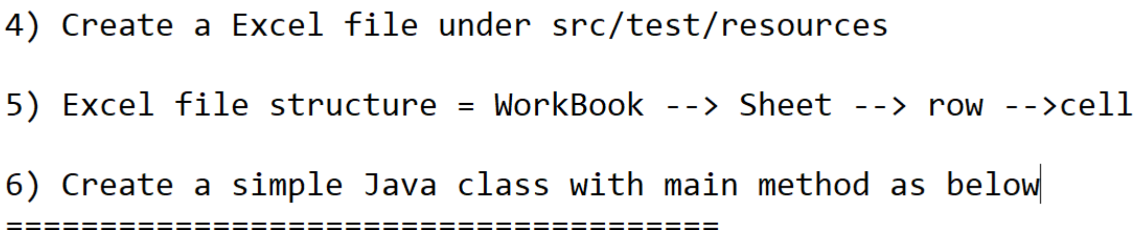
  JXL read data only from .xlsx

  If we are going to use Apache POI then we need to add
  two dependency i) poi ii) poi-ooxml



```
    <dependency>
       <groupId>org.apache.poi</groupId>
      <artifactId>poi-ooxml</artifactId>
      <version>5.2.2</version>
    </dependency>


   <dependency>
       <groupId>org.apache.poi</groupId>
        <artifactId>poi-ooxml</artifactId>
       <version>5.2.2</version>
  </dependency>
```

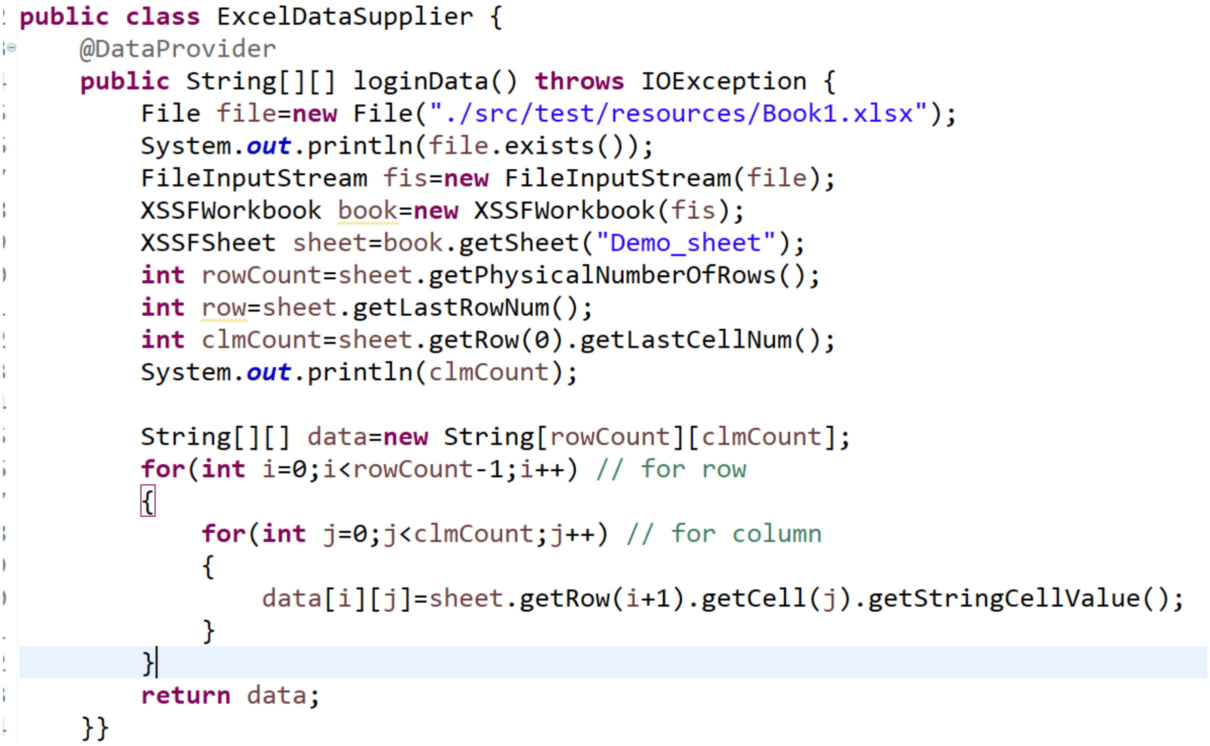
3) Why these dependency are required?
Ans: These dependency required to read data from Excel file



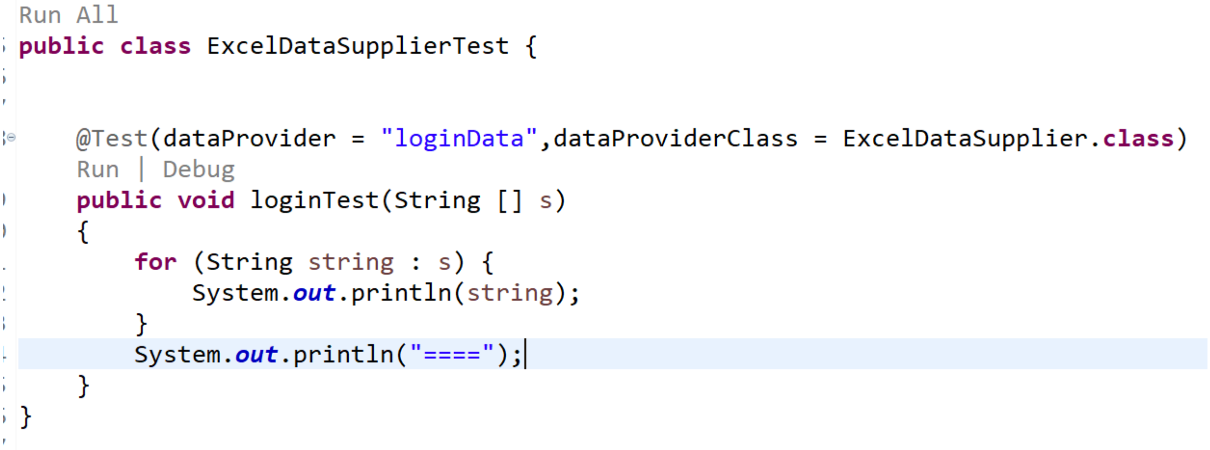
4) Create a Excel file under src/test/resources

5) Excel file structure = WorkBook --> Sheet --> row -->cell

6) Create a simple Java class with main method as below
=====================================



```
public class ExcelDataSupplier {
    @DataProvider
    public String[][] loginData() throws IOException {
        File file=new File("./src/test/resources/Book1.xlsx");
        System.out.println(file.exists());
        FileInputStream fis=new FileInputStream(file);
        XSSFWorkbook book=new XSSFWorkbook(fis);
        XSSFSheet sheet=book.getSheet("Demo_sheet");
        int rowCount=sheet.getPhysicalNumberOfRows();
        int row=sheet.getLastRowNum();
        int clmCount=sheet.getRow(0).getLastCellNum();
        System.out.println(clmCount);

        String[][] data=new String[rowCount][clmCount];
        for(int i=0;i<rowCount-1;i++) // for row
        {
            for(int j=0;j<clmCount;j++) // for column
            {
                data[i][j]=sheet.getRow(i+1).getCell(j).getStringCellValue();
            }
        }
        return data;
    }}
```



```
Run All
public class ExcelDataSupplierTest {


    @Test(dataProvider = "loginData",dataProviderClass = ExcelDataSupplier.class)
    Run | Debug
    public void loginTest(String [] s)
    {
        for (String string : s) {
            System.out.println(string);
        }
        System.out.println("====");
    }
}
```