**Exercise 1: Implementing the Singleton Pattern**

**Scenario:**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **SingletonPatternExample**.
2. **Define a Singleton Class:**
   * Create a class named Logger that has a private static instance of itself.
   * Ensure the constructor of Logger is private.
   * Provide a public static method to get the instance of the Logger class.
3. **Implement the Singleton Pattern:**
   * Write code to ensure that the Logger class follows the Singleton design pattern.
4. **Test the Singleton Implementation:**
   * Create a test class to verify that only one instance of Logger is created and used across the application.

SOLUTION:

**Logger.java**

**package** SingletonProblem;

**public** **class** Logger {

**private** **static** Logger *instance*;

**private** Logger() {

System.***out***.println("Logger instance created");

}

**public** **static** Logger getInstance() {

**if** (*instance* == **null**) {

*instance* = **new** Logger();

}

**return** *instance*;

}

**public** **void** log(String message) {

System.***out***.println("Log: " + message);

}

}

**TestClass.java**

**package** SingletonProblem;

**public** **class** TestClass {

**public** **static** **void** main(String[] args) {

Logger logger1 = Logger.*getInstance*();

Logger logger2 = Logger.*getInstance*();

logger1.log("First log");

logger2.log("Second log");

**if** (logger1 == logger2) {

System.***out***.println("Only one instance of logger created and used.");

} **else** {

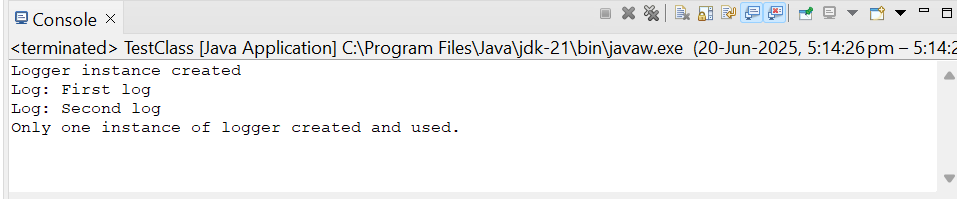
System.***out***.println("Different instances used.");

}

}

}

**OUTPUT:**

****

**Exercise 2: Implementing the Factory Method Pattern**

**Scenario:**

You are developing a document management system that needs to create different types of documents (e.g., Word, PDF, Excel). Use the Factory Method Pattern to achieve this.

**Steps:**

1. **Create a New Java Project:**
   1. Create a new Java project named FactoryMethodPatternExample.
2. **Define Document Classes:**
   1. Create interfaces or abstract classes for different document types such as WordDocument, PdfDocument, and ExcelDocument.
3. **Create Concrete Document Classes:**
   1. Implement concrete classes for each document type that implements or extends the above interfaces or abstract classes.
4. **Implement the Factory Method:**
   1. Create an abstract class DocumentFactory with a method createDocument().
   2. Create concrete factory classes for each document type that extends DocumentFactory and implements the createDocument() method.
5. **Test the Factory Method Implementation:**
   1. Create a test class to demonstrate the creation of different document types using the factory method.

SOLUTION:

**Doc.java**

**package** factorymethod;

**public** **interface** Doc {

**void** create();

}

**DocFac.java**

**package** factorymethod;

**public** **abstract** **class** DocFac {

**public** **abstract** Doc createDoc();

}

**Excel.java**

**package** factorymethod;

**import** java.awt.Desktop;

**import** java.io.File;

**import** java.io.FileWriter;

**import** java.io.IOException;

**public** **class** Excel **implements** Doc {

**public** **void** create() {

**try** {

File file = **new** File("C:\\Users\\anush\\OneDrive\\Desktop\\Doc\\sample.xlsx");

FileWriter writer = **new** FileWriter(file);

writer.write("This is an Excel file created by Factory Method.");

writer.close();

System.***out***.println("Excel file created.");

Desktop.*getDesktop*().open(file);

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**Word.java**

**package** factorymethod;

**import** java.awt.Desktop;

**import** java.io.File;

**import** java.io.FileWriter;

**import** java.io.IOException;

**public** **class** Word **implements** Doc {

**public** **void** create() {

**try** {

File file = **new** File("C:\\Users\\anush\\OneDrive\\Desktop\\Doc\\sample.docx");

FileWriter writer = **new** FileWriter(file);

writer.write("This is a Word document created by Factory Method.");

writer.close();

System.***out***.println("Word file created.");

Desktop.*getDesktop*().open(file);

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**PDF.java**

**package** factorymethod;

**import** java.awt.Desktop;

**import** java.io.File;

**import** java.io.FileWriter;

**import** java.io.IOException;

**public** **class** PDF **implements** Doc {

**public** **void** create() {

**try** {

File file = **new** File("C:\\Users\\anush\\OneDrive\\Desktop\\Doc\\sample.pdf");

FileWriter writer = **new** FileWriter(file);

writer.write("This is a PDF file created by Factory Method.");

writer.close();

System.***out***.println("PDF file created.");

Desktop.*getDesktop*().open(file);

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**ExcelDF.java**

**package** factorymethod;

**public** **class** ExcelDF **extends** DocFac {

**public** Doc createDoc() {

**return** **new** Excel();

}

}

**WordDF.java**

**package** factorymethod;

**public** **class** WordDF **extends** DocFac {

**public** Doc createDoc() {

**return** **new** Word();

}

}

**PDFDF.java**

**package** factorymethod;

**public** **class** PDFDF **extends** DocFac {

**public** Doc createDoc() {

**return** **new** PDF();

}

}

**Test.java**

**package** factorymethod;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

DocFac wordFactory = **new** WordDF();

Doc wordDoc = wordFactory.createDoc();

wordDoc.create();

DocFac pdfFactory = **new** PDFDF();

Doc pdfDoc = pdfFactory.createDoc();

pdfDoc.create();

DocFac excelFactory = **new** ExcelDF();

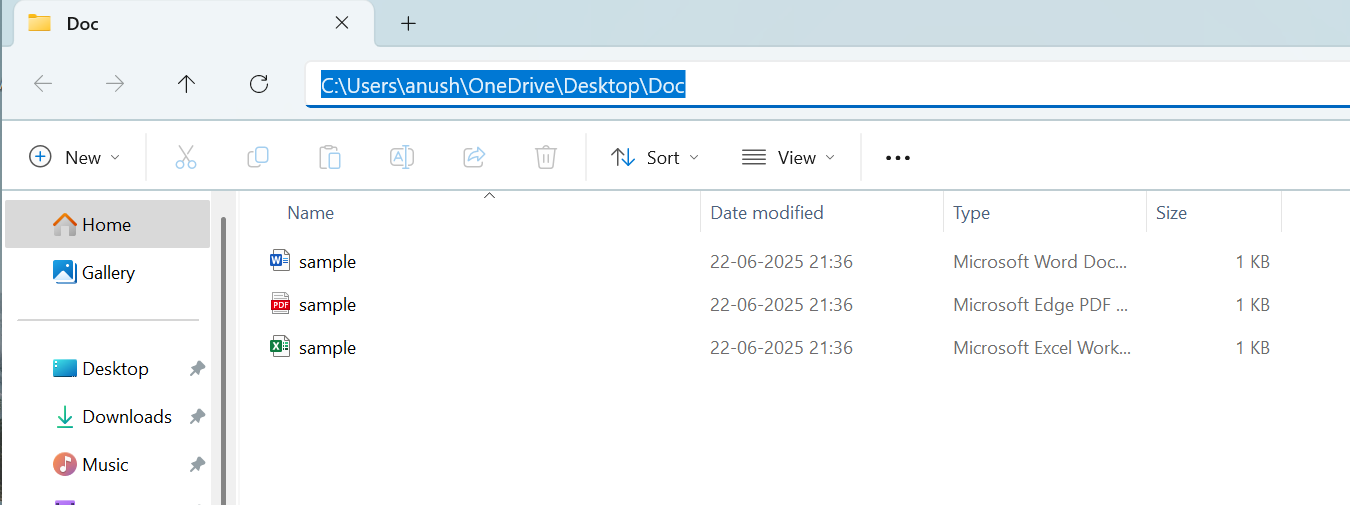
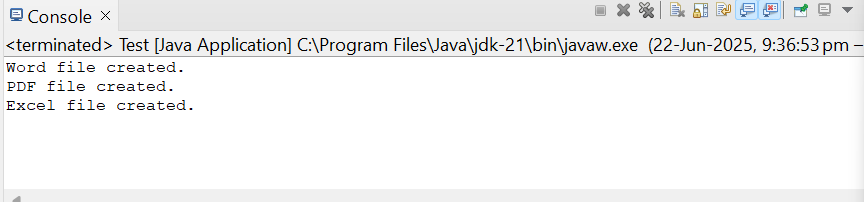
Doc excelDoc = excelFactory.createDoc();

excelDoc.create();

}

}

**OUTPUT:**

****