**R Tejashree (Superset ID : 6419704)**

WEEK 1

Design Patterns: Step-by-Step Guide

# Module 1: Factory Method Pattern

## Overview

The Factory Method Pattern is a creational design pattern that provides an interface for creating objects in a superclass, but allows subclasses to alter the type of objects that will be created.

## Folder Structure

- Documents/  
 - Document.java (interface or abstract class)  
 - PdfDocument.java, WordDocument.java, ExcelDocument.java (concrete implementations)  
- Factories/  
 - DocumentFactory.java (abstract factory)  
 - PdfDocumentFactory.java, WordDocumentFactory.java, ExcelDocumentFactory.java (concrete factories)  
- Test/  
 - FactoryPatternTest.java (driver class to test document creation)

## Step-by-Step to Run in Eclipse

1. Open Eclipse IDE.  
2. Go to File > New > Java Project.  
3. Name the project `FactoryMethodPattern` and click Finish.  
4. Right-click on `src` folder > New > Package (e.g., `com.patterns.factory.documents`).  
5. Copy the files from the `Documents` folder into this package.  
6. Repeat for `Factories` and `Test` packages as needed.  
7. Ensure all class files are placed correctly and there are no import errors.  
8. Run `FactoryPatternTest.java` as a Java Application.

## Output:

The program should create and print details about various document types based on factory selection, demonstrating the decoupling of object creation from usage.

# Module 1: Singleton Pattern

## Overview

The Singleton Pattern ensures that a class has only one instance and provides a global point of access to it. It is useful for logging, configuration, and resource sharing.

## File

Logger.java

## Step-by-Step to Run in Eclipse

1. Open Eclipse IDE.  
2. Go to File > New > Java Project.  
3. Name the project `SingletonPattern` and click Finish.  
4. Right-click on `src` > New > Package (e.g., `com.patterns.singleton`).  
5. Copy `Logger.java` into this package.  
6. Create a new class `TestLogger.java` to test singleton behavior:  
  
public class TestLogger {  
 public static void main(String[] args) {  
 Logger logger1 = Logger.getInstance();  
 Logger logger2 = Logger.getInstance();  
  
 logger1.log("First log message.");  
 logger2.log("Second log message.");  
  
 System.out.println(logger1 == logger2); // should print true  
 }  
}  
  
7. Run `TestLogger.java` as Java Application.

## Output

The logger should print the log messages and confirm both references are the same instance.