**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.

1. Create a new Java Project:

* Create a new Java Project named FactoryMethodPatternExample.
* A Java project named FactoryMethodPatternExample is created in eclipse IDE.

2. Define Document classes:

* Create interfaces or abstract classes for different document types such as **WordDocument**, **PdfDocument**, and **ExcelDocument**.

interface Document {

void open();

void close();

void save();

}

class WordDocument extends Document {

void open();

void close();

void save();

}

class PdfDocument extends Document {

void open();

void close();

void save();

}

class ExcelDocument extends Document {

void open();

void close();

void save();

}

* The Document interface defines common methods for all document types.
* WordDocument, PdfDocument, and ExcelDocument classes extend the Document interface.

3. Create Concrete Document Classes:

* Implement concrete classes for each document type that implements or extends the above interfaces or abstract classes.

class WordDocumentImpl implements WordDocument {

@Override

public void open() {

System.out.println("Opening Word document...");

}

@Override

public void close() {

System.out.println("Closing Word document...");

}

@Override

public void save() {

System.out.println("Saving Word document...");

}

}

class PdfDocumentImpl implements PdfDocument {

@Override

public void open() {

System.out.println("Opening PDF document...");

}

@Override

public void close() {

System.out.println("Closing PDF document...");

}

@Override

public void save() {

System.out.println("Saving PDF document...");

}

}

class ExcelDocumentImpl implements ExcelDocument {

@Override

public void open() {

System.out.println("Opening Excel document...");

}

@Override

public void close() {

System.out.println("Closing Excel document...");

}

@Override

public void save() {

System.out.println("Saving Excel document...");

}

}

* WordDocumentImpl, PdfDocumentImpl, and ExcelDocumentImpl implement their respective interfaces and provide specific implementations for the open(), close(), and save() methods.

4. Implement the Factory Method:

* Create an abstract class **DocumentFactory** with a method **createDocument()**.

public abstract class DocumentFactory {

public abstract Document createDocument();

}

* DocumentFactory is an abstract class with the abstract method createDocument().
* Create concrete factory classes for each document type that extends DocumentFactory and implements the **createDocument()** method.

class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocumentImpl();

}

}

class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocumentImpl();

}

}

class ExcelDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new ExcelDocumentImpl();

}

}

* WordDocumentFactory, PdfDocumentFactory, and ExcelDocumentFactory extend DocumentFactory and implement the createDocument() method to return instances of their respective document types.

5. Test the Factory Method Implementation:

* Create a test class to demonstrate the creation of different document types using the factory method.

public class FactoryMethodPatternExample {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

DocumentFactory factory = null;

System.out.println("Enter the type of document to create (word, pdf, excel) ");

String docType = scanner.nextLine();

switch (docType.toLowerCase()) {

case "word":

factory = new WordDocumentFactory();

break;

case "pdf":

factory = new PdfDocumentFactory();

break;

case "excel":

factory = new ExcelDocumentFactory();

break;

default:

System.out.println("Unknown document type.");

scanner.close();

return;

}

Document document = factory.createDocument();

document.open();

document.save();

document.close();

scanner.close();

}}

* In the FactoryMethodPatternExample class, user input is taken to determine which document type to create.
* Based on the user input (word, pdf, or excel), the appropriate factory is instantiated.
* The factory creates the corresponding document, and the open(), save(), and close() methods are called to demonstrate the implementation.
* Output:

