Design Patterns TP

El Gourji Nasreddine

October 2025

1 EX 1 :

Database.java > Database > getConnection(String name)

```java
public class Database {
    public static Database singleton ;
    private String name ;
    private Database( String n){
        this.name = n;
    }
    public static Database getConnection(String name){
        if (singleton == null){
            singleton = new Database (name);
        } else{
            System.out.println("The instance already exists . SO we will just ignore the name. ");
        }
        return  singleton ;
    }


    Run main | Debug main
    public static void main(String[] args ){
        Database db1 =  Database.getConnection("OneDb");

        Database db2 = Database.getConnection("TwoDb");
        System.out.println("Is the same instance : "+ (db1 == db2));
    }
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\HP\Desktop\Software development\TP\TP6> javac Database.java
PS C:\Users\HP\Desktop\Software development\TP\TP6> java Database
The instance already exists . SO we will just ignore the name
Is the same instance : true
PS C:\Users\HP\Desktop\Software development\TP\TP6>
```

Figure 1: Code and output of the first exercise.

2 EX 2 :

2.1 Naive Solution :

Client.java > Client > main(String[] args)

```java
public class Client{
    Run main | Debug main
    public static void main(String[] args){
        if(args.length==0){
            System.out.println("No args entered.");
            return ;
        }
        int arg = Integer.parseInt(args[0]); //casting to an integer from a string

        if (arg == 1){
            Program1 p = new Program1();
            System.out.println("I am main1");
            p.go();
        }
        else if (arg == 2){
            Program2 p = new Program2();
            System.out.println("I am main2");
            p.go();
        }
        else if (arg == 3){
            Program3 p = new Program3();
            System.out.println("I am main3");
            p.go();
        }
        else {
            System.out.println("wrong arg provided (out of range )");
        }
    }
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\HP\Desktop\Software development\TP\TP6> java Client 1
I am main1
Je suis le traitement 1
PS C:\Users\HP\Desktop\Software development\TP\TP6> java Client 2
I am main1
Je suis le traitement 2
PS C:\Users\HP\Desktop\Software development\TP\TP6> java Client 3
I am main1
```

Figure 2: Code and output of the second exercise part 1.

2.2 Apply Design Patterns:

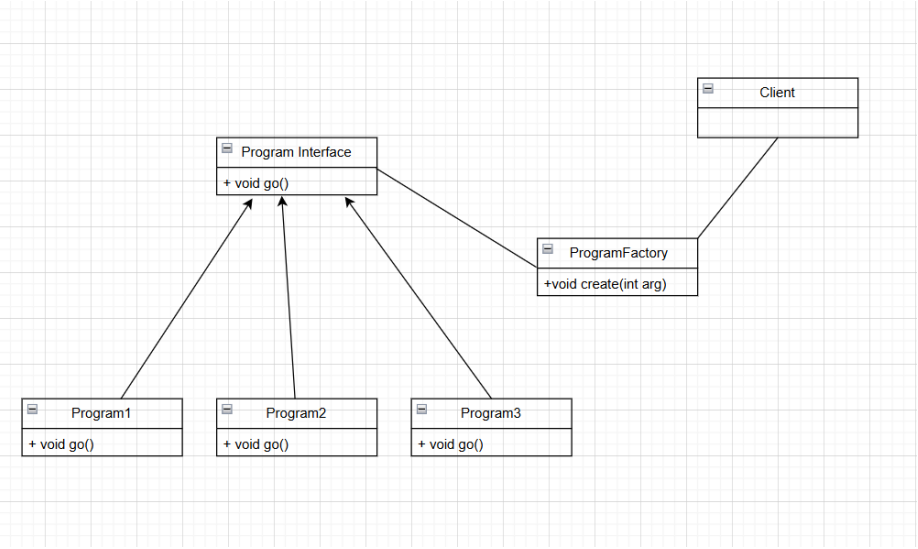

Figure 3: Class diagram

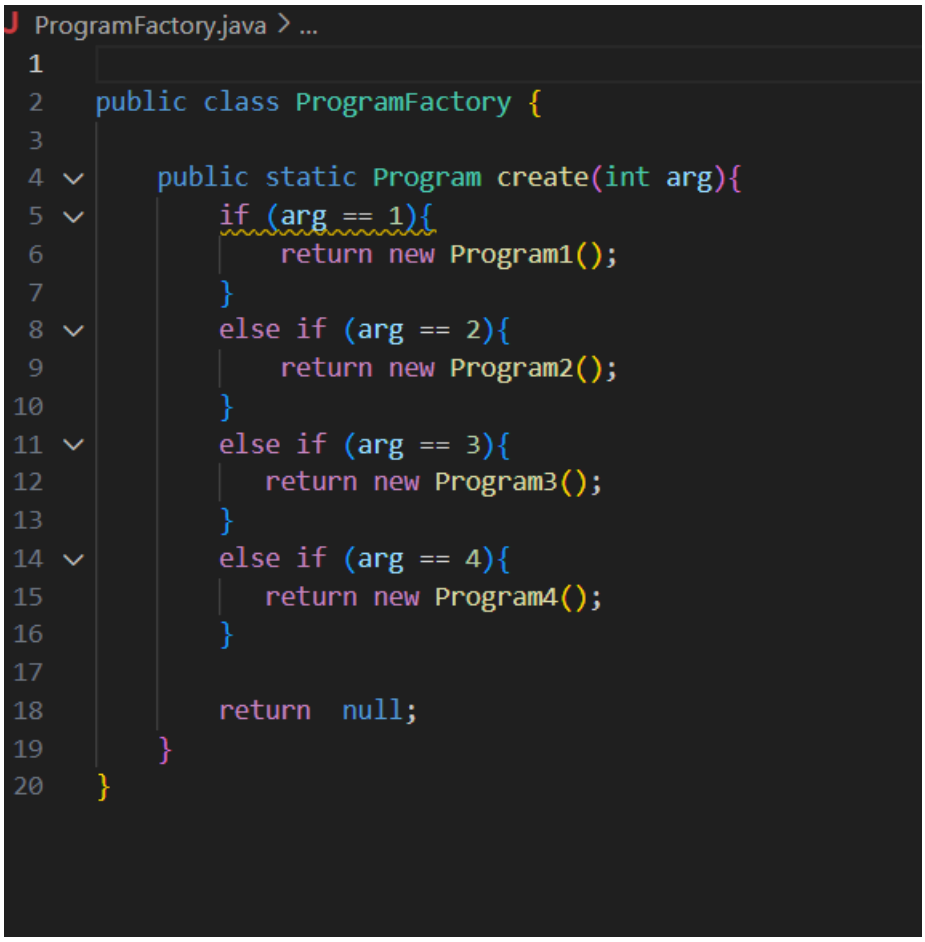

Figure 4: class ProgramFactory

```java
public class Program1 implements Program{
    public Program1 (){
    // The constructor does nothing .
    }
    public void go (){
        System . out . println ("Je suis le traitement 1") ;
    }
}
```

Figure 5: Class Program1 (Same implementation for the Program2,Program3 and Program4.)

```java
public  interface Program {
    void go();
}
```

Figure 6: interface Program

```java
public class Client{
    Run main | Debug main
    public static void main(String[] args){
        if(args.length==0){
            System.out.println("No args entered.");
            return ;
        }
        int arg = Integer.parseInt(args[0]); //casting to an integer from a string
        Program p  = ProgramFactory.create(arg);

        p.go() ;

    }
}
```

Figure 7: Updated Client Class