UNIVERSITY OF INFORMATION TECHNOLOGY AND SCIENCES (UITS)

DEPARTMENT OF INFORMATION TECHNOLOGY

LAB-1:

IT-324 : DESIGN PATTERN LAB

Singletone Design Pattern Implementation: Early Instantiation

Submitted To:

SIFAT NAWRIN NOVA
Lecturer,
Department of IT, UITS

Submitted By:

Name:Nazmul Zaman
Student ID:2014755055
Department of IT, UITS

Contents

1 Theory

The singleton design pattern is one of the twenty-three well-known "Gang of Four" design patterns that describe how to solve recurring design problems to design flexible and reusable object-oriented software with the aim of making it easier to implement, change, test, and reuse objects.

2 Abstraction

In this lab report we learn about singleton design pattern method and how it's work and why we used this method. In Java, Singleton is a design pattern that ensures that a class can only have one object. To create a singleton class, a class must implement the following properties: Create a private constructor of the class to restrict object creation outside of the class. Singleton is like a single resource which is being shared among multiple users; for example - sharing a single washing machine among all the residents in a hotel or sharing a single appliance like refrigerator among all the family members

3 Introduction

Singleton pattern is one of the simplest design patterns in Java. This type of design pattern comes under creational pattern as this pattern provides one of the best ways to create an object.

This pattern involves a single class which is responsible to create an object while making sure that only single object gets created. This class provides a way to access its only object which can be accessed directly without need to instantiate the object of the class.

The singleton pattern is a software design pattern that restricts the instantiation of a class to one object and provides a global access to it. Types:
1. Early Instantiation
2. Lazy Instantiation

4 Working Procedure

We're going to create a SingleObject class. SingleObject class have its constructor as private and have a static instance of itself.

SingleObject class provides a static method to get its static instance to outside world. SingletonPatternDemo, our demo class will use SingleObject class to get a SingleObject object.

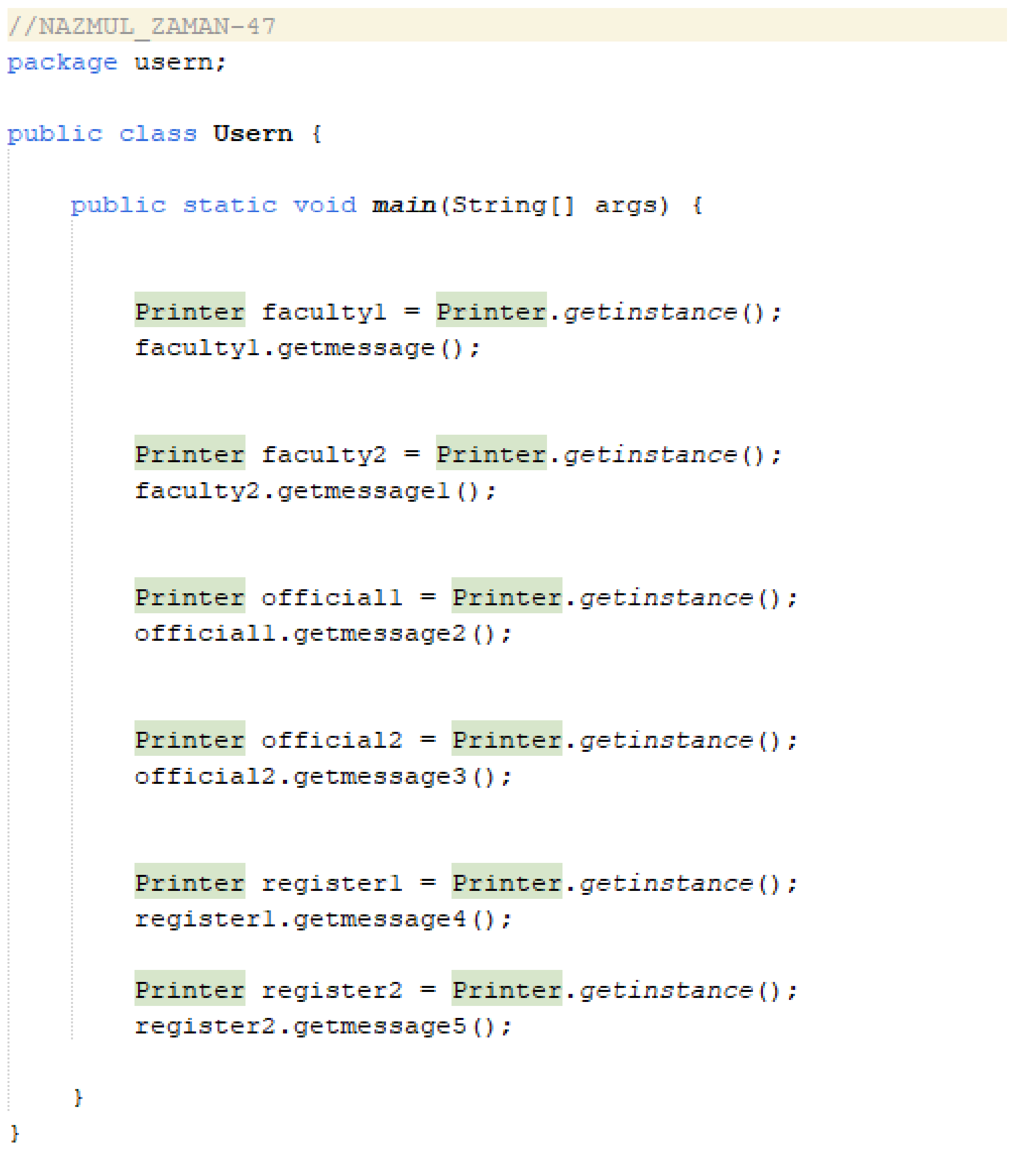
Static member: It gets memory only once because of static, itcontains the instance of the Singleton class.

Private constructor: It will prevent to instantiate the Singleton class from outside the class.

Static factory method: This provides the global point of access to the Singleton object and returns the instance to the caller.

5 Source Code

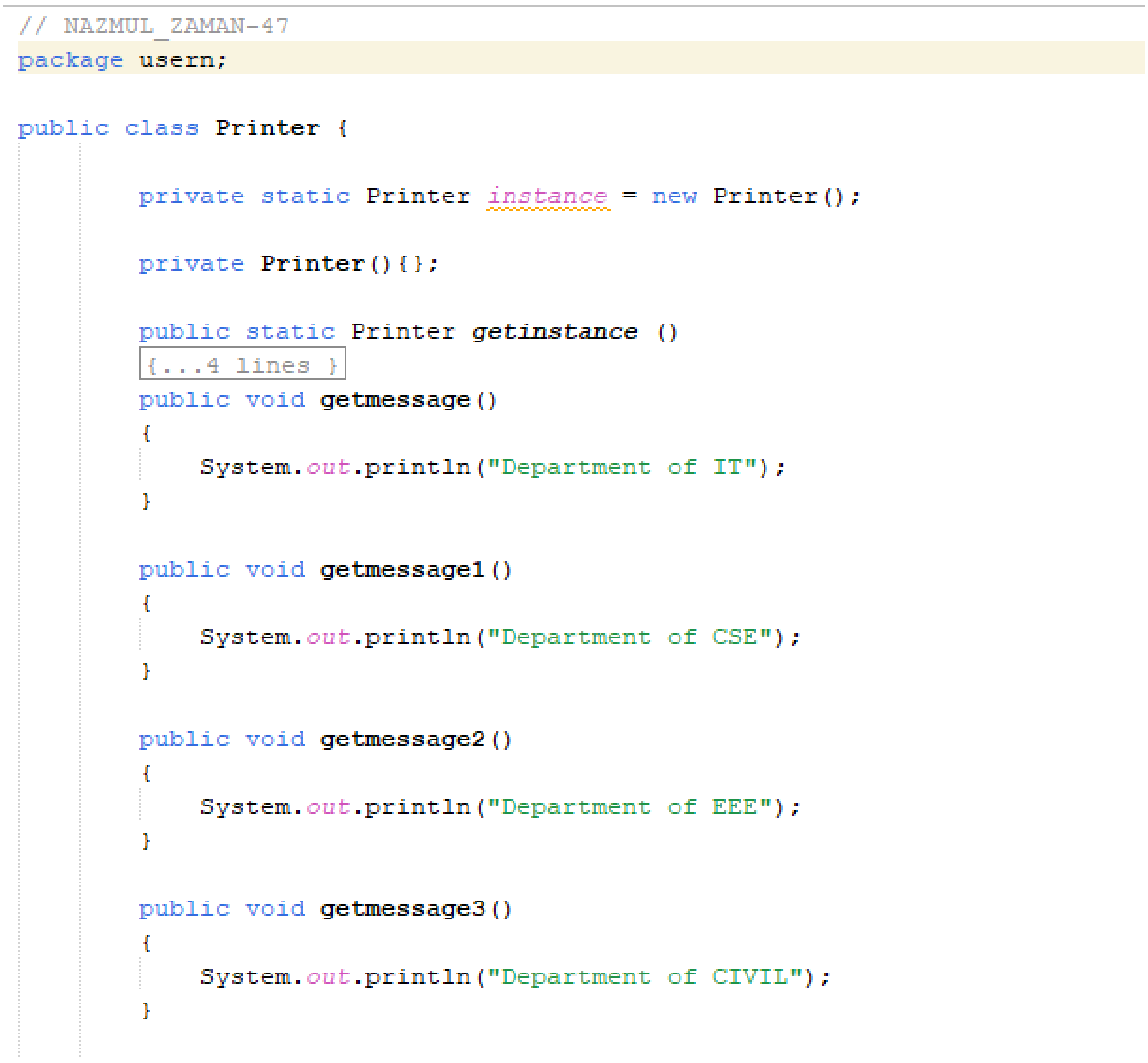
5.1 Package Name: Usern



```
//NAZMUL_ZAMAN-47
package usern;

public class Usern {

    public static void main(String[] args) {


        Printer faculty1 = Printer.getinstance();
        faculty1.getmessage();


        Printer faculty2 = Printer.getinstance();
        faculty2.getmessage1();


        Printer official1 = Printer.getinstance();
        official1.getmessage2();


        Printer official2 = Printer.getinstance();
        official2.getmessage3();


        Printer register1 = Printer.getinstance();
        register1.getmessage4();

        Printer register2 = Printer.getinstance();
        register2.getmessage5();

    }
}
```

Figure 1: Usern part

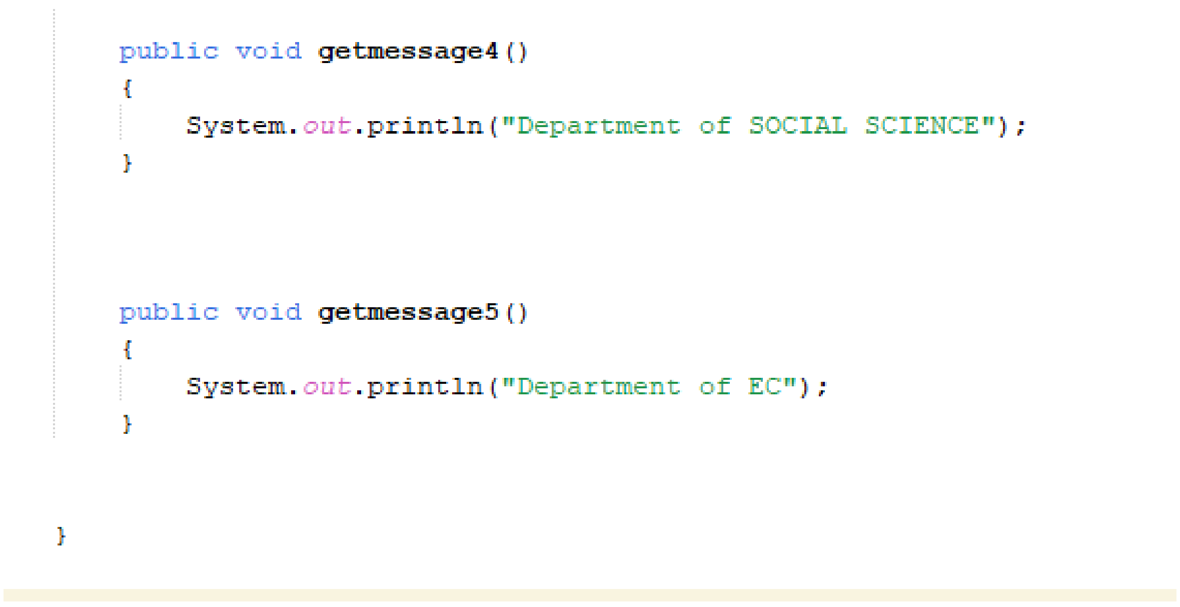
5.2 Class Name: Printer



```
// NAZMUL_ZAMAN-47
package usern;

public class Printer {

        private static Printer instance = new Printer();

        private Printer(){};

        public static Printer getinstance ()
        {...4 lines }
        public void getmessage()
        {
            System.out.println("Department of IT");
        }

        public void getmessage1()
        {
            System.out.println("Department of CSE");
        }

        public void getmessage2()
        {
            System.out.println("Department of EEE");
        }

        public void getmessage3()
        {
            System.out.println("Department of CIVIL");
        }
```

Figure 2: Printer part

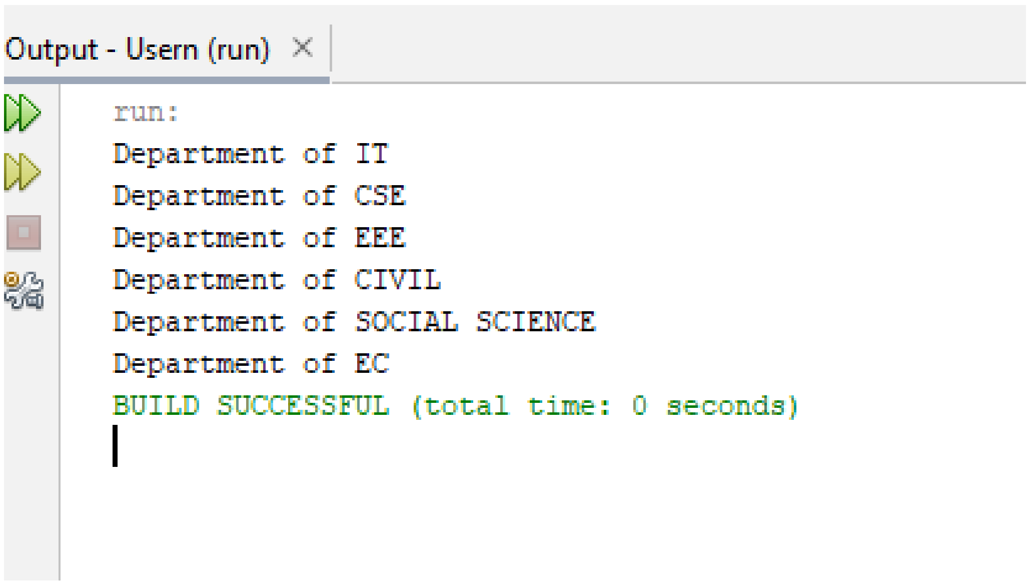
5.3 Class Name: Printer(Last part)



```
    public void getmessage4()
    {
        System.out.println("Department of SOCIAL SCIENCE");
    }



    public void getmessage5()
    {
        System.out.println("Department of EC");
    }


}
```

Figure 3: Printer part

5.4 OUTPUT



```
run:
Department of IT
Department of CSE
Department of EEE
Department of CIVIL
Department of SOCIAL SCIENCE
Department of EC
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 4: Output Part

6 Conclusion

In this lab report we learn about singleton design pattern in practically.

How we used singleton design pattern with real life example.It is used where only a single instance of a class is required to control the action throughout the execution.

7 References

1.https://www.tutorialspoint.com/design$_pattern/singleton_pattern.htm$
2.$https://www.geeksforgeeks.org/singleton-design-pattern-introduction$
3.$https://www.javatpoint.com/singleton-design-pattern-in-java$