University of Information Technology and Sciences (UITS)

Department of Information Technology

Lab Report : 2

IT-324 : Design Pattern Lab

Singletone Design Pattern Implementation: Lazy Instantiation

Submitted To:

Sifat Nawrin Nova
Lecturer,
Department of IT, UITS
Email:sifat.nawrin@uits.edu.bd

Submitted By:

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

Contents

1 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.

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

2 Theory

Lazy initialization: In this method, object is created only if it is needed. This may prevent resource wastage. An implementation of getInstance() method is required which return the instance. There is a null check that if object is not created then create, otherwise return previously created.

3 Objective

we are going to learn: - how to create a class
- how to create an instance
- how to create object
- how to create private constructor
- how to create a method
- how to call a method from another class using objec

3.1 Differences between Early and Lazy instantiation. method.

Early Instantiation: creation of instance at load time. Lazy Instantiation: Creation of instance when required.
Lazy Instantiation: The instance could be initialized only when the Singleton Class is used for the first time. Doing so creates the instance of the Singleton class in the JVM Java Virtual Machine during the execution of the method, which gives access to the instance, for the first time.

4 Working Procedure

we have different approaches but all of them have the following common concepts.

1. Private constructor to restrict instantiation of the class from other classes.
2. Private static variable of the same class that is the only instance of the class.
3. Public static method that returns the instance of the class, this is the global access point for outer world to get the instance of the singleton class.

Lazy initialization method to implement Singleton pattern creates the instance in the global access method.To implement a Singleton pattern.After created instance() method.It'll get instantiated when someone will call the getinstance() method. It'll check if it's null then object will be created and will be returned , else it's means already created instance then just return by reference of Singleton instance.
In early loading the instance is created once the class is loaded.

5 Source Code

5.1 (Lab2 class/Main Class)

```

package lab2;

public class Lab2 {

public static void main ( String [] args )

{

SingletonLazy obj = SingletonLazy.getinstance();

SingletonLazy obj1 = SingletonLazy.getinstance();

if (obj == obj1)
{

 System.out.println("STUDENT IN DEPARTMENT OF IT");

}
else
{
  System.out.println("NOT IN DEPARTMENT OF IT");
}
    obj.getmessage();
    obj1.getmessage1();


   }

}
```

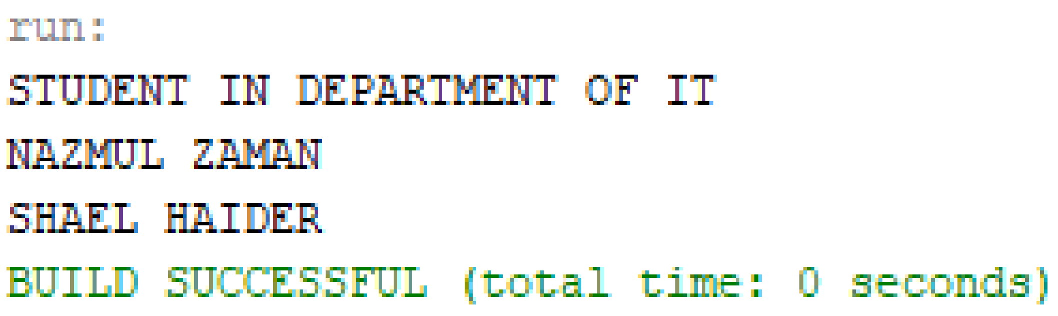
?

5.2 (SingletonLazy Class)

```

package lab2;

class SingletonLazy {

            private static SingletonLazy instance ;
            private SingletonLazy () {};

             public static SingletonLazy getinstance ()
            {


                 if ( instance == null )
                 instance = new  SingletonLazy () ;

                 return instance ;
            }


            public void getmessage ()
             {

            System.out.println("NAZMUL ZAMAN");

             }

             public void getmessage1 ()
            {
                System . out . println ("SHAEL HAIDER" ) ;
            }

}
```

?

5.3 Output



```
run:
STUDENT IN DEPARTMENT OF IT
NAZMUL ZAMAN
SHAEL HAIDER
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 1: Output

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 also we got a knowledge about private constructor,method,object,private attribute of a class etc.

7 References

1.https://www.tutorialspoint.com/design$_{p}attern/singleton_{p}attern.htm$
2.$https://www.geeksforgeeks.org/singleton-design-pattern-introduction$
3.$https://www.javatpoint.com/singleton - design - pattern - in - java$