Design pattern

**How to Design/Create a Singleton Class in Java?**

To create a singleton class, we must follow the steps, given below:

**1.**Ensure that only one instance of the class exists.

**2.**Provide global access to that instance by

* Declaring all constructors of the class to be private.
* Providing a static method that returns a reference to the instance. The lazy initialization concept is used to write the static methods.
* The instance is stored as a private static variable.

**Key Concepts of Singleton Method:**

* **Private Constructor:** The Singleton class has a private constructor to prevent the instantiation of the class from external entities.
* **Private Instance:** The class contains a private static instance of itself.
* **Static Method:** A static method is provided to access the instance, and it ensures that only one instance is created if it doesn’t exist.

// Java program implementing Singleton class

// with using getInstance() method

// Class 1

// Helper class

class Singleton {

// Static variable reference of single\_instance

// of type Singleton

private static Singleton single\_instance = null;

// Declaring a variable of type String

public String s;

// Constructor

// Here we will be creating private constructor

// restricted to this class itself

private Singleton()

{

s = "Hello I am a string part of Singleton class";

}

// Static method

// Static method to create instance of Singleton class

public static synchronized Singleton getInstance()

{

if (single\_instance == null)

single\_instance = new Singleton();

return single\_instance;

}

}

// Class 2

// Main class

class GFG {

// Main driver method

public static void main(String args[])

{

// Instantiating Singleton class with variable x

Singleton x = Singleton.getInstance();

// Instantiating Singleton class with variable y

Singleton y = Singleton.getInstance();

// Instantiating Singleton class with variable z

Singleton z = Singleton.getInstance();

// Printing the hash code for above variable as

// declared

System.out.println("Hashcode of x is "

+ x.hashCode());

System.out.println("Hashcode of y is "

+ y.hashCode());

System.out.println("Hashcode of z is "

+ z.hashCode());

// Condition check

if (x == y && y == z) {

// Print statement

System.out.println(

"Three objects point to the same memory location on the heap i.e, to the same object");

}

else {

// Print statement

System.out.println(

"Three objects DO NOT point to the same memory location on the heap");

}

}

}