Singleton Class in Java

In object-oriented programming, a singleton class is a class that can have only one object (an instance of the class) at a time.  
After first time, if we try to instantiate the Singleton class, the new variable also points to the first instance created. So whatever modifications we do to any variable inside the class through any instance, it affects the variable of the single instance created and is visible if we access that variable through any variable of that class type defined.  
To design a singleton class:

1. Make constructor as private.
2. Write a static method that has return type object of this singleton class. Here, the concept of [Lazy initialization](https://en.wikipedia.org/wiki/Lazy_initialization)in used to write this static method.

**Normal class vs Singleton class**: Difference in normal and singleton class in terms of instantiation is that, For normal class we use constructor, whereas for singleton class we use getInstance() method (Example code:I). In general, to avoid confusion we may also use the class name as method name while defining this method (Example code:II).

**Implementing Singleton class with getInstance() method**

filter\_none

edit

play\_arrow

brightness\_4

|  |
| --- |
| // Java program implementing Singleton class  // with getInstance() method  class Singleton  {      // static variable single\_instance of type Singleton      private static Singleton single\_instance = null;        // variable of type String      public String s;        // private constructor restricted to this class itself      private Singleton()      {          s = "Hello I am a string part of Singleton class";      }        // static method to create instance of Singleton class      public static Singleton getInstance()      {          if (single\_instance == null)              single\_instance = new Singleton();            return single\_instance;      }  }    // Driver Class  class Main  {      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();            // changing variable of instance x          x.s = (x.s).toUpperCase();            System.out.println("String from x is " + x.s);          System.out.println("String from y is " + y.s);          System.out.println("String from z is " + z.s);          System.out.println("\n");            // changing variable of instance z          z.s = (z.s).toLowerCase();            System.out.println("String from x is " + x.s);          System.out.println("String from y is " + y.s);          System.out.println("String from z is " + z.s);      }  } |

Output:

String from x is HELLO I AM A STRING PART OF SINGLETON CLASS

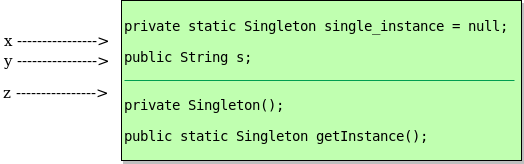
String from y is HELLO I AM A STRING PART OF SINGLETON CLASS

String from z is HELLO I AM A STRING PART OF SINGLETON CLASS

String from x is hello i am a string part of singleton class

String from y is hello i am a string part of singleton class

String from z is hello i am a string part of singleton class

  
**Explanation:** In the Singleton class, when we first time call getInstance() method, it creates an object of the class with name single\_instance and return it to the variable. Since single\_instance is static, it is changed from null to some object. Next time, if we try to call getInstance() method, since single\_instance is not null, it is returned to the variable, instead of instantiating the Singleton class again. This part is done by if condition.

In the main class, we instantiate the singleton class with 3 objects x, y, z by calling static method getInstance(). But actually after creation of object x, variables y and z are pointed to object x as shown in the diagram. Hence, if we change the variables of object x, that is reflected when we access the variables of objects y and z. Also if we change the variables of object z, that is reflected when we access the variables of objects x and y.

**Implementing Singleton class with method name as that of class name**

filter\_none

edit

play\_arrow

brightness\_4

|  |
| --- |
| // Java program implementing Singleton class  // with method name as that of class  class Singleton  {      // static variable single\_instance of type Singleton      private static Singleton single\_instance=null;        // variable of type String      public String s;        // private constructor restricted to this class itself      private Singleton()      {          s = "Hello I am a string part of Singleton class";      }        // static method to create instance of Singleton class      public static Singleton Singleton()      {          // To ensure only one instance is created          if (single\_instance == null)          {              single\_instance = new Singleton();          }          return single\_instance;      }  }    // Driver Code  class Main  {      public static void main(String args[])      {          // instantiating Singleton class with variable x          Singleton x = Singleton.Singleton();            // instantiating Singleton class with variable y          Singleton y = Singleton.Singleton();            // instantiating Singleton class with variable z          Singleton z = Singleton.Singleton();            // changing variable of instance x          x.s = (x.s).toUpperCase();            System.out.println("String from x is " + x.s);          System.out.println("String from y is " + y.s);          System.out.println("String from z is " + z.s);          System.out.println("\n");            // changing variable of instance x          z.s = (z.s).toLowerCase();            System.out.println("String from x is " + x.s);          System.out.println("String from y is " + y.s);          System.out.println("String from z is " + z.s);      }  } |

Output:

String from x is HELLO I AM A STRING PART OF SINGLETON CLASS

String from y is HELLO I AM A STRING PART OF SINGLETON CLASS

String from z is HELLO I AM A STRING PART OF SINGLETON CLASS

String from x is hello i am a string part of singleton class

String from y is hello i am a string part of singleton class

String from z is hello i am a string part of singleton class

**Explanation:** In the Singleton class, when we first time call Singleton() method, it creates an object of class Singleton with name single\_instance and return it to the variable. Since single\_instance is static, it is changed from null to some object. Next time if we try to call Singleton() method, since single\_instance is not null, it is returned to the variable, instead of instantiating the Singleton class again.