What are java **singleton classes** :

For any java class, if we are allowed to create only one object then it is called as singleton class. Eg. : Runtime, BusinessDelegate, ServiceLocator etc are some examples of singleton classes.

Advantage of Singleton class :

If several people have same requirement then it is not recommended to create a separate object for every requirement. We have to create only one object and we can re-use the same object for every similar requirement. So that performance and memory utilization will be improved. This is central idea of singleton classes.

We cannot create singleton classes with constructor or new operator. Because with these approaches, one can create multiple objects. We use Factory methods to create singleton class object. **Runtime.getRuntime()** never creates a new objects, it always returns existing object.

Eg. :

Runtime r1 = Runtime.getRuntime();

Runtime r2 = Runtime.getRuntime();

.

.

Runtime r100 = Runtime.getRuntime();

How to create our own singleton classes : We can create our own singleton classes for this we have to use **Private constructor** and private static variable and public factory method.

Approach 1 :

Class Test

{

Private static Test t = new Test(); // We are creating object here

Private Test() //private constructor

{

}

Public static Test getTest()

{

Return t;

}

}

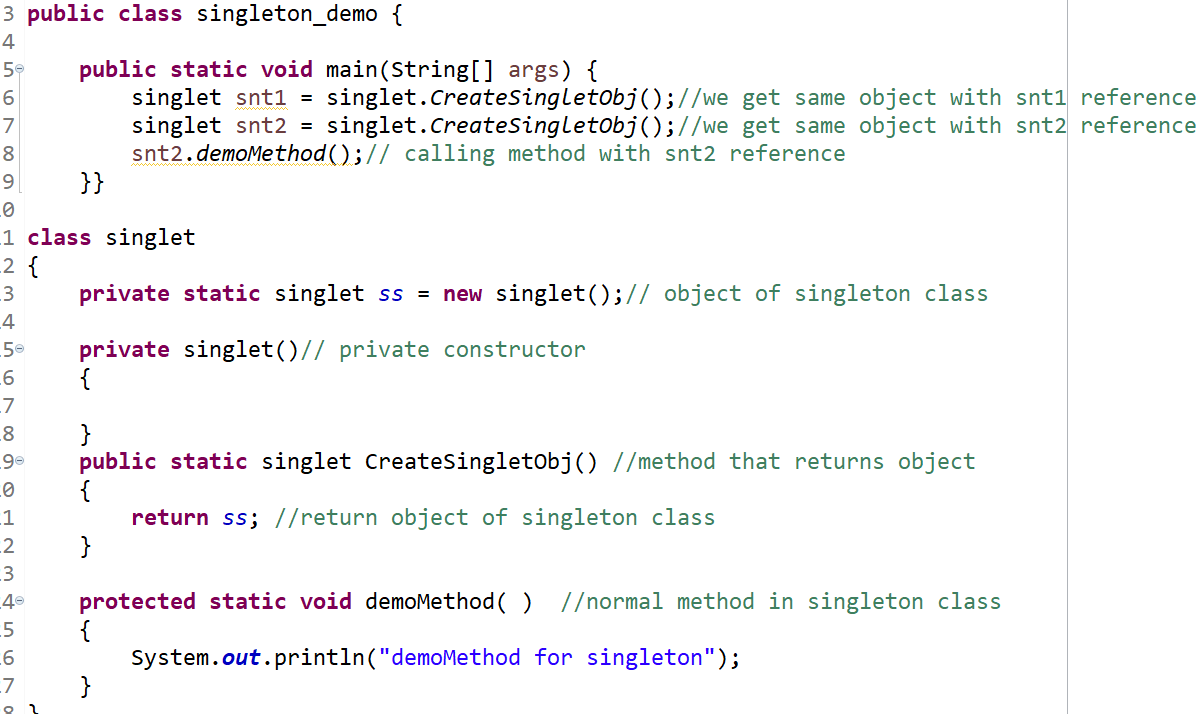
Create object (It is already created we just assign new references to it)

Test t1 = Test.getTest(); // Same object with t1 as reference

Test t2 = Test.getTest(); // Same object with t2 as reference

Note : Runtime class is internally implemented by using the above approach.

Example :



Approach 2 :

Class Test

{

Private static test t = null; // We are declaring object here

Private Test() //private constructor

{

}

Public static Test getTest()

{

If(t==null)

{

T = new Test(); // We are creating object here only when it Is requested for the first time.

}

Return t;

}

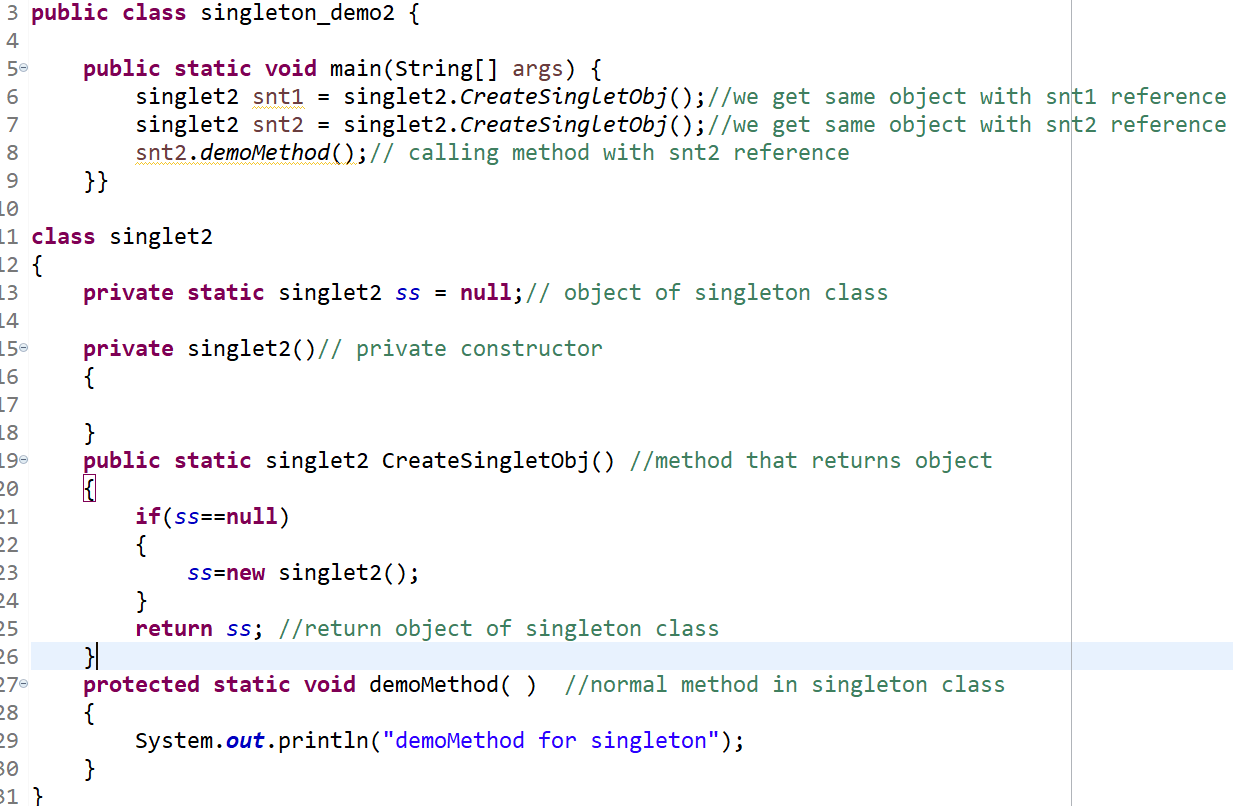
}

Create object (It is already created we just assign new references to it)

Test t1 = Test.getTest(); // Same object with t1 as reference

Test t2 = Test.getTest(); // Same object with t2 as reference

Example :



**At any point of time for test class we can create only one object hence test class is singleton class.**

Q. A class is not final but we are not allowed to create child classes. How it is possible?

Ans : Declare every constructor as private.