**String:**

**What is String Constant Pool (SCP) concept?**

When we create String object without using new operator the objects are created in SCP (String constant pool) area.

String s1=”ABC”; String s2=”DEF”;

S1

ABC

2

S3

DEF

String s3=”DEF”; S

When we create object in SCP area then

just before object creation it is always checking previous objects.

If the previous object is available with the same content

SCP Area

then it won’t create new object that reference variable pointing to existing object.

If the previous objects are not available then JVM will create new object.

SCP area does not allow duplicate objects.

**When does object get created in SCP?**

* When String object is created with new keyword.
* When only String variable with content is created.

**Why String is immutable?**

* String object is immutable means its internal state remains constant after it has been entirely created. This means that once the object has been assigned to a variable, we can neither update the reference nor mutate the internal state by any means.
* The String is widely used in Java applications to store sensitive pieces of information like usernames, passwords, connection URLs, network connections, etc. It's also used extensively by JVM class loaders while loading classes.
* Hence securing String class is crucial regarding the security of the whole application in general.
* Being immutable automatically makes the String thread safe since they won't be changed when accessed from multiple threads.
* As we saw previously, String pool exists because Strings are immutable. In turn, it enhances the performance by saving heap memory.
* E.g. Database connection properties are stored in String, so that no one can update that object to hack database data and database security can be achieved.

**Difference between String & StringBuffer?**

|  |  |
| --- | --- |
| String | StringBuffer |
| String class is immutable. | StringBuffer class is mutable. |
| String is slow and consumes more memory when you concat too many strings because  every time it creates new instance. | StringBuffer is fast and consumes less memory when you cancat strings. |
| String class overrides the equals() method  of Object class. | StringBuffer class doesn't override the  equals() method of Object class. |

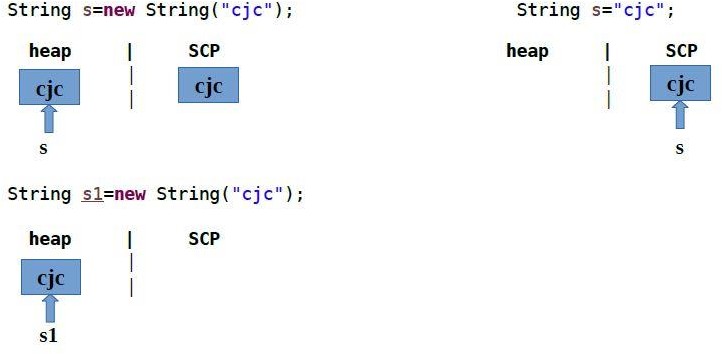
**What are the String Scenarios related to String object creation while using Concat ( )?**

Concatenation of String : String s="cjc";

String s1=s.concat("class");



**String scenarios :**



**How to use SCP objects in intern ( ) method?**

It can be used to return string from memory, if it is created by new keyword. It creates exact copy of heap string object in string constant pool.

E.g. (Sample Code) public class InternExample2 {

public static void main(String[] args) {

String s1 = "Javatpoint";

String s2 = s1.intern();

String s3 = new String("Javatpoint");

String s4 = s3.intern();

System.out.println(s1==s2); // True

System.out.println(s1==s3); // False

System.out.println(s1==s4); // True

System.out.println(s2==s3); // False

System.out.println(s2==s4); // True

System.out.println(s3==s4); // False

}

}

**How to create immutable class?**

* The instance variable of the class is final i.e. we cannot change the value of it after creating an object.
* The class is final so we cannot create the subclass.
* There is no setter methods i.e. we have no option to change the value of the instance variable.