**String in Java-**

String is the class that represents sequence of character.

Package is Java. Lang.

It’s a non-primitive data type.

String class implements Serializable, comparable, char sequence interface.

String is the immutable, once string object is created, it cannot changed but new string object is created.

Example-

**public** **class** StringDemo {

**public** **static** **void** main(String[] args) {

String s = "ram";

s.concat("patil");// concat() method appends the string at the end

System.***out***.println(s);// will print ram because strings are immutable objects

}

}

For mutable class, you can use String buffer and String builder class.

How to create the string object?

1. String literal
2. By new keyword
3. String literal-

It is created by using double quotes.

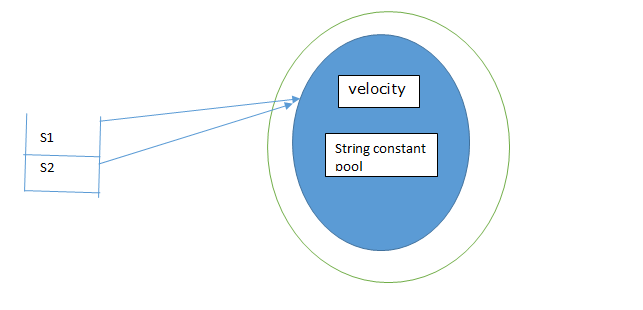
Example- String s=” velocity”;

Each time when you create string literal, the JVM check string constant pool first, if the string object is already present in the pool, reference to pooled instance is returned. If string does not present in the pool, new string instance is created and placed in pool.

Example- String s1=”velocity”;

String s2=” velocity”; //will not create the new instance.

In the above example, only one object will be created, firstly JVM will not find any string object with value “velocity” in the string constant pool, so it will create new object. After that it will find string with value= “velocity” in pool, it will not create the new object but will return reference to same instance.



Why Java uses the concept of string literal?

To make the java more memory efficient (because no new object is created if it exist already in string constant pool.)

1. By new Keyword-

Example- String s= new String(“pune”);

/\* create two objects \*/

In such case, JVM will create the new String object in normal(non-pool) heap memory and literal “pune” will be placed in string constant pool. The variable s refer to object in heap(non-pool).

**Q. Why String is immutable?**

String is the immutable, once string object is created, it cannot change but new string object is created.

Example-

public class StringDemo {

public static void main(String[] args) {

String s = "ram";

s.concat("patil");// concat() method appends the string at the end

System.out.println(s);// will print ram because strings are immutable objects

}

}

Example- 1

**package** com.velocity;

**public** **class** StringDemo {

**public** **static** **void** main(String[] args) {

String s1 = "velocity";

String s2 = **new** String("velocity");

System.***out***.println(s1 == s2);

System.***out***.println(s1.equals(s2));

}

}

Note:

1) “==” operator compares reference or memory location of objects in a heap, whether they point to the same location or not.

2) The String class overrides the equals method it inherited from the Object class and implemented logic to compare the two String objects character by character

Example- 2

**package** com.velocity;

**public** **class** StringDemo {

**public** **static** **void** main(String[] args) {

String s1 = "velocity";

String s2 = **new** String("pune");

s2=s1;

System.***out***.println(s1 == s2);

System.***out***.println(s1.equals(s2));

}

}

Output-

Example-3

**package** com.velocity;

**public** **class** StringDemo {

**public** **static** **void** main(String[] args) {

String s1 = **new** String("velocity");

String s2 = **new** String("pune");

s2 = s1;

System.***out***.println(s1 == s2);

System.***out***.println(s1.equals(s2));

}

}

Output-

Example- 4

**package** com.velocity;

**public** **class** StringDemo {

**public** **static** **void** main(String[] args) {

**public** **class** PropertiesFileDemo {

**public** **static** **void** main(String[] args) {

String s1 = "velocity";

String s2 = **new** String("pune");

System.***out***.println(s1 == s2);

System.***out***.println(s1.equals(s2));

System.***out***.println(s1.hashCode());

System.***out***.println(s2.hashCode());

}

}

Output-

Example-5

**package** com.velocity;

**public** **class** StringDemo {

**public** **static** **void** main(String[] args) {

String s1 = **new** String("velocity");

String s2 = **new** String("pune");

System.***out***.println(s1 == s2);

System.***out***.println(s1.equals(s2));

}

}

Output-