

Java String

In java, string is basically an object that represents sequence of char values. An array of characters works same as java string. For example:

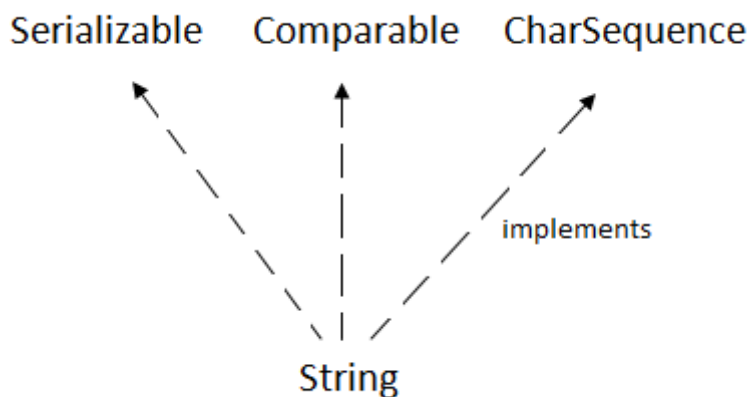
```
char[] ch={'j','a','v','a','t','p','o','i','n','t'};  
String s=new String(ch);
```

is same as:

```
String s="javatpoint";
```

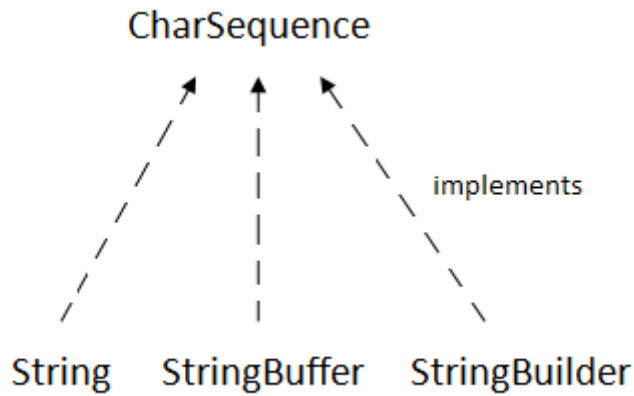
Java String class provides a lot of methods to perform operations on string such as `compare()`, `concat()`, `equals()`, `split()`, `length()`, `replace()`, `compareTo()`, `intern()`, `substring()` etc.

The `java.lang.String` class implements *Serializable*, *Comparable* and *CharSequence* interfaces.



CharSequence Interface

The `CharSequence` interface is used to represent sequence of characters. It is implemented by `String`, `StringBuffer` and `StringBuilder` classes. It means, we can create string in java by using these 3 classes.



The java String is immutable i.e. it cannot be changed. Whenever we change any string, a new instance is created. For mutable string, you can use StringBuffer and StringBuilder classes.

We will discuss about immutable string later. Let's first understand what is string in java and how to create the string object.

What is String in java

Generally, string is a sequence of characters. But in java, string is an object that represents a sequence of characters. The `java.lang.String` class is used to create string object.

How to create String object?

There are two ways to create String object:

1. By string literal
2. By new keyword

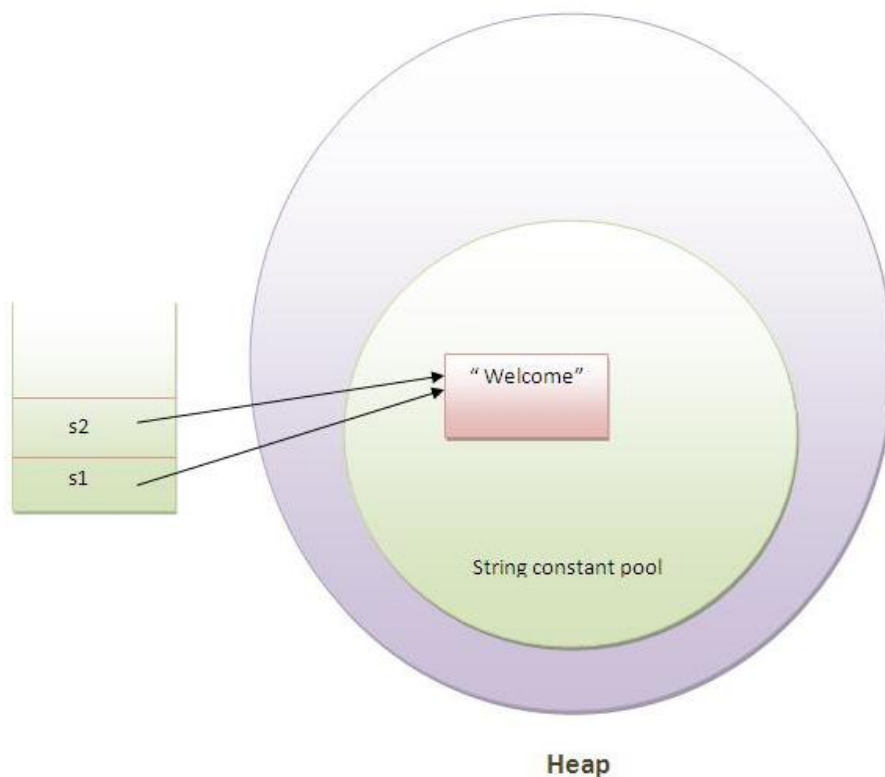
1) String Literal

Java String literal is created by using double quotes. For Example:

```
String s="welcome";
```

Each time you create a string literal, the JVM checks the string constant pool first. If the string already exists in the pool, a reference to the pooled instance is returned. If string doesn't exist in the pool, a new string instance is created and placed in the pool. For example:

```
String s1="Welcome";  
String s2="Welcome";//will not create new instance
```



In the above example only one object will be created. Firstly JVM will not find any string object with the value "Welcome" in string constant pool, so it will create a new object. After that it will find the string with the value "Welcome" in the pool, it will not

create new object but will return the reference to the same instance.

Note: String objects are stored in a special memory area known as string constant pool.

Why java uses concept of string literal?

To make Java more memory efficient (because no new objects are created if it exists already in string constant pool).

2) By new keyword

```
String s=new String("Welcome");//creates two objects and one reference variable
```

In such case, JVM will create a new string object in normal(non pool) heap memory and the literal "Welcome" will be placed in the string constant pool. The variable s will refer to the object in heap(non pool).

Java String Example

```
public class StringExample{  
public static void main(String args[]){  
    String s1="java";//creating string by java string literal  
    char ch[]={'s','t','r','i','n','g','s'};  
    String s2=new String(ch);//converting char array to string  
    String s3=new String("example");//creating java string by new keyword  
    System.out.println(s1);  
    System.out.println(s2);  
    System.out.println(s3);  
}}
```

Test it Now

```
java  
strings
```

example

Java String class methods

The java.lang.String class provides many useful methods to perform operations on sequence of char values.

No.	Method	Description
1	char charAt(int index)	returns char value for the particular index
2	int length()	returns string length
3	static String format(String format, Object... args)	returns formatted string
4	static String format(Locale l, String format, Object... args)	returns formatted string with given locale
5	String substring(int beginIndex)	returns substring for given begin index
6	String substring(int beginIndex, int endIndex)	returns substring for given begin index and end index
7	boolean contains(CharSequence s)	returns true or false after matching the sequence of char value
8	static String join(CharSequence delimiter, CharSequence... elements)	returns a joined string
9	static String join(CharSequence delimiter, Iterable<? extends CharSequence> elements)	returns a joined string
10	boolean equals(Object another)	checks the equality of string with object
11	boolean isEmpty()	checks if string is empty
12	String concat(String str)	concatinates specified string
13	String replace(char old, char new)	replaces all occurrences of specified char value

14	<code>String replace(CharSequence old, CharSequence new)</code>	replaces all occurrences of specified CharSequence
15	<code>static String equalsIgnoreCase(String another)</code>	compares another string. It doesn't check case.
16	<code>String[] split(String regex)</code>	returns splitted string matching regex
17	<code>String[] split(String regex, int limit)</code>	returns splitted string matching regex and limit
18	<code>String intern()</code>	returns interned string
19	<code>int indexOf(int ch)</code>	returns specified char value index
20	<code>int indexOf(int ch, int fromIndex)</code>	returns specified char value index starting with given index
21	<code>int indexOf(String substring)</code>	returns specified substring index
22	<code>int indexOf(String substring, int fromIndex)</code>	returns specified substring index starting with given index
23	<code>String toLowerCase()</code>	returns string in lowercase.
24	<code>String toLowerCase(Locale l)</code>	returns string in lowercase using specified locale.
25	<code>String toUpperCase()</code>	returns string in uppercase.
26	<code>String toUpperCase(Locale l)</code>	returns string in uppercase using specified locale.
27	<code>String trim()</code>	removes beginning and ending spaces of this string.
28	<code>static String valueOf(int value)</code>	converts given type into string. It is overloaded.

Do You Know ?

- Why String objects are immutable?
- How to create an immutable class?
- What is string constant pool?

- What code is written by the compiler if you concat any string by + (string concatenation operator)?
- What is the difference between StringBuffer and StringBuilder class?

What we will learn in String Handling ?

- Concept of String
- Immutable String
- String Comparison
- String Concatenation
- Concept of Substring
- String class methods and its usage
- StringBuffer class
- StringBuilder class
- Creating Immutable class
- toString() method
- StringTokenizer class

[next →](#)

Share this page



Latest 4 Tutorials

