# Java String

In [Java](https://www.javatpoint.com/java-tutorial), string is basically an object that represents sequence of char values. An [array](https://www.javatpoint.com/array-in-java) of characters works same as Java string. For example:

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

is same as:

1. String s="javatpoint";

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

**Creating String:-**There are three ways to create string in java:

We can create astring just by assigning agroup of character to a string type variable.

String s;

S=”hello”;

We can creating an object to string by allocating memory by using new operator

String s =new String (“hello”);

The third way to create

char arr[]={‘c’,’h’,’a’,’i’,’r’,’s’};

string s =nrw string(arr);

String class method

The concat() method appends (concatenate) a string to the end of another string.

Syntax:

public String concat(String string2)

example : String firstName = "John ";

String lastName = "Doe";

System.out.println(firstName.concat(lastName));

“JohnDoe”

Length( ):

The length() method returns the length of a specified string.

**Note:** The length of an empty string is 0.

Syntax

public int length()

The length() method returns the length of a specified string.

**Note:** The length of an empty string is 0.

Charat( ) The charAt() method returns the character at the specified index in a string.

The index of the first character is 0, the second character is 1, and so on.

Syntax

public char charAt(int *index*)

CompareTo( ):- The compareTo() method compares two strings lexicographically.

The comparison is based on the Unicode value of each character in the strings.

The method returns 0 if the string is equal to the other string. A value less than 0 is returned if the string is less than the other string (less characters) and a value greater than 0 if the string is greater than the other string (more characters).

**Tip:** Use [compareToIgnoreCase()](https://www.w3schools.com/java/ref_string_comparetoignorecase.asp) to compare two strings lexicographyically, ignoring lower case and upper case differences.

**Tip:** Use the [equals()](https://www.w3schools.com/java/ref_string_equals.asp) method to compare two strings without consideration of Unicode values.

## Syntax

public int compareTo(String string2)

public int compareTo(Object object)

[compareToIgnoreCase()](https://www.w3schools.com/java/ref_string_comparetoignorecase.asp)

The compareToIgnoreCase() method compares two strings lexicographically, ignoring lower case and upper case differences.

The comparison is based on the Unicode value of each character in the string converted to lower case.

The method returns 0 if the string is equal to the other string, ignoring case differences. A value less than 0 is returned if the string is less than the other string (less characters) and a value greater than 0 if the string is greater than the other string (more characters).

Syntax

public int compareToIgnoreCase(String *string2*)

Equal():-

The equals() method compares two strings, and returns true if the strings are equal, and false if not.

**Tip:** Use the [compareTo()](https://www.w3schools.com/java/ref_string_compareto.asp) method to compare two strings lexicographically.

## Syntax

public boolean equals(Object anotherObject)

equalsIgnoreCase():-

The equalsIgnoreCase() method compares two strings, ignoring lower case and upper case differences.

This method returns true if the strings are equal, and false if not.

**Tip:** Use the [compareToIgnoreCase()](https://www.w3schools.com/java/ref_string_comparetoignorecase.asp) method to compare two strings lexicographically, ignoring case differences.

## Syntax

public boolean equalsIgnoreCase(String anotherString)

EndsWith():-

The endsWith() method checks whether a string ends with the specified character(s).

**Tip:** Use the [startsWith()](https://www.w3schools.com/java/ref_string_startswith.asp) method to check whether a string **starts** with the specified character(s).

## Syntax

public boolean endsWith(String chars)

IndexOf()

The indexOf() method returns the position of the first occurrence of specified character(s) in a string.

**Tip:** Use the [lastIndexOf](https://www.w3schools.com/java/ref_string_lastindexof.asp) method to return the position of the **last** occurrence of specified character(s) in a string.

## Syntax

startsWith():-

The startsWith() method checks whether a string starts with the specified character(s).

**Tip:** Use the [endsWith()](https://www.w3schools.com/java/ref_string_endswith.asp) method to check whether a string **ends** with the specified character(s).

## Syntax

public boolean startsWith(String chars)

lastIndexOf()

The lastIndexOf() method returns the position of the last occurrence of specified character(s) in a string.

**Tip:** Use the [indexOf](https://www.w3schools.com/java/ref_string_indexof.asp) method to return the position of the **first** occurrence of specified character(s) in a string.

## Syntax

There are 4 lastIndexOf() methods:

public int lastIndexOf(String str)

public int lastIndexOf(String str, int fromIndex)

public int lastIndexOf(int char)

public int lastIndexOf(int char, int fromIndex)

Replace():-

The replace() method searches a string for a specified character, and returns a new string where the specified character(s) are replaced.

Syntax

public String replace(char *searchChar*, char *newChar*)

toLowerCase():-

The toLowerCase() method converts a string to lower case letters.

**Note:** The [toUpperCase()](https://www.w3schools.com/java/ref_string_touppercase.asp) method converts a string to upper case letters.

## Syntax

public String toLowerCase()

UpperCase():-

The toUpperCase() method converts a string to upper case letters.

**Note:** The [toLowerCase()](https://www.w3schools.com/java/ref_string_tolowercase.asp) method converts a string to lower case letters.

## Syntax

public String toUpperCase()

Trim():-

The trim() method removes whitespace from both ends of a string.

**Note:** This method does not change the original string.

Syntax

public String trim()

getChar( ):-

|  |  |
| --- | --- |
| getChars() | Copies characters from a string to an array of chars |

# Immutable String in Java

In java, **string objects are immutable**. Immutable simply means unmodifiable or unchangeable.

Once string object is created its data or state can't be changed but a new string object is created.

Let's try to understand the immutability concept by the example given below: