http://www.tutorialspoint.com/java/java strings.htm

Copyright © tutorialspoint.com

Strings, which are widely used in Java programming, are a sequence of characters. In the Java programming language, strings are objects.

The Java platform provides the String class to create and manipulate strings.

## **Creating Strings:**

The most direct way to create a string is to write:

```
String greeting = "Hello world!";
```

Whenever it encounters a string literal in your code, the compiler creates a String object with its value in this case, "Hello world!'.

As with any other object, you can create String objects by using the new keyword and a constructor. The String class has eleven constructors that allow you to provide the initial value of the string using different sources, such as an array of characters.

```
public class StringDemo{

public static void main(String args[]){
    char[] helloArray = { 'h', 'e', 'l', 'o', '.'};
    String helloString = new String(helloArray);
    System.out.println( helloString );
}
```

This would produce the following result:

```
hello.
```

**Note:** The String class is immutable, so that once it is created a String object cannot be changed. If there is a necessity to make a lot of modifications to Strings of characters, then you should use <a href="String Buffer & String Builder">String Builder</a> Classes.

## **String Length:**

Methods used to obtain information about an object are known as accessor methods. One accessor method that you can use with strings is the length method, which returns the number of characters contained in the string object.

Below given program is an example of **length**, method String class.

```
public class StringDemo {

   public static void main(String args[]) {
      String palindrome = "Dot saw I was Tod";
      int len = palindrome.length();
      System.out.println( "String Length is : " + len );
   }
}
```

This would produce the following result:

```
String Length is : 17
```

# **Concatenating Strings:**

The String class includes a method for concatenating two strings:

```
string1.concat(string2);
```

This returns a new string that is string1 with string2 added to it at the end. You can also use the concat method with string literals, as in:

```
"My name is ".concat("Zara");
```

Strings are more commonly concatenated with the + operator, as in:

```
"Hello," + " world" + "!"
```

which results in:

```
"Hello, world!"
```

Let us look at the following example:

```
public class StringDemo {
   public static void main(String args[]) {
       String string1 = "saw I was ";
       System.out.println("Dot " + string1 + "Tod");
   }
}
```

This would produce the following result:

```
Dot saw I was Tod
```

## **Creating Format Strings:**

You have printf and format methods to print output with formatted numbers. The String class has an equivalent class method, format, that returns a String object rather than a PrintStream object.

Using String's static format method allows you to create a formatted string that you can reuse, as opposed to a one-time print statement. For example, instead of:

you can write:

# **String Methods:**

Here is the list of methods supported by String class:

### SN Methods with Description

1

char charAtintindex

Returns the character at the specified index.

2 int compareToObjecto

Compares this String to another Object.

3 <u>int compareToStringanotherString</u>

Compares two strings lexicographically.

4 <u>int compareTolqnoreCaseStringstr</u>

Compares two strings lexicographically, ignoring case differences.

5 String concatStringstr

Concatenates the specified string to the end of this string.

6 <u>boolean contentEqualsStringBuffersb</u>

Returns true if and only if this String represents the same sequence of characters as the specified StringBuffer.

7 <u>static String copyValueOfchar[]data</u>

Returns a String that represents the character sequence in the array specified.

8 static String copyValueOfchar[]data, intoffset, intcount

Returns a String that represents the character sequence in the array specified.

9 <u>boolean endsWithStringsuffix</u>

Tests if this string ends with the specified suffix.

10 <u>boolean equalsObjectanObject</u>

Compares this string to the specified object.

boolean equalsIgnoreCaseStringanotherString

Compares this String to another String, ignoring case considerations.

<u>byte getBytes</u>

Encodes this String into a sequence of bytes using the platform's default charset, storing the result into a new byte array.

11

12

### byte[] getBytes(String charsetName

Encodes this String into a sequence of bytes using the named charset, storing the result into a new byte array.

14

#### void getCharsintsrcBegin, intsrcEnd, char[]dst, intdstBegin

Copies characters from this string into the destination character array.

15

### int hashCode

Returns a hash code for this string.

16

### int indexOfintch

Returns the index within this string of the first occurrence of the specified character.

17

### int indexOfintch, intfromIndex

Returns the index within this string of the first occurrence of the specified character, starting the search at the specified index.

18

### int indexOfStringstr

Returns the index within this string of the first occurrence of the specified substring.

19

#### int indexOfStringstr, intfromIndex

Returns the index within this string of the first occurrence of the specified substring, starting at the specified index

20

#### String intern

Returns a canonical representation for the string object.

21

### int lastIndexOfintch

Returns the index within this string of the last occurrence of the specified character.

22

#### int lastIndexOfintch, intfromIndex

Returns the index within this string of the last occurrence of the specified character, searching backward starting at the specified index.

23

### int lastIndexOfStringstr

Returns the index within this string of the rightmost occurrence of the specified substring.

#### int lastIndexOfStringstr, intfromIndex

Returns the index within this string of the last occurrence of the specified substring, searching backward starting at the specified index.

25

#### int length

Returns the length of this string.

26

#### boolean matchesStringregex

Tells whether or not this string matches the given regular expression.

27

### boolean regionMatchesbooleanignoreCase, inttoffset, Stringother, intooffset, intlen

Tests if two string regions are equal.

28

### boolean regionMatchesinttoffset, Stringother, intooffset, intlen

Tests if two string regions are equal

29

### String replacecharoldChar, charnewChar

Returns a new string resulting from replacing all occurrences of oldChar in this string with newChar.

30

#### String replaceAll(String regex, String replacement

Replaces each substring of this string that matches the given regular expression with the given replacement.

31

### String replaceFirstStringregex, Stringreplacement

Replaces the first substring of this string that matches the given regular expression with the given replacement.

32

### String[] splitStringregex

Splits this string around matches of the given regular expression.

33

### String[] splitStringregex, intlimit

Splits this string around matches of the given regular expression.

34

#### boolean startsWithStringprefix

Tests if this string starts with the specified prefix.

### boolean startsWithStringprefix, inttoffset

Tests if this string starts with the specified prefix beginning a specified index.

36 CharSequence subSequenceintbeginIndex, intendIndex

Returns a new character sequence that is a subsequence of this sequence.

37 <u>String substringintbeginIndex</u>

Returns a new string that is a substring of this string.

38 String substringintbeginIndex, intendIndex

Returns a new string that is a substring of this string.

39 <u>char[] toCharArray</u>

Converts this string to a new character array.

40 <u>String toLowerCase</u>

Converts all of the characters in this String to lower case using the rules of the default locale.

41 String toLowerCaseLocalelocale

Converts all of the characters in this String to lower case using the rules of the given Locale.

42 String toString

This object which is already astring! is itself returned.

43 String toUpperCase

Converts all of the characters in this String to upper case using the rules of the default locale.

44 String to Upper Case Localelocale

Converts all of the characters in this String to upper case using the rules of the given Locale.

45 <u>String trim</u>

Returns a copy of the string, with leading and trailing whitespace omitted.

# static String valueOfprimitivedatatypex

Returns the string representation of the passed data type argument.

Loading [MathJax]/jax/output/HTML-CSS/jax.js