



STRING

CDAC Patna

String

- A string is sequence of characters for e.g. “Welcome” is a string of 7 characters.
- In Java , strings are used as an object. The Java provides the **String** class to create string and also provides a lot of methods to perform operations on strings such as compare(), concat(), equals(), split(), length(), replace(), etc.
- When we create a string in java, it actually creates an object of type String.
- String is immutable class which means that it cannot be changed once it is created.

- **Creating a String:-** We can create strings in Java by using these three classes. (String , StringBuffer and StringBuilder)

The Java String is immutable which means it cannot be changed. So for mutable strings, you can use StringBuffer and StringBuilder classes.

1. String literal

2. Using new keyword

- String literal:- Here String is created by using double quotes (" ").

For Example- String course="java";

- Using New Keyword:- We can create String object using new operator, just like any normal java class.

Example- String course = **new** String("Java");

- We can also create array of string


Example- String str[]= new String[3];

str[0]="Mohan"


str[1]="Ramesh"

str[2]="Kumar"


0	1	2
Mohan	Ramesh	Kumar



```
public class String1 {  
  
    public static void main(String[] args) {  
        // creating string using string literal  
        String str1="Notebook";  
        String str2 ="Notebook"; // it doesn't create a new object  
  
        if(str1==str2)  
            System.out.println("str1 and str2 has same reference/address ");  
        else  
            System.out.println("str1 and str2 has not same reference/address");  
  
        //creating strings using new operator  
        String str3 = new String("Book");  
        String str4 = new String("Book");// But it create a new object  
        if(str3==str4)  
            System.out.println("str3 and str4 has same  reference/address");  
        else  
            System.out.println("str3 and str4 has not same reference/address ");  
  
    }  
  
}
```

- 
- **String Class Methods :-** There are various String class methods used to perform operations on strings.

Methods	Description
length()	returns length of given string
toUpperCase()	returns a string in uppercase.
toLowerCase()	returns a string in lowercase.
concat()	this method combines a specific string at the end of another string and returns a combined string
charAt()	This method is used to retrieve a single character from a given String.
contains()	It is used to determine whether a substring is a part of the given main String or not and it return Boolean(True/False).
replace()	this method is used to replace the character with the new characters in a String.
trim()	It remove the spaces from the beginning and end of the given string.
substring()	returns the substring of the string




<code>compareTo()</code>	This method is used to compare two Strings..
<code>compareToIgnoreCase()</code>	Same as Compare To method however it ignores the case during comparison.
<code>isEmpty()</code>	It checks if string is empty.
<code>equals()</code>	Compares the string with the specified string and returns true if both matches else false.
<code>equalsIgnoreCase()</code>	It works same as equals method but it doesn't consider the case while comparing strings.
<code>indexOf()</code>	returns the position of the specified character in the string
<code>startsWith()</code>	checks if the string begins with the given string
<code>endsWith()</code>	checks if the string ends with the given string

```
public class String1 {  
    public static void main(String[] args) {  
        // creating java string by new keyword  
        String s1 = new String("Java Programming");  
        String s2 = new String(); // create a empty string  
        String s3 = new String("Computer s cien ce ");  
  
        //length of string  
        System.out.println(s1.length());  
        //concatenate two string  
        System.out.println(s1.concat(s3));  
        //compare the strings  
        System.out.println(s1.compareTo(s3)); // 7(j is greater than c)  
        //isEmpty function  
        System.out.println(s2.isEmpty());  
        //remove spaces from the given string  
        System.out.println(s3.trim());  
        // to upper case  
        System.out.println(s1.toUpperCase());  
        //to lower case  
        System.out.println(s1.toLowerCase());  
        //replace()  
        System.out.println(s1.replace('r', 't'));  
    }  
}
```

- **Note-** we can also perform Concatenation of strings using “+” symbol. It also join two or more strings. ‘+’ operator for both addition and concatenation. Numbers are added and strings are concatenated.

```
public class String1 {  
  
    public static void main(String[] args) {  
        // creating string using string literal  
        String str1="Software";  
        String str2 ="application";  
  
        //Method 1:using concat()  
        String str3 =str1.concat(str2);  
        System.out.println(str3);  
  
        //Method 1:using +  
        String str4 =str1+str2;  
        System.out.println(str4);  
  
    }  
}
```


- 
- **NOTE-** If the expression begins with a string and uses the + operator , then the next argument is converted to a string.
 - Strings cannot be compared with == and !=

CDAC Patna



➡ **StringBuffer and StringBuilder class** – Both classes are used to create the string object but it is a mutable class which means that the strings passed through this can be changed as per requirement.

- The StringBuffer and StringBuilder class in Java is the same as String class except it is mutable i.e. it can be changed. But String class object can't be changed.
- **Important methods :-**

Methods	Descriptions
append()	It is used to append the specified string with this given string (but at the end)
insert()	It is used to insert the specified string with given string at the specified no. of position.
replace()	It is used to replace the string from specified start Index value and end Index value.
delete()	It is used to delete the string from specified start Index value and end Index value.
reverse()	used to reverse the given string.

Example-

```
public class String1 {  
  
    public static void main(String[] args) {  
        // creating string using string buffer class  
        StringBuffer sb = new StringBuffer("java");  
  
        //now the content of the string is changed  
        sb.append(" Programming");  
        System.out.println(sb);  
  
        //delete the substring from given string  
        System.out.println(sb.delete(1, 3));  
  
        //reverse the string  
        System.out.println(sb.reverse());  
    }  
}
```

- A string is an object that represents a number of character values. Each letter in the string is a separate character .
- An array of characters works same as Java string.
- For example:

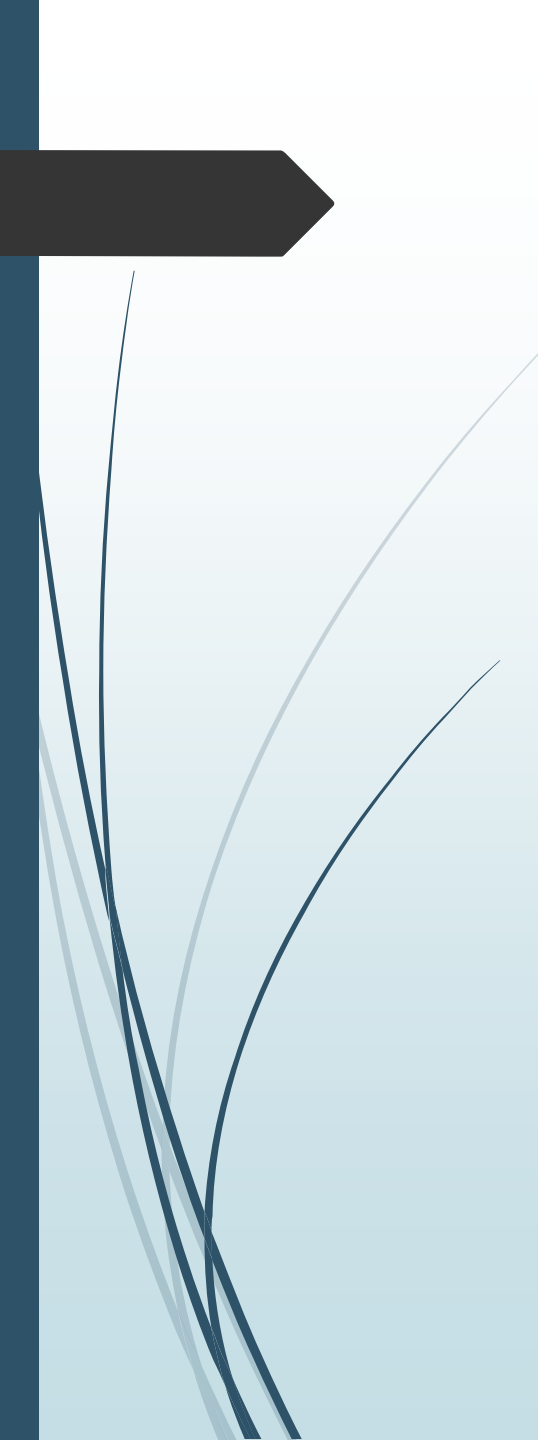
```
char[] ch = { 't', 'e', 'c', 'h', 'n', 'i', 'c', 'a', 'l' };
```

```
String s = new String(ch);
```

is same as:

```
String s = "technical";
```

CDAC Patna



```
public class String1 {  
  
    public static void main(String[] args) {  
        // creating string by string literal  
        String s1="Programming";  
  
        //converting char array to string  
        char ch[]= {'s','t','r','o','n','g'};  
        String s2 = new String(ch);  
  
        // creating java string by new keyword  
        String s3 = new String("Computer");  
  
        System.out.println(s1);  
        System.out.println(s2);  
        System.out.println(s3);  
    }  
}
```