Methods of String class

length()
length(): This method is used to get the number of characters of any string.

Example
```
class StringHandling
{
public static void main(String arg[])
{
int l;
String s=new String("Java");
l=s.length();
System.out.println("Length: "+l);
}
}
```
Output
Length: 4

charAt(index)

charAt(): This method is used to get the character at a given index value.

Example
```
class StringHandling
{
public static void main(String arg[])
{
char c;
String s=new String("Java");
c=s.charAt(2);
System.out.println("Character: "+c);
}
}
```
Output
Character: v

toUpperCase()

toUpperCase(): This method is use to convert lower case string into upper case.

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s="Java";
System.out.println("String: "+s.toUpperCase());
}
}
```
Output
String: JAVA

toLowerCase()

toLowerCase(): This method is used to convert upper case string into lower case.

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s="JAVA";
System.out.println("String: "+s.toLowerCase());
}
}
```
Output
String: java

concat()
concat(): This method is used to combine two strings.

Example
```
class StringHandling

{
public static void main(String arg[])
{
String s1="Hitesh";
String s2="Reddy";
System.out.println("Combined String: "+s1.concat(s2));
}
}
```
Output
Combined String: HiteshReddy
startsWith()

startsWith(): This method returns true if string starts with given another string, otherwise it returns false.

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s="Java is programming language";
System.out.println(s.startsWith("Java"));
}
}
```
Output
true

endsWith()

endsWith(): This method returns true if string ends with given another string, otherwise it returns false.

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s="Java is programming language";
System.out.println(s.endsWith("language"));
}
}
```
Output
true

substring()

substring(): This method is used to get the part of given string.

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s="Java is programming language";
System.out.println(s.substring(8)); // 8 is starting index
}
}
```
Output
programming language

Example2
```
class StringHandling
{
public static void main(String arg[])
{
String s="Java is programming language";
System.out.println(s.substring(8, 12));
}
}
```
Output
prog

indexOf()
indexOf(): This method is used find the index value of given string. It always gives starting index value of first occurrence of string.

Example
```
class StringHandling
{
public static void main(String arg[])
{
```

```
String s="Java is programming language";
System.out.println(s.indexOf("programming"));
}
}
```
Output
8

lastIndexOf()

lastIndexOf(): This method used to return the starting index value of last occurence of the given string.

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s1="Java is programming language";
String s2="Java is programming language programming language";
System.out.println(s1.indexOf("programming"));
System.out.println(s2.lastIndexOf("programming"));
}
}
```
Output
8
29

trim()
trim(): This method remove space which are available before starting of string and after ending of string.

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s="    Java is programming language    ";
System.out.println(s.trim());
}
}
```
Output
Java is programming language

split()

split(): This method is used to divide the given string into number of parts based on delimiter (special symbols like @ space ,).

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s="contact@srecwarangal.ac.in";
```

```
String[] s1=s.split("@"); // divide string based on @
for(String c:s1) //  foreach loop
{
System.out.println(c);
}
}
}
```

Output

```
contact
@srecwarangal.ac.in
```

replace()

replace(): This method is used to return a duplicate string by replacing old character with new character.
Note: In this method data of original string will never be modify.

Example

```
class StringHandling
{
public static void main(String arg[])
{
String s1="java";
String s2=s1.replace('j', 'k');
System.out.println(s2);
}
}
```

Output

```
kava
```

String Comparison

String comparison can be done in 3 ways.

1. Using equals() method
2. Using == operator
3. By CompareTo() method

Using equals() method

equals() method compares two strings for equality. Its general syntax is,

```
boolean equals (Object str)
```

It compares the content of the strings. It will return true if string matches, else returns false.

```
String s = "Hell";
String s1 = "Hello";
String s2 = "Hello";
s1.equals(s2);    //true
s.equals(s1) ;  //false
```

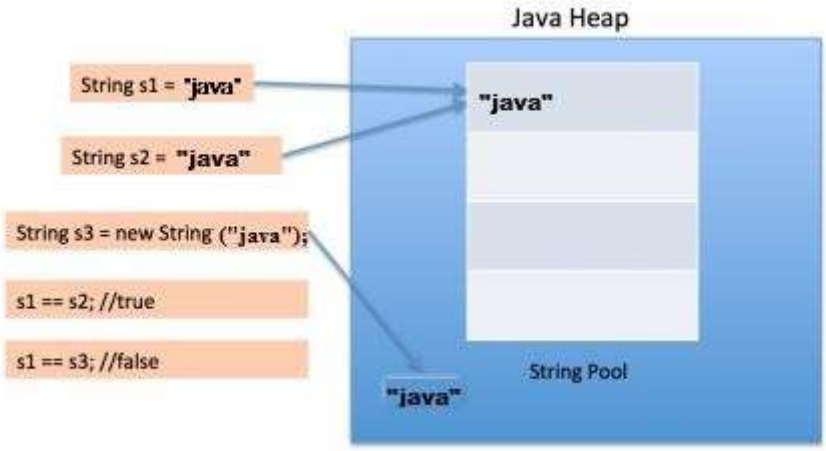
Using == operator

== operator compares two object references to check whether they refer to same instance. This also, will return true on successful match.

```
String s1 = "Java";
String s2 = "Java";
```

String s3 = new string ("Java");
test(s1 == s2) //true
test(s1 == s3) //false

Reason:
Its because we are creating a new object using new operator, and thus it gets created in a non-pool memory area of the heap. s1 is pointing to the String in string pool while s3 is pointing to the String in heap and hence, when we compare s1 and s3, the answer is false.
The following image will explain it more clearly.



By compareTo() method
compareTo() method compares values and returns an int which tells if the string compared is less than, equal to or greater than the other string. It compares the String based on natural ordering i.e alphabetically. Its general syntax is,

int compareTo(String str)

String s1 = "Abhi";
String s2 = "Viraaj";
String s3 = "Abhi";
s1.compareTo(S2); //return -1 because s1 < s2
s1.compareTo(S3); //return 0 because s1 == s3
s2.compareTo(s1); //return 1 because s2 > s1

equalsIgnoreCase()
equalsIgnoreCase(): This method is case insensitive method, It return true if the contents of both strings are same otherwise false.

Example
```
class StringHandling
{
public static void main(String arg[])
{
String s1="Hitesh";
String s2="HITESH";
String s3="Raddy";
System.out.println("Compare String: "+s1.equalsIgnoreCase(s2));
System.out.println("Compare String: "+s1.equalsIgnoreCase(s3));
}
}
```

Output
Compare String: true
Compare String: false
compareToIgnoreCase()
compareToIgnoreCase(): This method is case insensitive method, which is used to compare two strings similar to compareTo().

Example

```
class StringHandling
{
public static void main(String arg[])
{
String s1="Hitesh";
String s2="HITESH";
int i;
i=s1.compareToIgnoreCase(s2);
if(i==0)
{
System.out.println("Strings are same");
}
else
{
System.out.println("Strings are not same");
}
}
}
```

Output

Strings are same