String handling:

- It plays a very important role. Most of the data that transmits on internet will be in the form of String (group of characters).
- In java it is not a character array terminated by NULL ('\0') operator like C and C++. Its an object of String class.
- Represented by the java.lang.String class
- String characteristics
  - Reference type
  - Immutable
  - final
- Each character is represented by java.lang.Char (char)
  - Uses UTF-16 encoding

Creating Strings

- Using a String literal
- Using a Constructor
- Calling toString( ) on another object.

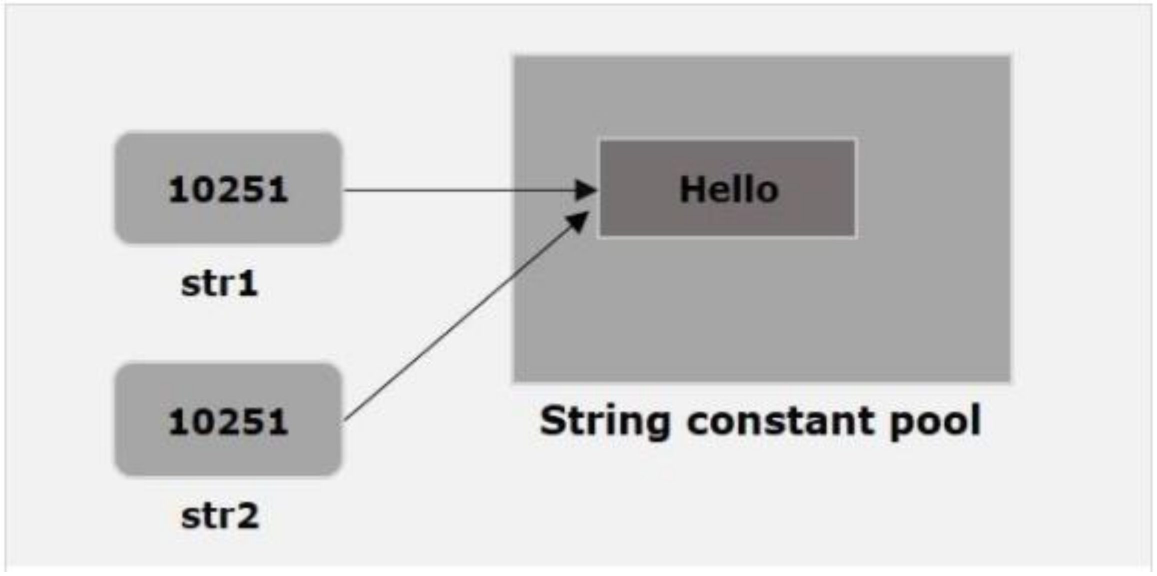
```
String str ;
str   = “Hello”;                    (or)
String  str  = “Hello”;
```

```
String str = new String( “Hello” );
```

```
char arr [ ] = { ‘C’,’O’,’M’,’P’,’U’,’T’,’E’,’R’ };
String str = new String( arr );
String str = new String( arr , 2 , 4);
```

String constant pool

- JVM creates a String object with the given value in a separate block of memory known as String constant pool.



String Length

- Accessible through the length ( ) method, which returns the number of characters contained in the String object.

```
public class StringDemo {
    public static void main ( String args[ ] ) {
        String str = “VITAP” ;
        int  len = str.length();
        System.out.println( "String Length is : " + len );
    }
}
```

String Concatenation

- Java Strings can be concatenated (joined) using the **+** and **+=** operators to create new Strings.
- Every time an operation modifies a String object, a new read-only String object is created.
- Using **concat ( )** method

```
String language = " Java ";
String course = "Introduction to " + language ;
course += " Programming ";
System.out.println( course );      // Introduction to Java Programming
```

```
String  str1 = " Hello ";
String  str2 = " World ";
String  str3 = str1.concat(str2);
System.out.println(str3) ;          //  Hello World
```

Comparing Strings

- Strings are compared to determine equality and for sorting
- Java provides variety of methods to compare String objects
- Use of **= =** operator only tests whether two String object references are same or not

Method	Description
int compareTo (String)	Compares two strings lexicographically and returns int value (0 , >0 , <0)
int compareToIgnoreCase (String)	Compares two strings, ignoring case differences.
boolean equals (String)	Compares two strings and returns true or false.
boolean equalsIgnoreCase (String)	Compares two strings, ignoring case differences.

String Comparison Methods

Method	Description
boolean startsWith (String prefix)	Tests whether the current string starts with specified prefix or not
boolean endsWith (String suffix)	Tests whether the current string ends with specified suffix or not
int indexOf (String)	Returns the index of the first occurrence of the specified string.
int lastIndexOf (String)	Returns the index of the last occurrence of the specified string, searching backward.

String Manipulation Methods

Method	Description
String toLowerCase ( ) String toUpperCase ( )	Transforms the String into either upper or lower case
String replace (char old , char new)	Replaces old character to new character
String substring (int beginIndex) String substring (int beginIndex, int endIndex)	Returns to the index of the string to end.
String [ ] split (String regex)	Splits the current string in to string array.
String trim ( )	Returns the string, with leading and trailing whitespace omitted.
char charAt (int index)	Returns the character at the specified index.

String Manipulation Methods

Usage	Prints
String str = "Hello world!";	
System.out.println(str.toLowerCase());	hello world!
System.out.println(str.toUpperCase());	HELLO WORLD!
str.replace("Hello", "Good morning");	Good morning world!
System.out.println(str.substring(0, 5));	Hello

String list = "1,2,3,4,5";
String [ ] listItems = list.split(',');
for (String item : listItems) {
    System.out.println(item);
}
	1
	2
	3
	4
	5

StringBuilder or StringBuffer

- String is immutable.
- StringBuilder and StringBuffer is mutable.
- StringBuilder is not always more efficient than string.
- These classes provide methods which can modify the content of the objects directly.
- Some methods insert(), delete() and reverse() which are not available in String class.
- StringBuffer class will take more execution time than the StringBuilder.

StringBuilder Methods

Method	Description
StringBuilder append (String)	It is used to append the specified string with this string.
void insert (int index , String)	It is used to insert the specified string with this string at the specified position.
void reverse ( )	It is used to reverse the string.
void delete (int start , int end)	It is used to delete the string from specified startIndex and endIndex.
void setCharAt (int index , char ch)	Replace char at index position
String toString ( )	The toString() method returns the String representation of the object.

The main difference between the StringBuffer and StringBuilder is that StringBuilder methods are not thread safe(not Synchronized).