STRING

A string is an object that represents a sequence of characters

A string is actually a final class which is defined in java.lang package

Strings are immutable. By immutable, we mean that Strings are constant, their values cannot be changed after they are created.

HOW TO CREATE STRING OBJECT

1.By string literal: Using double quotes

Example: String str = “Joyful”;

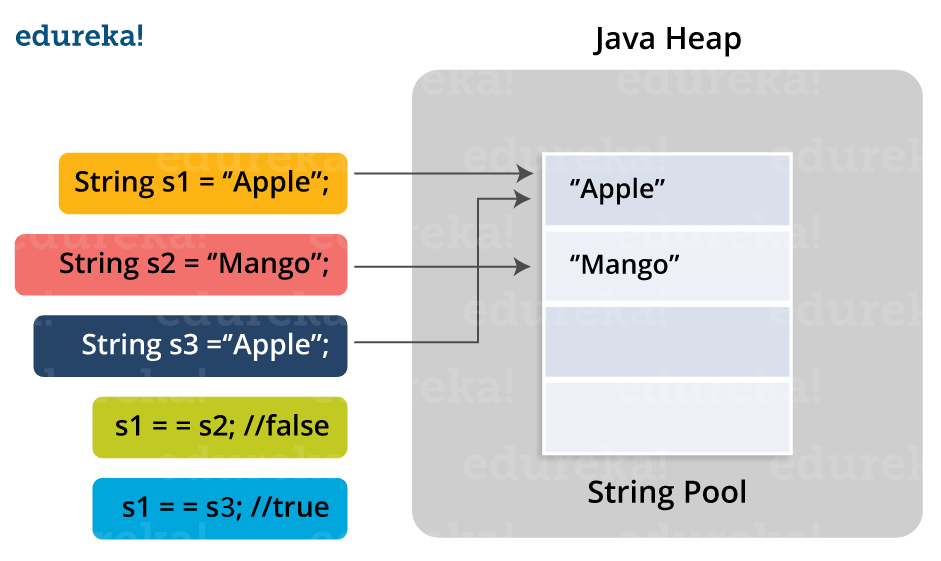
2.By new Keyword: Using new keyword

Example: String str = new String(“Joyful”);

**Java String Pool:**

Java String pool refers to collection of Strings which are stored in heap memory. It is maintained by the JVM.

In this, whenever a new object is created, String pool first checks whether the object is already present in the pool or not. If it is present, then same reference is returned to the variable else new object will be created in the String pool and the respective reference will be returned.



**JAVA STRING METHODS**

**Java String length()**: The Java String length() method tells the length of the string.

String s1="hello";

System.out.println(s1.length());

**String compareTo()**: The Java String compareTo() method compares the given string with current string.

String s1="hello";

String s2="hello";

System.out.println(s1.compareTo(s2)); // 0 because both are equal

if s1 > s2, it returns a positive number    
if s1 < s2, it returns a negative number   
if s1 == s2, it returns 0

**String concat() :**The Java String concat() method combines a specific string at the end of another string and ultimately returns a combined string.

String s1="hello";

s1=s1.concat("how are you");

System.out.println(s1);

**String IsEmpty()** : This method checks whether the String contains anything or not.

String s1="";

String s2="hello";

System.out.println(s1.isEmpty());      // true

System.out.println(s2.isEmpty());      // false

**String Trim()** : The java string trim() method removes the leading and trailing spaces.

String s1="  hello  ";

System.out.println(s1.trim());

**String toLowerCase()**: The java string toLowerCase() method converts all the characters of the String to lower case.

**String toUpperCase()**: The java string toUpperCase() method converts all the characters of the String to Upper case.

String s1="HELLO HOW Are You?”;

String s1lower=s1.toLowerCase();

System.out.println(s1lower);

**String replace()**: The Java String replace() method returns a string, replacing all the old characters or CharSequence to new characters.

String s1="hello how are you";

String replaceString=s1.replace('h','t');

**String replace(CharSequence target, CharSequence replacement) method**:

String s1="Hey, welcome to Edureka";

String replaceString=s1.replace("Edureka","Brainforce");

**String contains()** :The java string contains() method searches the sequence of characters in the string.

String name=" hello how are you doing";

System.out.println(name.contains("hello"));        // returns true

**String equals()** : The Java String equals() method compares the two given strings on the basis of content of the string

String s1="hello";

String s2="hello";

System.out.println(s1.equals(s2));   // returns true

**String equalsIgnoreCase():**This method compares two string on the basis of content but it does not check the case like equals() method.

System.out.println(s1.equalsIgnoreCase(s2));   // returns true

**String toCharArray():**This method converts the string into a character array i.e first it will calculate the length of the given Java String including spaces and then create an array of char type with the same content.

String s1="Welcome to Edureka";

**char**[] ch=s1.toCharArray();

**for**(**int** i=0;i<ch.length;i++){

System.out.print(ch[i]);

**String endsWith()** : The Java String endsWith() method checks if this string ends with the given suffix.

String s1="hello how are you”;

System.out.println(s1.endsWith("u"));       // returns true

System.out.println(s1.endsWith("you"));     // returns true

**String split()**

The split() method divides the string at the specified regex and returns an array of substrings.

**Note:** If the regular expression passed to split() is invalid, the split() method raises PatternSyntaxExpression exception.

Syntax: split(regex) or split(regex,limit);

Char charAt(int index);

Returns the character at the specified index.

Syntax: charAt(int index);

String copyValueOf(char[] data)

Syntax: copyValueOf(char[] data)

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

Int indexOf(int ch)

Syntax: indexOf(int ch);

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

JAVA PROVIDES TWO UTILITY CLASSES FOR STRING:

StringBuffer and StringBuilder-- StringBuffer and StringBuilder are mutable classes.They are used for String Manipulation

* .StringBuffer operations are thread-safe and synchronized whereas StringBuilder operations are not thread-safe.
* StringBuffer is to be used in multi threaded application and StringBuilder in the single threaded environment.
* StringBuilder performance is faster when compared to StringBuffer because of no overhead of synchronized.
* StringBuffer and StringBuilder are final classes ie., these classes will not support inheritance
* StringBuffer and StringBuilder objects can be created only using new keyword and not by literals
* StringBuffer and StringBuilder objects are mutable ie., object can be changed in the same memory address
* Both supports special functions like append ,insert, delete and reverse which are not supported in string class

SYNTAX:

StringBuffer sbfr = new StringBuffer(“Grace and peace”);

StringBuilder strbldr = new StringBuilder(“Grace and peace”);