Explore | Expand | Enrich

TEST TIME ON OBJECT CLASS METHODS

URL: https://forms.gle/DLskMHTLcYfiAbyZ9

QR CODE:

Strings

What you'll Learn

- String
- String Methods
- Stringbuilder

String

In Java, string is basically an object that represents sequence of char values.

An array of characters works same as Java string.

```
char[ ]  ch = {'e','t','h','n','u','s'};
```

Syntax

```
<String_Type> <string_variable> = “<sequence_of_string>”;
String str="Hello!";
```

Ways to Create String

Using String literal

You can create String objects with String literal

```
String s1="Welcome";
String s2="Welcome";  //It doesn't create a new instance
```

Using new keyword

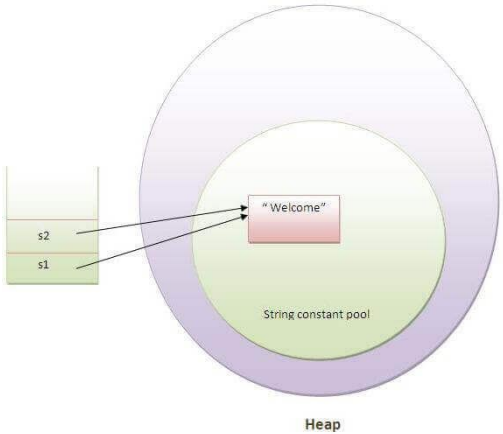
This is the common way to create a String object in java.

```
String str1= new String("Hello!");
```

Using character array:

You could also convert character array into String here

```
char ch[]={ 'H','e','l','l','o','!'};
```



Memory Allocation

String Object is created, two objects will be created

- The Heap Area
- The String constant pool

The String object reference always points to heap area object.

Example: String str = “Ethnus”;

0	1	2	3	4	5	6
E	t	h	n	u	s	\0
0x23452	0x23453	0x23454	0x23455	0x23456	0x23457	0x23458

String class constructors

1. String s1 = new String( );
2. String s2 = new String(String literal);
3. char[] ch1 = {‘H’, ‘e’, ‘l’, ‘l’, ‘o’};

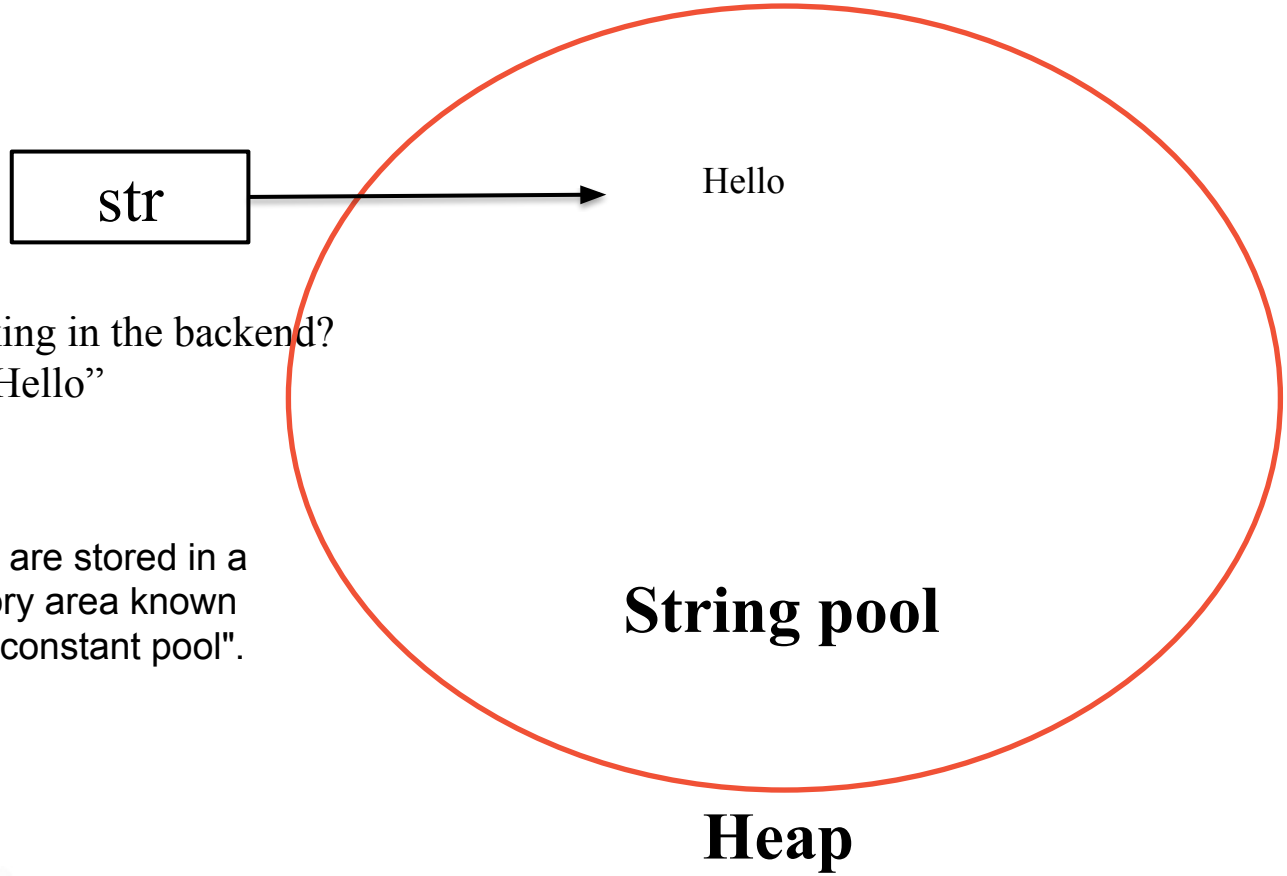
   String s3 = new String(ch1);

PROGRAM

```
// Predict the output

class Main{
    public static void main(String args[]){
        String str = "Programming";
        String s = new String("World");
        System.out.println(str);
        System.out.println(s);
    }
}
```

String Literal - String str = “Hello”



How it is working in the backend?
String str1 = “Hello”

String objects are stored in a special memory area known as the "string constant pool".

String Literal - String str = “Hello”

working in the backend?

Heap

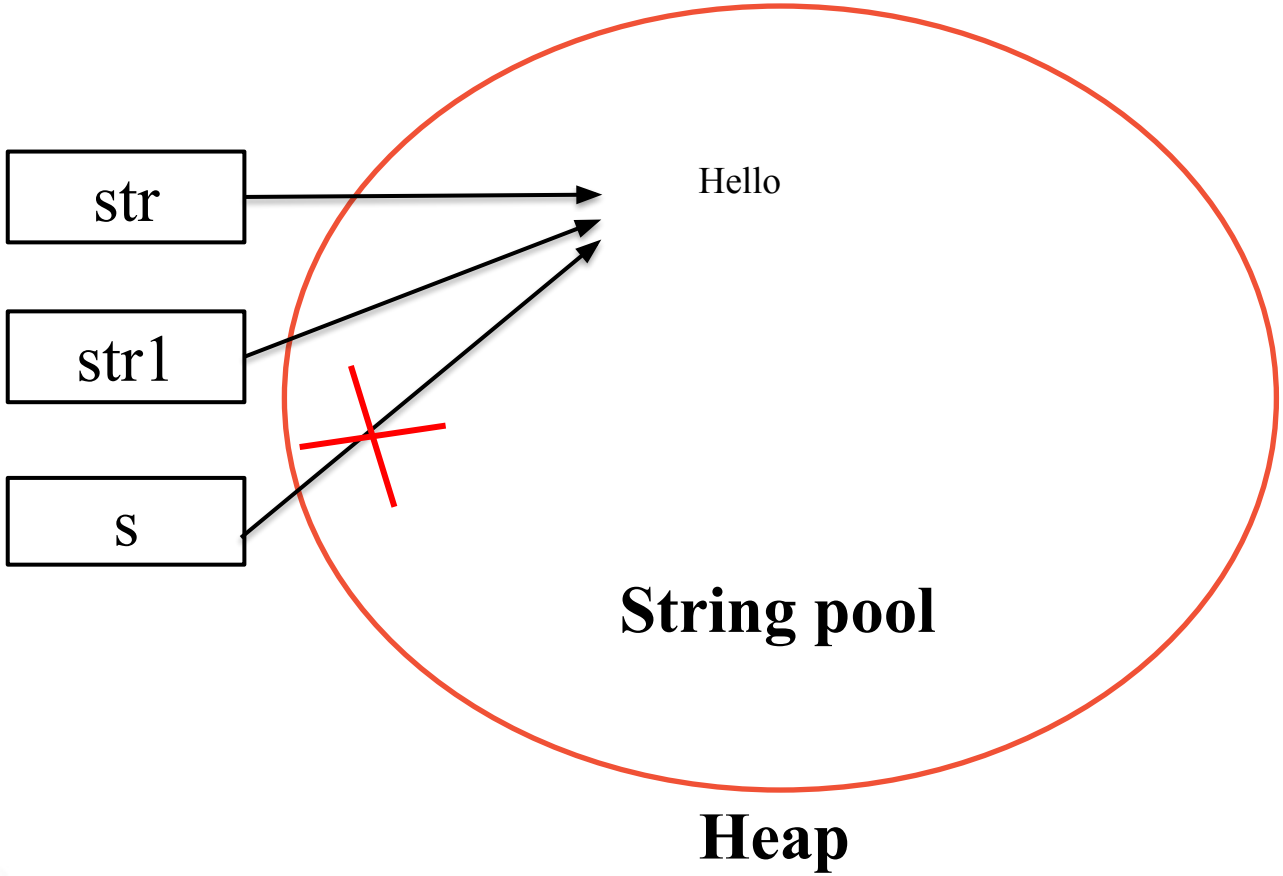
If we create an object using String literal. It may return an existing object from the String pool, if it already exists.
Otherwise, it will create a new String object and put in the string pool for future re-use.

The same reference will be assigned to str and str1.

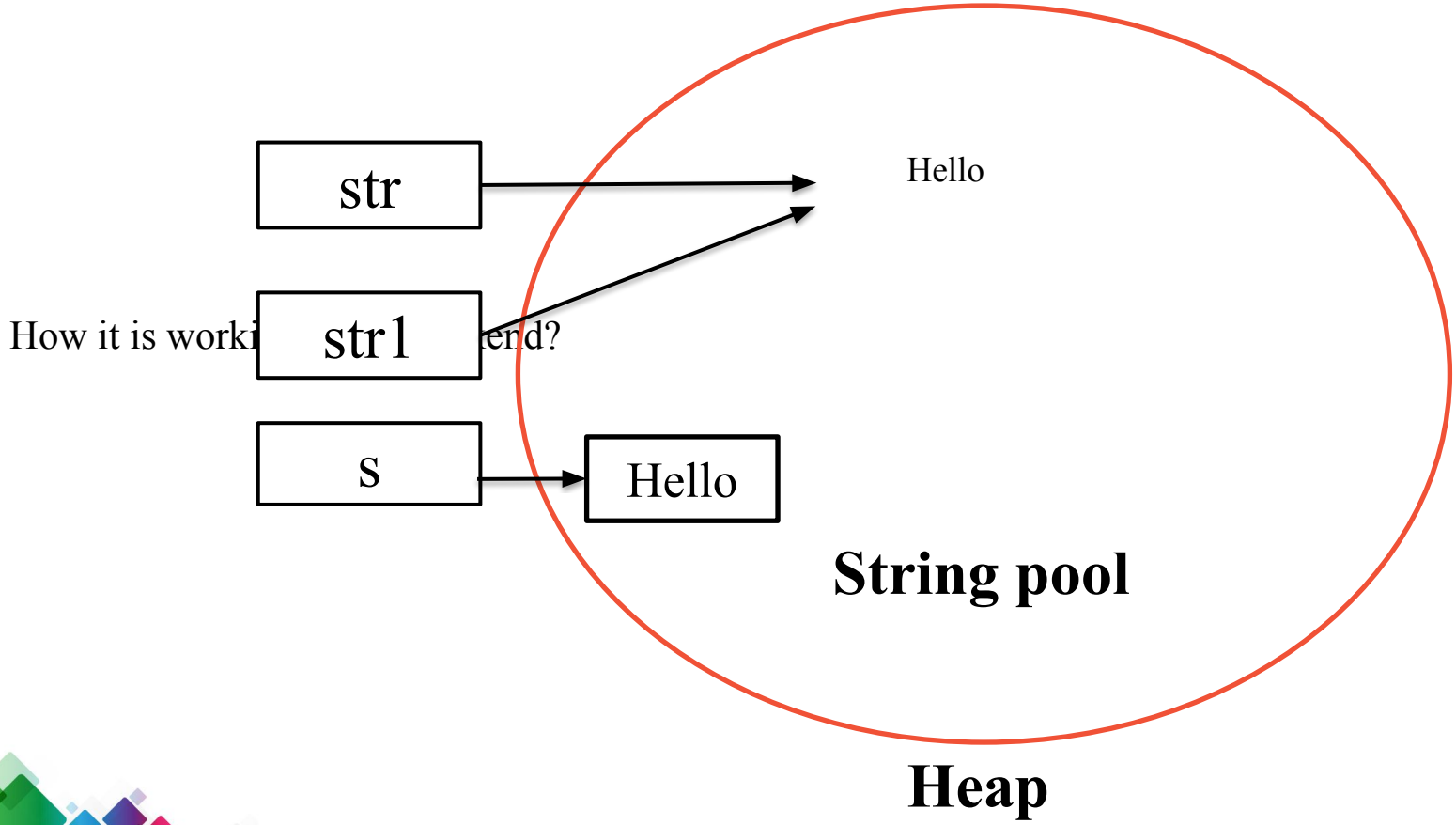
Why Java uses the concept of String literal?

To make Java more memory efficient (because no new objects are created if it exists already in the string constant pool).

new - String s = new String("Hello")



new - String s = new String("Hello")



```
// Predict the output of the below code

class Main{
    public static void main(String args[]){
        String str = "Hello";
        String str1 = "Hello";
        String str2 = new String("Hello");
        System.out.println(str == str1);
        System.out.println(str == str2);
    }
}
```

```
true
false
```

```
// Predict the output of the below code

class Main{
    public static void main(String args[]){
        String str1 = "Hello";
        String str2 = "Hello";
        String s1 = new String("Hello");
        String s2 = new String("Hello");
        System.out.println(str1 == str2);
        System.out.println(str2 == s1);
        System.out.println(s1 == s2);
    }
}
```

```
true
false
false
```

```
// Predict the output of the below code

class Main{
    public static void main(String args[]){
        String s1 = new String("Noughat");
        String s2 = new String("Noughat");
        System.out.println(s1 == s2);
        System.out.println(s1.equals(s2));
    }
}
```

```
false
true
```

String Methods

<Codemithra /> ™

ETHNUS ™

Explore | Expand | Enrich

Method	Description
char charAt(int index)	returns char value for the particular index
int length()	returns string length
String substring(int beginIndex)	returns substring for given begin index.
boolean equals(Object another)	checks the equality of string with the given object.
String concat(String str)	concatenates the specified string.
String replace(char old, char new)	replaces all occurrences of the specified char value.
int indexOf(int ch)	returns the specified char value index.

charAt() - Example

<Codemithra /> ™

ETHNUS ™

Explore | Expand | Enrich

The **java string charAt()** method returns a char value at the given index number

```
class Main {
  public static void main(String args[]){
  String name="codemithra";
   char ch=name.charAt(4);
   System.out.println(ch);
  }
}
```

length() - Example

The **java string length()** method length of the string. It returns count of total number of characters

```
class Main {
   public static void main(String args[]){
   String s1="ETHNUS";
   String s2="Codemithra.com";
   System.out.println("string length is: "+s1.length());//6 is
the length of ETHNUS string
  System.out.println("string length is: "+s2.length());
  //14 is the length of Codemithra.com string


   }}
```

subString() - Example

<Codemithra /> ™

ETHNUS ™

Explore | Expand | Enrich

The **java string substring()** method returns a part of the string

```
public class Main {
   public static void main(String[] args) {
        String s1="Ethnus and Codemithra";
        String substr = s1.substring(0);// Starts with 0 and goes
to end
       System.out.println(substr);
        String substr2 = s1.substring(5,10);// Starts from 5 and
goes to 10
      System.out.println(substr2);
   }
}
```

concat() - Example

The **java string concat()** method combines specified string

```
public class Main{
  public static void main(String args[]){
    String s1="Today you";
    System.out.println(s1);
    s1=s1.concat(" will learn Java String");
    System.out.println(s1);
  }
}
```

contains() - Example

<Codemithra /> ™

ETHNUS ™

Explore | Expand | Enrich

The **java string contains()** method searches the sequence of characters in this string

```
class Main{
  public static void main(String args[]){
    String name="what do you know about me";
    System.out.println(name.contains("do you know"));
    System.out.println(name.contains("about"));
    System.out.println(name.contains("hello"));
  }
}
```

endsWith - Example

The **java string endsWith()** method checks if this string ends with given suffix

```
class Main{
  public static void main(String args[]){
    String s1="java by ethnus";
    System.out.println(s1.endsWith("s"));
    System.out.println(s1.endsWith("ethnus"));
  }
}
```

Equals - Example

The **String equalsIgnoreCase()** method compares the two given strings on the basis of content of the string irrespective of case of the string

```
public class Main{
  public static void main(String args[]){
    String s1="codemithra";
    String s2="codemithra";
    String s3="CODEMITHRA";
    String s4="ethnus";
    System.out.println(s1.equals(s2));
    System.out.println(s1.equals(s3));
    System.out.println(s1.equals(s4));
  }
}
```

indexOf()

<Codemithra /> ™

ETHNUS ™

Explore | Expand | Enrich

The **java string indexOf()** method returns index of given character value or substring.

If it is not found, it returns -1. The index counter starts from zero.

There are 4 types of indexOf method in java

Method	Description
int indexOf(int ch)	returns index position for the given char value
int indexOf(int ch, int fromIndex)	returns index position for the given char value and from index
int indexOf(String substring)	returns index position for the given substring
int indexOf(String substring, int fromIndex)	returns index position for the given substring and from index

indexOf() - Example


```
public class Main{
  public static void main(String args[]){
    String s1="this is index of example";
    int index1=s1.indexOf("is");
    int index2=s1.indexOf("index");
    System.out.println(index1+"  "+index2);
    int index3=s1.indexOf("is",4);
    System.out.println(index3);
    int index4=s1.indexOf('s');
    System.out.println(index4);
  }
}
```

indexOf(substring) - Example

This method takes substring as an argument and returns index of first character of the substring.

```
class Main {
   public static void main(String[] args) {
     String s1 = "This is indexOf method";
     int index = s1.indexOf("method");
     System.out.println("index of substring "+index);
 }
}
```

isEmpty - Example

<Codemithra /> ™

ETHNUS ™

Explore | Expand | Enrich

The **java string isEmpty()** method checks if this string is empty or not. It returns true, if length of string is 0 otherwise false

```
class Main{
  public static void main(String args[]){
    String s1="";
    String s2="ethnus";
    System.out.println(s1.isEmpty());
    System.out.println(s2.isEmpty());
  }
}
```

lastIndexOf() - Example

<Codemithra />™

ETHNUS™

Explore | Expand | Enrich

The **java string lastIndexOf()** method returns last index of the given character value or substring. If it is not found, it returns -1

```
class Main{
  public static void main(String args[]){
    String s1="this is index of example";
    int index1=s1.lastIndexOf('s');
    System.out.println(index1);
  }
}
```

replaceAll() - Example

<Codemithra />™

ETHNUS™

Explore | Expand | Enrich

The **java string replaceAll()** method returns a string replacing all the sequence of characters matching regex and replacement string.

```
class Main{
  public static void main(String args[]){
    String s1="codemithra is a very good learning platform";
    String replaceString=s1.replaceAll("a","e");
    System.out.println(replaceString);
  }
}
```

toLowerCase() - Example

The **java string toLowerCase()** method returns the string in lowercase letter

```
class Main{
  public static void main(String args[]){
    String s1="ETHNUS HELLO guYS";
    String s1lower=s1.toLowerCase();
    System.out.println(s1lower);
  }
}
```

toUpperCase() - Example

The **java string toUpperCase()** method returns the string in uppercase letter.

```
class Main{
  public static void main(String args[]){
    String s1="hello guys";
    String s1upper=s1.toUpperCase();
    System.out.println(s1upper);
  }
}
```

trim() - Example

The **java string trim()** method eliminates leading and trailing spaces.
The unicode value of space character is '\u0020'.

```
public class Main{
  public static void main(String args[]){
    String s1="  hello string   ";
    System.out.println(s1+"ethnus");
    System.out.println(s1.trim()+"ethnus");
  }
}
```

Number to String - Example


```
class Main {
    public static void main(String[] args) {
        double d = 858.48;
        String s = Double.toString(d);
        int dot = s.indexOf('.');
        System.out.println(dot + " digits " + "before decimal point.");
        System.out.println( (s.length() - dot - 1) + " digits after decimal
point.");
    }
}
```

codemithra@ethnus.com

+91 7815 095 095

+91 9019 921 340

https://learn.codemithra.com

ETHNUS™

Explore | Expand | Enrich

THANK YOU