JAVA STRING

By Umesh Sir

Contact us 7758094241

STRING

- ★ Introduction to Strings
- ★ String Constant Pool and String Interning
- ★ String Comparison
- ★ String Class Methods
- ★ StringBuffer
- ★ StringBuilder
- ★ String Programs

★ Introduction to Strings

- Strings are sequences of characters.
- In Java, strings are represented by the String class.
- Strings are immutable, meaning their values cannot be changed once created.
- There are two ways to create String Object in Java.

  1] By Using Litral :- `String greeting = "Hello, world!";`

  2] By new Keyword :- `String greeting = new String(“Hello, world!”);`

Concatenation:

```
String firstName = “CodeMind";
String lastName = “Technology";
String fullName = firstName + " " + lastName;
```

String Length:

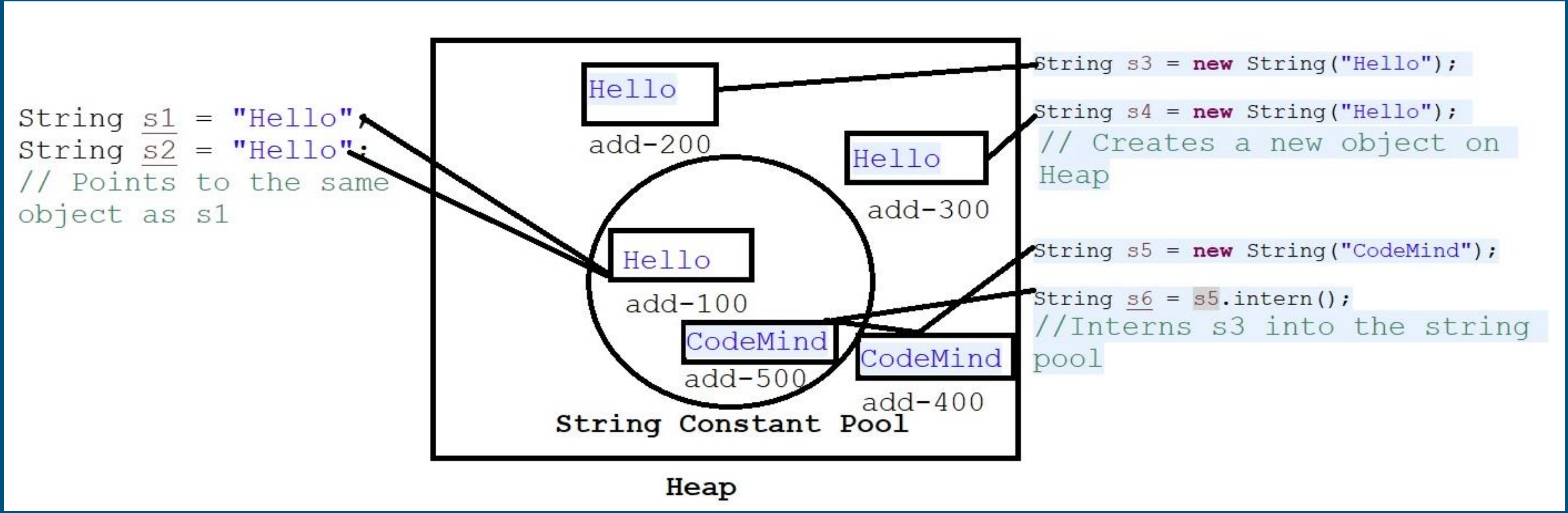
```
int length = str.length();
```

Accessing Characters:

```
char firstChar = str.charAt(0);
```

★ String Constant Pool

- Strings are stored in a special pool known as the "string Constant pool."
- String literals are automatically interned (added to the pool).
- You can explicitly intern strings using the intern() method.



```
String s1 = "Hello";
String s2 = "Hello";
// Points to the same
object as s1
```

```
String s3 = new String("Hello");
String s4 = new String("Hello");
// Creates a new object on
Heap
String s5 = new String("CodeMind");
String s6 = s5.intern();
//Interns s3 into the string
pool
```

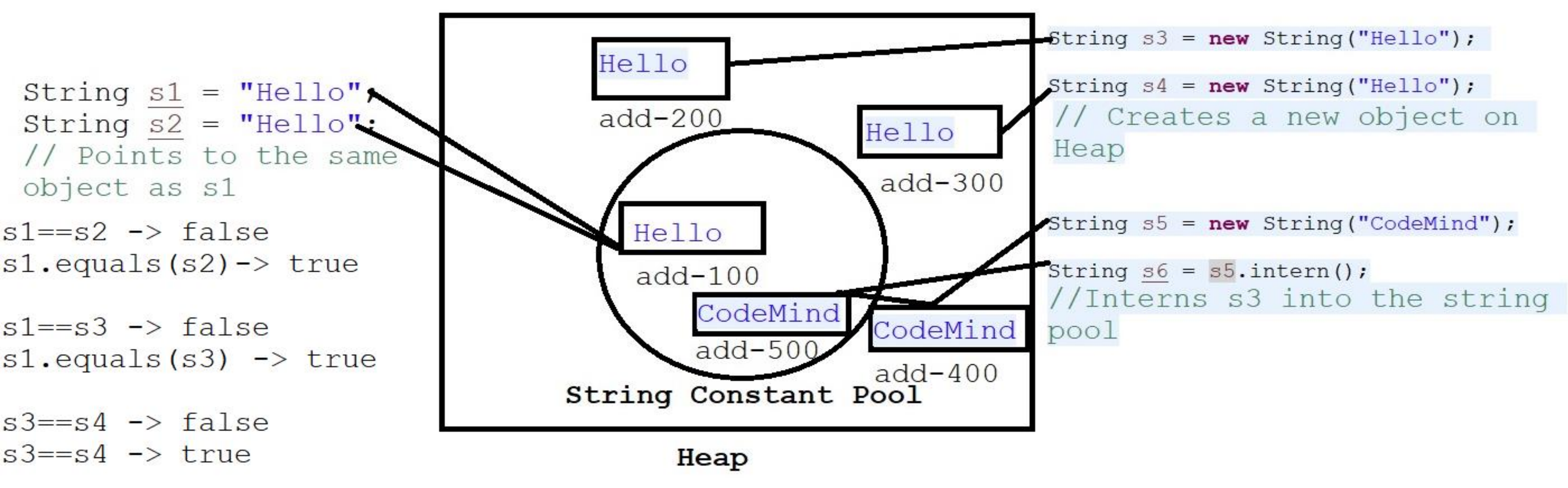
★ String Comparison

1] Using == operator

➢ It check addresses of objects

2] Using equals() Method of String Class and equals() method of object class

➢ String Class equals() method checks content of String and Object class equals() method checks address of two objects.



```
String s1 = "Hello";
String s2 = "Hello";
// Points to the same
object as s1
```

```
s1==s2 -> false
s1.equals(s2)-> true

s1==s3 -> false
s1.equals(s3) -> true

s3==s4 -> false
s3==s4 -> true
```

```
String s3 = new String("Hello");

String s4 = new String("Hello");
// Creates a new object on
Heap

String s5 = new String("CodeMind");

String s6 = s5.intern();
//Interns s3 into the string
pool
```

★ String Class Methods

1] String Concatenation - There are two ways to concat string

1.1] By + (String concatenation) operator :- String concatenation operator (+) is used to add strings.

```
String s = "Code";

s = s+"Mind";

System.out.println(s); //CodeMind

String s1 = 40+40+"CodeMind"+50+20;

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

1.2] By concat() method

```
Syntax - Pubic String concat(String another)

String st = "CodeMind";

st = st.concat(" Technology");

System.out.println(st);
```

2] Substring in Java

- A part of string is called substring.
- In case of substring start Index is inclusive and end index is exclusive.
- Start Index starts from 0 and end Index start from 1.

It has two methods:-

1] String subString(int startIndex)

```
String s= "CodeMind";
String res1 = s.substring(2);
System.out.println(res1);
```

2] String subString(int startIndex, int endIndex)

```
String s= "CodeMind";
String res = s.substring(4, 8);
System.out.println(res);
```

3] Other Methods in String Class

```
Public String toUpperCase();
Public String toLowerCase();
Public String trim();
String st = " Code ";
st = st.trim();
System.out.println(st);
Public string length();
Public boolean startswith(String prefix);
Public boolean endwith(String sufix);

Public char charAt(int index);
char val = s.charAt(4);
System.out.println("Char val -"+val);
```

★ StringBuffer

- StringBuffer is a class & it has only one way to create an object.

```
StringBuffer sb = new StringBuffer("Code");
StringBuffer sb1 = new StringBuffer("Mind");
Sb.append(sb1); // CodeMind
```

- StringBuffer class is used to create mutable (Midifieable) String.
- StringBuffer Class is same as String class except it is mutable (it can be changed).
- Important about StringBuffer

  1] It create mutable String 2] All methods are synchronized. 3] It is slower than String.

- StringBuffer class methods are synchronized means thread safe (Thread safe means -Multiple thread can't access it simultaneously so it is safe & will result in an order).

```
StringBuffer sb = new StringBuffer("Code");
StringBuffer sb1 = new StringBuffer("Mind");
    sb.append(sb1);
    sb.append(" Technology");
   System.out.println(sb);
```

★ StringBuilder

- JDK1.5 has introduced StringBuilder concept it is exact same as StringBuffer except it has non-synchronized methods.

-It is faster than StringBuffer.

```
StringBuilder sbuilder = new StringBuilder();

sbuilder.append("Code");

sbuilder.append("Mind");

System.out.println(sbuilder);
```

- ★ String Programs

- ★ T

Thank you