Java Strings - Complete Notes

1. Introduction to Strings

Strings in Java are objects that represent sequences of characters. They are immutable, meaning once created, they cannot be changed.

Java provides the `String` class in the `java.lang` package to work with strings.

2. String Constant Pool and Memory Storage

In Java, strings are stored in a special memory area called the String Constant Pool (SCP).

When a string is created using a string literal, it is stored in this pool to avoid memory duplication.

Example:

String s1 = "Hello";

String s2 = "Hello"; // s1 and s2 both refer to the same object in SCP

However, when a string is created using the `new` keyword, it is stored in the heap memory:

String s3 = new String("Hello"); // Stored outside the SCP

3. Ways to Create Strings

1. Using string literals (stored in SCP)

2. Using `new` keyword (stored in Heap)

3. From character arrays: new String(char[])

4. From byte arrays: new String(byte[])

```
char[] chars = {'J','a','v','a'};
```

```
String s1 = new String(chars);
System.out.println(s1);
```

Output: Java

4. Advantages of Strings in Java

- Strings are immutable, which makes them thread-safe and secure.
- Java optimizes memory usage via the String Constant Pool.
- The String class has many useful built-in methods.
- Easy to use and integrate with other APIs and libraries.
- Strings are widely used in file handling, networking, and user input.

5. Important String Methods with Examples

Method: length()

```
String str = "Hello";
System.out.println(str.length());
```

Output: 5

Method: charAt()

```
String str = "Hello";
System.out.println(str.charAt(1));
```

Output: e

Method: substring()

```
String str = "HelloWorld";
System.out.println(str.substring(5));
```

Output: World

Method: contains()

```
String str = "OpenAI ChatGPT";
System.out.println(str.contains("Chat"));
```

Output: true

Method: equals()

```
String a = "Java";
String b = "Java";
System.out.println(a.equals(b));
```

Output: true

Method: equalsIgnoreCase()

```
String a = "Java";
String b = "java";
System.out.println(a.equalsIgnoreCase(b));
```

Output: true

Method: toLowerCase()

```
String str = "HELLO";
System.out.println(str.toLowerCase());
```

Output: hello

Method: toUpperCase()

```
String str = "hello";
```

```
System.out.println(str.toUpperCase());
```

Output: HELLO

Method: trim()

```
String str = "  Hello  ";
System.out.println(str.trim());
```

Output: Hello

Method: replace()

```
String str = "banana";
System.out.println(str.replace('a', 'o'));
```

Output: bonono

Method: split()

```
String str = "a,b,c";
String[] parts = str.split(",");
System.out.println(parts[1]);
```

Output: b

Method: indexOf()

```
String str = "programming";
System.out.println(str.indexOf('g'));
```

Output: 3

Method: lastIndexOf()

```
String str = "programming";
```

```
System.out.println(str.lastIndexOf('g'));
```

Output: 10

Method: startsWith()

```
String str = "Java";
System.out.println(str.startsWith("Ja"));
```

Output: true

Method: endsWith()

```
String str = "Java";
System.out.println(str.endsWith("va"));
```

Output: true