Week 6 - LAQ's

Instructions

Explain various string handling methods in Java

In Java, the String class provides a variety of methods to manipulate and handle strings. Here's an overview of some of the most commonly used string handling methods along with examples:

1. Creating Strings

Strings in Java can be created using string literals or the new keyword.

String str1 = "Hello, World!"; // String literal

String str2 = new String("Hello, World!"); // Using new keyword

2. Length of a String

The length() method returns the number of characters in a string.

String str = "Hello";

int length = str.length(); // length = 5

3. Character Extraction

You can extract a character at a specific index using the charAt(int index) method.

char ch = str.charAt(1); // ch = 'e'

4. Substring

The substring(int beginIndex) and substring(int beginIndex, int endIndex) methods are used to extract a portion of the string.

String sub1 = str.substring(1); // sub1 = "ello"

String sub2 = str.substring(0, 3); // sub2 = "Hel"

5. String Comparison

The equals() and equalsIgnoreCase() methods are used to compare strings.

String strA = "Hello";

String strB = "hello";

boolean isEqual = strA.equals(strB); // isEqual = false

boolean isEqualIgnoreCase = strA.equalsIgnoreCase(strB); // isEqualIgnoreCase = true

6. String Concatenation

You can concatenate strings using the concat() method or the + operator.

String strC = strA.concat(" World!"); // strC = "Hello World!"

String strD = strA + " World!"; // strD = "Hello World!"

7. String Replacement

The replace(char oldChar, char newChar) and replace(CharSequence target, CharSequence replacement) methods replace characters or substrings.

String replaced = strA.replace('e', 'a'); // replaced = "Hallo"

String replacedAll = strA.replace("Hello", "Hi"); // replacedAll = "Hi"

8. Trimming Whitespace

The trim() method removes leading and trailing whitespace from the string.

String strWithSpaces = "   Hello   ";

String trimmed = strWithSpaces.trim(); // trimmed = "Hello"

9. Changing Case

You can change the case of a string using toLowerCase() and toUpperCase() methods.

String lower = strA.toLowerCase(); // lower = "hello"

String upper = strA.toUpperCase(); // upper = "HELLO"

10. Finding Substrings

The indexOf() method returns the index of the first occurrence of a specified substring or character.

int index = strA.indexOf('e'); // index = 1

int lastIndex = strA.lastIndexOf('l'); // lastIndex = 3

11. Splitting Strings

The split(String regex) method splits the string into an array based on the given regular expression.

String strSplit = "apple,banana,cherry";

String[] fruits = strSplit.split(","); // fruits = ["apple", "banana", "cherry"]

12. Joining Strings

The String.join() method allows you to join multiple strings with a specified delimiter.

String joined = String.join(", ", fruits); // joined = "apple, banana, cherry"

13. String Formatting

The String.format() method is used to create formatted strings.

String formatted = String.format("Hello, %s! You are %d years old.", "Alice", 30);2// formatted = "Hello, Alice! You are 30 years old."

14. Checking String Content

The contains() method checks if a string contains a specified sequence of characters.

boolean contains = strA.contains("ell"); // contains = true

Summary

Java's String class provides a rich set of methods for string manipulation, including methods for creating, comparing, modifying, splitting, and formatting strings. Understanding these methods is essential for effective string handling in Java applications.