

CS 313e - Elements of Software Design

Reading Assignment: Morelli, chapter 7
Ignore applet and cyberpet examples
Do exercises: 7.3, 7.4, 7.5, 7.6

Substrings

substring method of String class: extracts a substring from another string

Example:
String artist = "Pink Floyd";
String first = artist.substring(0, 4);

Syntax: substring(1st index to include, 1st index to exclude)

So first is "Pink".

Strings

Strings are sequences of characters.

Ex: "hello"

Java does not have a built-in string type.
Instead: String class in standard Java library

Examples:
String empty = ""; // empty string
String heyYou = "hello";

Concatenation of Strings

Use the + sign to append one string to another.

Example:
String hey = "hello";
String you = "Mary";

String message = hey + " + you";
Value of message: "hello Mary"

If you concatenate a string with a value that is not a string, the non-string is converted to string:

Ex: String emergency = "Call " + 911;
Sets emergency to "Call 911"

Editing Strings

Well, you cannot really edit a String. But you can modify a String and store the new String in a different String variable.

Recall: The length method returns the number of characters in a String.

String hey = "hello";
int len = hey.length(); // len is 5

To get individual characters of a String: charAt() method

Syntax: stringName.charAt(n)
Returns the character at position n in stringName

Now: modifying an existing string to get a new string.

Example:
hey = "hello";
hey = hey.substring(0,1)+"u"+hey.substring(2, 5);

Now variable hey refers to a different string - the string "hullo".

Testing Strings for Equality

The equals() method tests whether 2 strings are equal.

Syntax: s.equals(t)

Returns true if s and t are equal, false otherwise
s and t can be string variables or string constants.

Examples:

```
String hey = "hello";  
"hello".equals(hey) is true.
```

To test for equality while ignoring differences in case:
equalsIgnoreCase() method

Example:

```
"HELLO".equalsIgnoreCase(hey) is true  
hey.equalsIgnoreCase("Hello") is true
```

IMPORTANT: Do NOT use the == operator to test if 2 strings are identical. You will only test if the strings are in the same memory location.

If 2 strings are in the same memory location, they are certainly equal. But 2 strings can be identical and stored in different locations.

String replace (char oldChar, char newChar)

returns a new string that is obtained by replacing all occurrences of oldChar in the string with newChar.

boolean startsWith(String prefix)

returns true if string begins with prefix

String substring(int beginIndex)

String substring(int beginIndex, int endIndex)

returns a new string consisting of all characters from beginIndex until either end of string or endIndex (exclusive).

String toLowerCase()

returns a new string containing all characters in the original string with uppercase characters converted to lowercase.

String toUpperCase()

String trim()

returns a new string by eliminating all leading and trailing spaces in the original string

Methods from java.lang.String:

char charAt(int index)

returns character at specified location

int compareTo(String other)

returns negative value if string comes before other in dictionary order, positive value if string comes after other in dictionary order, or 0 if strings are equal

boolean endsWith(String suffix)

returns true if the string ends with suffix

boolean equalsIgnoreCase(String other)

returns true if the string equals other, except for case differences

int indexOf(String str)

int indexOf(String str, int fromIndex)

returns starting position of first substring equal to str, starting at either index 0 or fromIndex. returns -1 if str is not a substring.

int lastIndexOf(String str)

int lastIndexOf(String str, int fromIndex)

returns starting position of last substring equal to str, starting at index 0 or fromIndex

int length()

returns length of string