Strings

ACTLEA
Active Learning

Learning objectives

- String helper methods
- Comparing Strings
- Immutability of Strings

Strings

- A String is an object and not a primitive type
- A String can represent a number of Unicode characters (ie text)
- A String has many helper methods
- The text in the String is internally stored in a char array

Demo 1 - Strings

- Creating Strings
- Checking equality of Strings
- Some useful methods in Strings

Comparing Strings

- The == operator compare if two String variables are referring to the same object
- The .equals method compare if the value in two Strings are the same
- Use the .equals method if you want to compare the text in Strings!

Demo 2 - Comparing Strings

- Comparing Strings with ==
- Comparing Strings with equals

Strings are immutable

- The value of a String can never change!
- But what about this:

```
String msg = "Hello";
msg = msg + " World";
System.out.println(msg);
```

- This code will print "Hello World", didn't it change?
- The answer is actually no, the String didn't change but the reference in the variable did!

ACTLEA Active Learning

Strings are immutable

- Line 1: A String object is created in memory, a String variable with the name msg is created and a reference to the object is stored in the variable

- Line 2: A new String object is created in memory with the value "Hello World", and a reference to this new object is replacing the old reference in the variable

- If there is no other reference to the old String object with the value "Hello" then this object will be deleted from memory by the Garbage Collector

```
String msg = "Hello";
msg = msg + " World";
System.out.println(msg);
```

ACTLEA
Active Learning

StringBuilder

- StringBuilder represents a mutable String
- StringBuilder has many methods for modifying text that is not available in a String
- StringBuilder is more efficient to use when modifying text

Demo 3 - StringBuilder

- Creating a StringBuilder
- Using the StringBuilder
- StringBuilder is more efficient when modifying text

Useful methods in Strings

```
•String name = "Hello";
•int length = name.length(); // returns the length of the String
•char c = name.charAt(2); // returns the char at index 2
•int index = name.indexOf(c); // returns the first index of char
•boolean e = name.isEmpty(); // returns it it is empty or not
•boolean e2 = name.endsWith("o"); // if it ends with "y" or not
•String newName = name.toLowerCase(); // returns lowercase
•String newName2 = name.trim(); // trims beginning and end
•String[] s = name.split(”e"); // splits the String on "d" to an array
```

ACTLEA
Active Learning

Exercise 1 - Split

- Copy this String to the main method:

```
String names = "apples,bananas,lemons";
```

- Create a program that prints messages like this to the console:

```
Remember to buy apples!
Remember to buy bananas!
Remember to buy lemons!
```

- Hint 1: Run the split method with "," as input argument on the String names to get a String array with the individual fruit names
- Hint 2: Then loop over the String array and print the message with each fruit

ACTLEA
Active Learning

Learning objectives

- String helper methods
- Comparing Strings
- Immutability of Strings