STRINGS

Computer Engineering Department
Java Course

Asst. Prof. Dr. Ahmet Sayar

Kocaeli University - Fall 2014

Usefu Shortcuts for Netbeans IDE

- sout+Tab: shortcut for System.out.println("");
- fori+Tab
- trycatch+Tab
- iff+Tab
- ifelse+Tab
- forv+Tab
- dowhile+Tab
- psvm+Tab

Strings

- Strings are essentially arrays of characters
- The String class provides many functions for manipulating strings
  - Searching/matching operations
  - Replacing characters
  - Finding characters
  - Trimming whitespace
  - Etc.

Char vs. String

- "h" is a String
- 'h' is a char
- String is an Object; it contains methods

```
String s = "h";
s = s.toUpperCase();        // 'H'
int len = s.length();       //  1
char first = s.charAt(0);   // 'H'
```

- Char is primitive; you cant call methods on it

```
char c = 'h';
c = c.toUpperCase();    // ERROR: "cannot be
dereferenced"
```

Comparing char values

- You can compare char values with relational operators:
  - `'a' < 'b'` and `'X' == 'X'` and `'Q' != 'q'`
- An example that prints the alphabet:
```
for (char c = 'a'; c <= 'z'; c++) {
      System.out.print(c);
}
```
  - abcdefghijklmnopqrstuvwxyz

Declaring and Printing Strings

- ## declaring

```
String greeting;
greeting = "Hello!";
```
or
```
String greeting = "Hello!";
```
or
```
String greeting = new String("Hello!");
```

- ## printing

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

Concatenation of Strings

- Two strings are *concatenated* using the + operator.

```
String greeting = "Hello";
String sentence;
sentence = greeting + " officer";
System.out.println(sentence);
```

- Any number of strings can be concatenated using the + operator.

```
String solution;

solution = "The temperature is " + 72;
System.out.println (solution);
System.out.println (solution.length());
```

Some more useful codes
Conversion by method

- int to String:
  - String five = 5;   // ERROR!
  - String five = Integer.toString(5); // Double.toString(5)?
  - String five = "" + 5;   // five = "5"

- String to int:
  - int foo = "18";   // ERROR!
  - int foo = Integer.parseInt("18"); //works
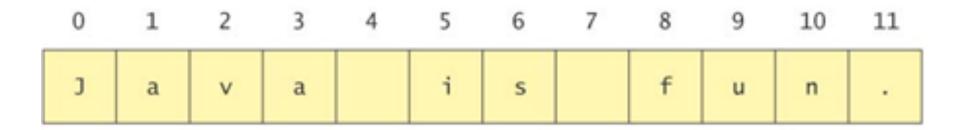 - Double d = Double.parseDouble("18"); //also works

Double-double vs. Integer-int

- Integer is class type – object type
  - int is primitive version of Integer

- Double is class type – object type
  - double primitive version of Double

Strings are Arrays

- ## A position is referred to an *index.*
  - The 'f' in "Java is fun." is at index 8.

0	1	2	3	4	5	6	7	8	9	10	11
J	a	v	a		i	s		f	u	n	.

positions start with 0, not 1.
The 'J' in "Java is fun." is in position 0

-String str = "testString";
-char[] charArray = str.toCharArray();

Strings

- No methods allow you to change the value of a `String` object.

- But you can change the value of a String variable.

	value of pause
`String pause = "    Hmm    ";`	**Hmm**
`pause = pause.trim()`	**Hmm**
`pause = pause + "mmm!";`	**Hmmmmm**!
`pause = "Ahhh";`	**Ahhh**

Escape Characters

\"	Double quote.
\'	Single quote.
\\	Backslash.
\n	New line. Go to the beginning of the next line.
\r	Carriage return. Go to the beginning of the current line.
\t	Tab. Add whitespace up to the next tab stop.

- Each escape sequence is a single character even though it is written with two symbols.

Examples

```
System.out.println("abc\\def");
```
abc\def
```
System.out.println("new\nline");
```
new

line
```
char singleQuote = '\'';
System.out.println(singleQuote);
```
`

String word1="A", word2="B", word3="C";

System.out.println("\""+word1 +"\"" +"\n" + word2+ "\t" + word3);

//How about using single quotation mark '\n' ?

String Concatenation (+)

- String text **= "hello" + " world";**
- text **= text + " number " + 5;**

- **Commenting:**
  - // text = "hello world number 5";
  - /* text = "hello world number 5" */

Using ==, cont.

- To test the equality of objects of class String, use method `equals`.

  `s1.equals(s2)`

  or

  `s2.equals(s1)`

- To test for equality ignoring case, use method `equalsIgnoreCase`.

  `("Hello".`**`equalsIgnoreCase`**`("hello"))`

Screen Output

- `System.out.print("aaa")`

- `System.out.println("bbb")`

- The concatenation operator (`+`) is useful when everything does not fit on one line.

  ```
  System.out.println("When everything " +
    "does not fit on one line, use the" +
    " concatenation operator (\'+\')");
  ```

- Alternatively, use `print()`

  ```
  System.out.print("When everything ");
  System.out.print("does not fit on ");
  System.out.print("one line, use the ");
  System.out.print("\"print\" ");
  System.out.println("statement");
  ```

  ending with a `println()`.

String as an Argument to Main

```java
public static void main (String[] arguments){
    System.out.println(arguments.length);
    System.out.println(arguments[0]);
    System.out.println(arguments[1]);
}
```

Arguments for the Method `main`, cont.

- Alternatively, an array of `String` values can be provided in the command line.

- example

```
java TestProgram Mary Lou
```

- `args[0]` is set to "`Mary`"

- `args[1]` is set to "`Lou`"

```
System.out.println("Hello " + args[0] +
      " " + args[1]);
```

prints `Hello Mary Lou.`

class String

Method name	Description
`indexOf(`**`str`**`)`	index where the start of the given string appears in this string (-1 if it is not there)
`length()`	number of characters in this string
`substring(`**`index1, index2`**`)` or `substring(`**`index1`**`)`	the characters in this string from *index1* (inclusive) to *index2* (exclusive); if *index2* omitted, grabs till end of string
`toLowerCase()`	a new string with all lowercase letters
`toUpperCase()`	a new string with all uppercase letters

These methods are called using the dot notation:

```
String gangsta = "Dr. Dre";
System.out.println(gangsta.length());    // 7
```

Class String - Cntd

Method	Description
equals(**str**)	whether two strings contain the same characters
equalsIgnoreCase(**str**)	whether two strings contain the same characters, ignoring upper vs. lower case
startsWith(**str**)	whether one contains other's characters at start
endsWith(**str**)	whether one contains other's characters at end
contains(**str**)	whether the given string is found within this one

```
String name = console.next();

if (name.startsWith("Dr.")) {
    System.out.println("Are you single?");
} else if (name.equalsIgnoreCase("LUMBERG")) {
    System.out.println("I need your TPS
reports.");
}
```

Class String -Cntd

Method	Description
`compareTo(String str)`	Compare two strings character by character. Returns 0 if this string is the same as str
`concat(String **str**)`	Returns the string that is this string concatenated
`charAt(int index)`	Returns the character at the position specified by index
`indexOf(char ch)`	Returns the index of the first occurrence of the character specified by ch. If the character is not found returns -1
`Replace(char c1, char c2)`	Returns the string in which every occurrence of c1 is replaced with c2.

```
String major = "CSE";
    for (int i = 0; i < major.length(); i++) {
        char c = major.charAt(i);
        System.out.println(c);
    }
```

```java
public class StringDemo
{
    public static void main(String[] args)
    {
        String sentence = "Text processing is hard!";
        int position;
        position = sentence.indexOf("hard");
        System.out.println(sentence);
        System.out.println("01234567890123456789 0123");
        System.out.println("The word \"hard\" starts at index
                            + position);
        sentence = sentence.substring(0, position) + "easy!";
        System.out.println("The changed string is:");
        System.out.println(sentence);
    }
}
```

Sample Screen Dialog

```
Text processing is hard!
01234567890123456789 0123
The word "hard" starts at index 19
The changed string is:
Text processing is easy!
```

The meaning of \" is discussed in the subsection entitled "Escape Characters."

```java
public class StringDemo_1 {

    public static void main(String[] args)
    {
        String sentence = new String( "Text processing is
hard!");
        int position;
        char a =sentence.charAt(1);
        System.out.println(a);
        sentence.concat("....");
        System.out.println(sentence);
        sentence=sentence.concat("....");

        position = sentence.indexOf("hard");
        System.out.println(sentence);
        System.out.println("01234567890123456789 0123");
        System.out.println("The word \"hard\" starts at index "
                            + position);
        sentence = sentence.substring(0, position) + "easy!";
        System.out.println("The changed string is:");
        System.out.println(sentence);
    }
}
```