COMS W1114 - Java Lab

Lab 7
Wednesday, March 10, 2004
&
Thursday, March 11, 2004

-1-

Note

- HW3 Due by Tuesday, March 23, at 11:00am
- Any midterm questions? How was it?

-2-

What we are covering today

- Review from Lab 6
  - File Output
  - Debugging Strategies
- Formatting & Advanced I/O
  - Strings
  - StringTokenizer
  - Envelopes
- HW3
  - review assignment specifications
  - Readme Files

-3-

File Output

- Instead of printing to the console (using System.out) we want to print to a file
- We need to create our own output object to redirect the output

```
import java.io.*;
public class Lab5Example{
        public static void main(String[] args)throws IOException{
                System.out.print(“Please enter your name: ”);
                FileWriter writer = new FileWriter(“output.txt”);
                writer.write(“this sentence will get written to a file\n”);
                writer.close();
        }

}                               OR
import java.io.*;
public class Lab5Example{
        public static void main(String[] args)throws IOException{
                System.out.print(“Please enter your name: ”);
                File f = new File(“output.txt”);
                FileWriter writer = new FileWriter(f);
                writer.write(“this sentence will get written to a file\n”);
                writer.close();
        }
```

-4-

Debugging

- Syntax vs. Semantic errors
- Basic testing (aka going beyond “it compiles!”)
- Your friend: System.out.println();

-5-

Strings(1)

- String is a class. NOT a primitive data type.
  `String big = “hippopotamus”;  //note: not using new`
- String has built-in + operator
  ```
  String s1=“4”;
  String s2=“5”;
  String s3 = s1 + s2;  //  s3 is “45”;
  ```
- Once created, cannot be changed
  - string can be assigned new value but not inherently changed.
- A difference between initialized but empty Strings and non-initialized (null) Strings.
  ```
  String s1;
  String s2 = “”;
  ```

-6-

Strings(2)

- String as several constructors (see Java API)
  - We will usually use assignment from a string literal or variable\

```
String s3=“Hello”;
String s4=args[0];
String s5= in.readLine(); //assume you have a reader in
```

- Strings have many methods (see Java API)
  - class methods
  - instance methods

-7-

Strings(3)

- Examples:

```
String big=“hippopotamus”;
char study [] = big.toCharArray();

int bang = big.indexOf(“pop”);
int fizz = big.indexOf(“up”);

String small = big.substring(3,5);

System.out.println(big.equals(small);
System.out.println(big.compareTo(small);
```

- What do we expect as the result of each?
- ChequeDetector.java example from book.

-8-

StringTokenizer(1)

- Breaks a String into tokens
  - Tokens are substrings separated by some character (space, comma, tab, etc.)
- declaration

```
StringTokenizer st = new StringTokenizer(aString);
```

- Now easily iterate through the tokens

```
while(st.hasMoreTokens()){
    System.out.println(st.nextToken());
 }
```

- Assume

```
String aString=“The quick brown fox jumped over the lazy dog.”;
```

- What does the above code do?

-9-

StringTokenizer(2)

- You do not have to tokenize on a space (“ “). You can change the delimiter when you declare the StringTokenizer:

```
StringTokenizer st = new StringTokenizer(aString,”,. ”,false );
```

- Now what happens with:

```
 String aString=“Really, the quick brown fox jumped over the
    (lazy) dog.No joke.”;
 StringTokenizer st = new StringTokenizer(aString,”,. ”,false);
 while(st.hasMoreTokens()){
     System.out.println(st.nextToken());
 }
```

-10-

Envelopes (aka. wrappers)

- We’ve seen them already - used for data conversion
- Integer and int conversion examples:

```
Integer myInteger = new Integer(50);
Integer myInteger2 = Integer.valueOf(“100”);
int iterations = Integer.parseInt(argv[0]);
int a = myInteger2.intValue();
int b = myInteger.parseInt(“100”);
```

  – results of each?

Also:

- Java rule: Values of primitive types and Objects cannot be mixed
- This is only a problem when a package requires an object and all we have is a primitive. Must “wrap” our primitive with an Object (Boolean, Character, Double, Float, Integer, Long)

-11-

HW3

- Let’s review the specs
  – be sure to follow the naming conventions
  – be sure to only what you are asked (I.e. no spurious System.out.print’s
- A Readme file
  – a TEXT file (no msword, postscript, etc. think: notepad)
  – includes your name and UNI
  – homework #
  – a sentence or two outlining what your program does
  – a few instructions how to run your program
  – list known limitations or bugs
  – add anything else you deem important.

-12-

Example Readme

readme.txt

```
William Beaver  (wmb2013)
cs1004 HW3 - Bank.java

My program, Bank.java, keeps  track of bank accounts by name and
balance. It uses two  arrays -- one for people's names and one
for their balances.  While running the program, type “h” for help
on the commands to execute.
etc…

To compile, type “javac Bank.java” at the command prompt
To run, type “java Bank” after compiling.
Follow the onscreen instructions to run the program.

There are no known bugs or limitations based on the significant
testing I’ve done.

I added a feature to etc…in addition to the original
specifications.  Etc. Etc.
(end.)
```

-13-

Next time

- Formatters
  - Locale
  - DateFormat
  - SimpleDateFormat
  - NumberFormat
  - DecimalFormat
  - MessageFormat
- Begin OO (Design, properties, references)

-14-