Java Programming

Arthur Hoskey, Ph.D.
Farmingdale State College
Computer Systems Department

Today's Lecture

- Java Input/Output

- The Scanner class is used to read input from the keyboard.

- Here is code to create an instance of the Scanner class:

Read from standard input (the keyboard)

```
Scanner input = new Scanner(System.in);
```

Java Input - Console

```
import java.util.Scanner;

public class Welcome1
{

   public static void main( String args[] )
   {
         int n;
         String s
         Scanner input = new Scanner(System.in);

         System.out.printf("Enter a string: ");
         s = input.nextLine();

         System.out.printf("Enter an int: ");
         n = input.nextInt();
   }
}
```

Use nextLine() to read in a string

Use nextInt() to read in an int

Java Input - Console

nextInt() – Reads the next integer from the Scanner. DOES NOT CONSUME ANY DATA AFTER THE INTEGER THAT WAS READ!!! So, if there is a newline character after the int it will remain in the input stream. It will skip any whitespace that appears BEFORE the int.

nextDouble() – Reads the next double from the Scanner. DOES NOT CONSUME ANY DATA AFTER THE INTEGER THAT WAS READ!!! So, if there is a newline character after the double it will remain in the input stream. It will skip any whitespace that appears BEFORE the double.

next() – Reads the next string from the Scanner. This means it will read characters until it reaches a whitespace character (newline, tab etc…). DOES NOT CONSUME ANY DATA AFTER THE STRING THAT WAS READ!!! So, if there is a newline character after the string it will remain in the input stream. It will skip any whitespace that appears BEFORE the string.

nextLine() – Reads the next line of data from the Scanner. Reads all data up until it reaches a newline character. This will consume the newline character ('\n'). The newline character doe not appear in the output. Does not skip whitespace.

Java Input - A Few Scanner Methods

- Read from the console.

Connect the Scanner to System.in

```
String s;
Scanner input = new Scanner(System.in);

s = input.nextLine();
```

This call to nextLine reads a string from the keyboard since the Scanner is connected to System.in

Java Input - Console

Java Input - File

- Use the scanner to read from a file.
- Create a FileReader instance connected to the file you want to read from.
- Pass the FileReader instance into the Scanner.

```
String filename = "MyInputFile.txt";
Scanner  inputScanner;


FileReader fr = new FileReader(filename);

inputScanner = new Scanner( fr );
String s;
s = inputScanner.nextLine();
```

Create a FileReader instance connected to the file you want to read from

Pass the FileReader instance into the Scanner

This call to nextLine reads a string from the file since the Scanner is connected to the file

Java Output – File

- Instead of printing to the screen you can write data to a file.
- Use the code on the next slide to write to a file…

```
PrintStream ps = null;

try
{
    ps = new PrintStream("ArthurOutput.txt");
}
catch (Exception e)
{
    System.out.println("ERROR. Could not open file!");
}

ps.println("Done");
```

Only need to open the file in a try block

Output filename

If an error occurs when opening the file the catch block code runs

Java Output - File

```
PrintStream ps = null;

try
{
    ps = new PrintStream(System.out);
}
catch (Exception e)
{
    System.out.println("ERROR. Could not open file!");
}

ps.println("Done");
```

Use System.out to make a PrintStream send data to the console (instead of a file)

Java Output - Console Using PrintStream

print vs println

print does NOT go to the next line:

```
System.out.print("Yanks are ");
System.out.print("number 1");
```

Prints
Yanks are number 1

println goes to the next line:

```
System.out.println("Yanks are ");
System.out.println("number 1");
```

Prints
Yanks are
number 1

print vs println

printf

- printf – print formatted
- Use format specifiers to put data inside of a string.

%d is a format specifier.

This data will be put into the format specifier

```
System.out.printf("Yanks are number %d", 1);
```

- This statement prints "Yanks are number 1".
- The %d is replaced by 1.

- printf does NOT automatically add a carriage return.

printf

printf

- Use a variable argument.

Better to use a variable as the argument

```
int rating=1;
System.out.printf("Yanks are number %d", rating);
```

- Output is the same as before.

printf

printf

- You can use as many format specifiers as you want.
- The arguments at the end go into the format specifiers in the order that they appear

```
String team="Yanks";
int rating=1;

System.out.printf("%s are number %d", team, rating);
```

Two format specifiers

team is the first argument so it goes in the first format specifier (%s)

rating is the second argument so it goes in the second format specifier (%d)

%s is the string format specifier
%d is the number format specifier

printf

printf

- Add the "\n" to the string to insert a carriage return

```
System.out.printf("Yanks are number 1\n");
```

- You can use as many "\n" as you like.
- "\n" is called an escape sequence

- Note: To print a "\" in the output you must use two slashes in a row. For example, "\\" will print one "\" in the output.

printf

Format Specifier	Variable Type
%d	int
%f	float, double
%s	String
%b	boolean

printf – Some Format Specifiers

printf - Columns

- You can print data in columns using printf.

Column Widths (string)

You can set a columns width using printf. This code sets the column width for a String format specifier.

Make the string appear in a column 20 wide

```
String  name = "Arthur";
System.out.printf("Name: %20s\n", name);
```

This will display the following (notice the padding after is):

Name: Arthur

"Name: " is not in a column

Puts name in a column 20 characters wide and pads with spaces

printf - Columns

printf – Floating Point Numbers

- You can print floating point data using printf.

Floating Point Formatting

Show a certain number of places after the decimal point:

Use . followed by the number of places after the decimal that you want

```
double num = 10.4567;
System.out.printf("num is %.2f\n", num);
```

This will display the following:

num is 10.46

Note: Java rounds off the number automatically.

printf – Floating Point Numbers

printf – Columns and Floating Point

- You can print floating point numbers in columns.

Column Widths and Floating Point Numbers

You can set a column's width for a floating point format specifier using printf. The following sets the column width to 20 and the decimal places to 2.

Column Width (20)

Places after decimal (2)

```
double num = 10.4567;
System.out.printf("num is %20.2f\n", num);
```

This will display the following (notice the padding after is):

num is 10.46

Rounds the number to 2 places after the decimal and puts it in a column 20 wide

printf – Columns and Floating Point

- End of Slides

End of Slides