Computer Programming I Instructor: Greg Shaw

COP 2210

***Reading User Input***

(Using the *showInputDialog* method of the *JOptionPane* class)

This document shows how Java makes it easy to use GUI elements, in this case a *dialog box* that allows the user of a program to enter input.

1. **The *showInputDialog* Method**

* The *JOptionPane* class has a static method called *showInputDialog* that displays an input dialog box and returns a string entered by the user.
* Syntax:

*string-var* = JOptionPane.showInputDialog(*prompt*) ;

where *string-var* is a String object variable, and

*prompt* is a string expression

* Execution:

1. An input dialog box containing the prompt is displayed
2. The user types the input and clicks the **OK** button
3. The user input is returned and stored in *string-var*

* For the *JOptionPane* class, use this **import** statement:

**import** javax.swing.JOptionPane ;

* Note that the user input is *always* returned as a *string*.

1. **The *parseInt* and *parseDouble* Methods**

* To get numeric input, you must explicitly convert the string returned by *showInputDialog* to **int** or **double.**
* The **Integer** class has a static method, *parseInt* that takes a string argument and returns the argument converted to an **int.**
* Similarly,the **Double** class has a static method, *parseDouble* that takes a string argument and returns the argument converted to a **double.**
* Syntax:

*int-var* = **Integer**.parseInt(*arg*) ; *or*

*double-var* = **Double**.parseDouble(*arg*) ;

where *arg* is a string literal, variable, or expression

* Execution:

The *argument* is converted to an **int** or **double**, respectively, and stored in the variable to the left

* Note that if the argument contains any characters that are not valid for an **int** or **double** literal, respectively, then a **NumberFormatException** is thrown

1. **System.exit(0)**

When using the *JOptionPane* class, the last statement in the *main* method must be:

System.exit(0) ;

*showInputDialog* starts a new *thread* to handle user input. A thread is a program unit that is executed independently of other parts of the program. This thread does not terminate when all statements in *main* have been executed so your program will not end unless you explicitly call the *exit* method. The parameter *0* tells the operating system that the program execution was successful

1. **An Overloaded *showInputDialog* Method (Optional)**

* The JOptionPane class also contains an overloaded version of showInputDialog that takes four arguments:

*string-var* = JOptionPane.showInputDialog

(**null**, *prompt, title,*

JOptionPane.**QUESTION\_MESSAGE**) ;

* *string-var* is a String object variable
* **null** is a Java keyword
* *prompt* is a string that displays above the text box
* *title* is a string that displays in the Title Bar
* JOptionPane.**QUESTION\_MESSAGE** causes a "**?**" icon to appear
* As an example of the four-argument *showInputDialog* method, the statement:

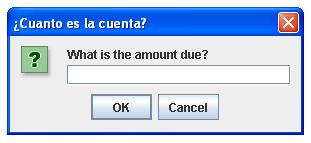
String input = JOptionPane.showInputDialog(null,

"What is the amount due?",

"\u00BFCuanto es la cuenta?",

JOptionPane.QUESTION\_MESSAGE) ;

will display this input dialog:



* For examples of the one-argument *showInputDialog* method and the *parseInt and parseDouble* methods, see *InputTest.java* and *ChangeMakerTest.java*