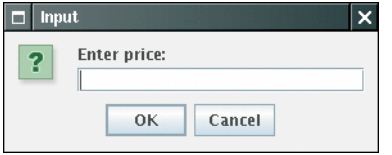
Special Topic 4.7

Using Dialog Boxes for Input and Output

Most program users find the console window rather old-fashioned. The easiest alternative is to create a separate pop-up window for each input (see the figure).



An Input Dialog Box

Call the static `showInputDialog` method of the `JOptionPane` class, and supply the string that prompts the input from the user. For example,

```
String input = JOptionPane.showInputDialog("Enter price:");
```

That method returns a `String` object. Of course, often you need the input as a number. Use the `Integer.parseInt` and `Double.parseDouble` methods to convert the string to a number:

```
double price = Double.parseDouble(input);
```

You can also display output in a dialog box:

```
JOptionPane.showMessageDialog(null, "Price: " + price);
```

Finally, whenever you call the `showInputDialog` or `showMessageDialog` method in a program that does not show any other frame windows, you need to add a line

```
System.exit(0);
```

to the end of your `main` method. The `showInputDialog` method starts a user interface thread to handle user input. When the `main` method reaches the end, that thread is still running, and your program won't exit automatically. To force the program to exit, you need to call the `exit` method of the `System` class. The parameter of the `exit` method is the status code of the program. A code of `0` denotes successful completion; you can use nonzero status codes to denote various error conditions.