How to Use JOptionPane Option for User Input

Dialog Boxes

- A *dialog box* is a small graphical window that displays a message to the user or requests input.
- A variety of dialog boxes can be displayed using the *JOptionPane* class.
- Two of the dialog boxes are:
  - Message Dialog - a dialog box that displays a message.
  - Input Dialog - a dialog box that prompts the user for input.

Using the *import* Statement

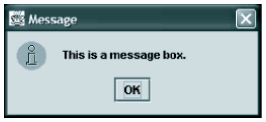
- The JOptionPane class is not automatically available to your Java programs.
- The following statement must be before the program’s class header:

```
import javax.swing.JOptionPane;
```

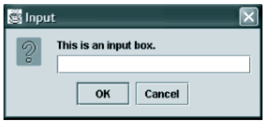
- This statement tells the compiler where to find the JOptionPane class.

Dialog Boxes

The JOptionPane class provides static methods to display each type of dialog box.



This is a message box.



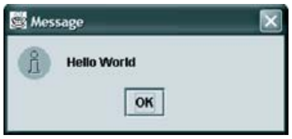
This is an input box.

Message Dialogs

- JOptionPane.showMessageDialog method is used to display a message dialog.

```
JOptionPane.showMessageDialog(null, "Hello World");
```

- The first argument will be discussed later.
- The second argument is the message that is to be displayed.



Hello World

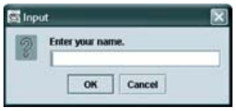
Input Dialogs

- An input dialog is a quick and simple way to ask the user to enter data.
- The dialog displays a text field, an Ok button and a Cancel button.
- If Ok is pressed, the dialog returns the user’s input.
- If Cancel is pressed, the dialog returns null.

Input Dialogs

```
String name;
name = JOptionPane.showInputDialog(
         "Enter your name.");
```

- The argument passed to the method is the message to display.
- If the user clicks on the OK button, name references the string entered by the user.
- If the user clicks on the Cancel button, name references null.



Enter your name.

The System.exit() Method

- A program that uses JOptionPane does not automatically stop executing when the end of the main method is reached.
- Java generates a *thread*, which is a process running in the computer, when a JOptionPane is created.
- If the System.exit method is not called, this thread continues to execute.

The System.exit() Method

- The System.exit method requires an integer argument.

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

- This argument is an *exit code* that is passed back to the operating system.
- This code is usually ignored, however, it can be used outside the program:
  - to indicate whether the program ended successfully or as the result of a failure.
  - The value 0 traditionally indicates that the program ended successfully.

Converting a String to a Number

- The JOptionPane's showInputDialog method always returns the user's input as a String
- String containing a number, such as "127.89, can be converted to a numeric data type.

The Parse Methods

- Each of the numeric wrapper classes, Chapter 8, has a static method that converts a string to a number.
  - The Integer class has a method that converts a string to an int,
  - The Double class has a method that converts a string to a double, and
  - etc.
- These methods are known as *parse methods* because their names begin with the word "parse."

The Parse Methods

```
byte bVar = Byte.parseByte("1");
  // Store 1 in bVar.
int iVar = Integer.parseInt("2599");
  // Store 2599 in iVar.
short sVar = Short.parseShort("10");
  // Store 10 in sVar.
long lVar = Long.parseLong("15908");
  // Store 15908 in lVar.
float fVar = Float.parseFloat("12.3");
  // Store 12.3 in fVar.
double dVar = Double.parseDouble("7945.6");
  // Store 7945.6 in dVar.
```