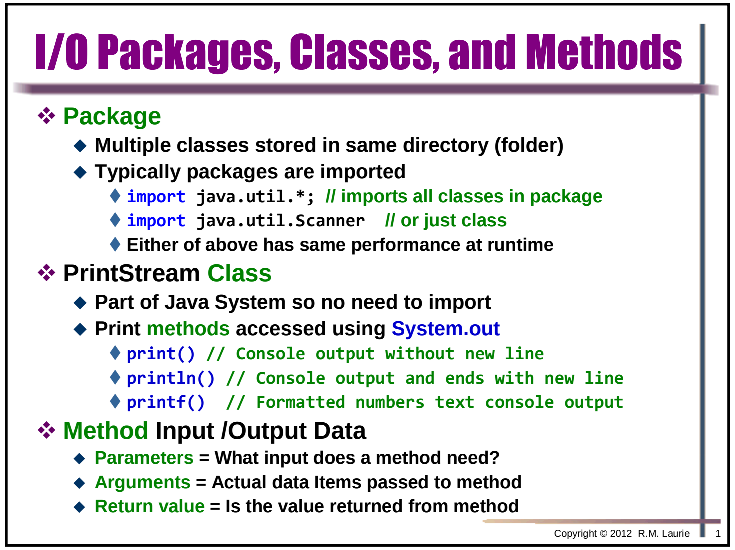
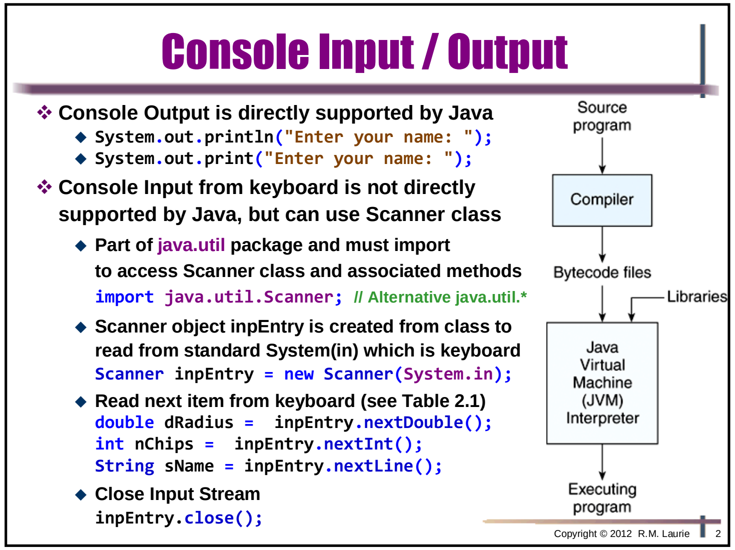


I/O Packages, Classes, and Methods

- ❖ Package
  - Multiple classes stored in same directory (folder)
  - Typically packages are imported
    - `import java.util.*;` // imports all classes in package
    - `import java.util.Scanner` // or just class
    - Either of above has same performance at runtime
- ❖ PrintStream Class
  - Part of Java System so no need to import
  - Print methods accessed using System.out
    - `print() // Console output without new line`
    - `println() // Console output and ends with new line`
    - `printf() // Formatted numbers text console output`
- ❖ Method Input /Output Data
  - Parameters = What input does a method need?
  - Arguments = Actual data Items passed to method
  - Return value = Is the value returned from method

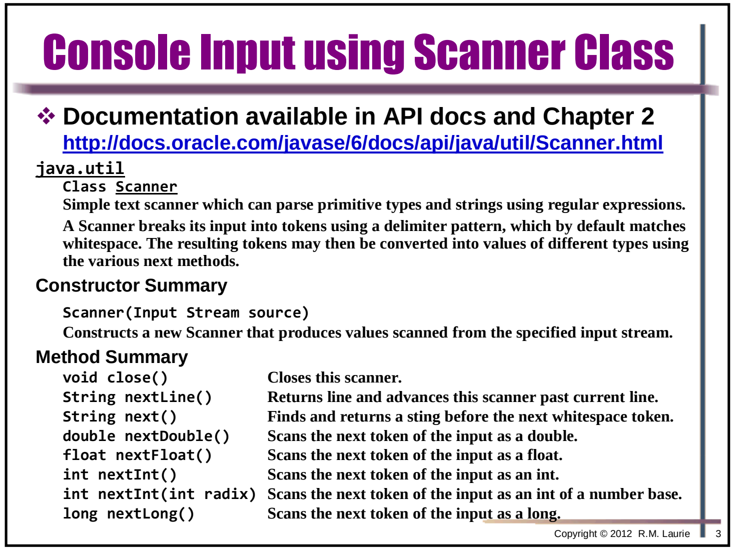


Console Input / Output

- ❖ Console Output is directly supported by Java
  - `System.out.println("Enter your name: ");`
  - `System.out.print("Enter your name: ");`
- ❖ Console Input from keyboard is not directly supported by Java, but can use Scanner class
  - Part of java.util package and must import to access Scanner class and associated methods
    `import java.util.Scanner;` // Alternative java.util.*
  - Scanner object inpEntry is created from class to read from standard System(in) which is keyboard
    `Scanner inpEntry = new Scanner(System.in);`
  - Read next item from keyboard (see Table 2.1)

```
double dRadius =  inpEntry.nextDouble();
int nChips =  inpEntry.nextInt();
String sName = inpEntry.nextLine();
```

  - Close Input Stream
    `inpEntry.close();`



Console Input using Scanner Class

- ❖ Documentation available in API docs and Chapter 2
  http://docs.oracle.com/javase/6/docs/api/java/util/Scanner.html

`java.util`

`Class Scanner`

Simple text scanner which can parse primitive types and strings using regular expressions.

A Scanner breaks its input into tokens using a delimiter pattern, which by default matches whitespace. The resulting tokens may then be converted into values of different types using the various next methods.

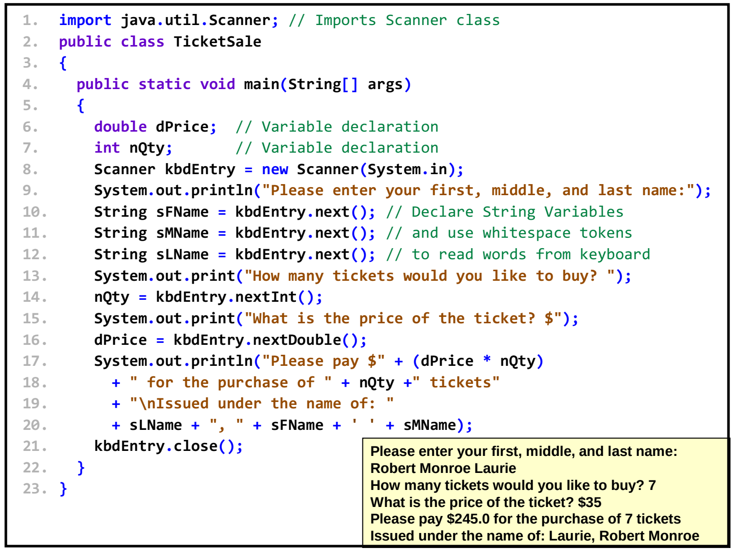
Constructor Summary

`Scanner(Input Stream source)`

Constructs a new Scanner that produces values scanned from the specified input stream.

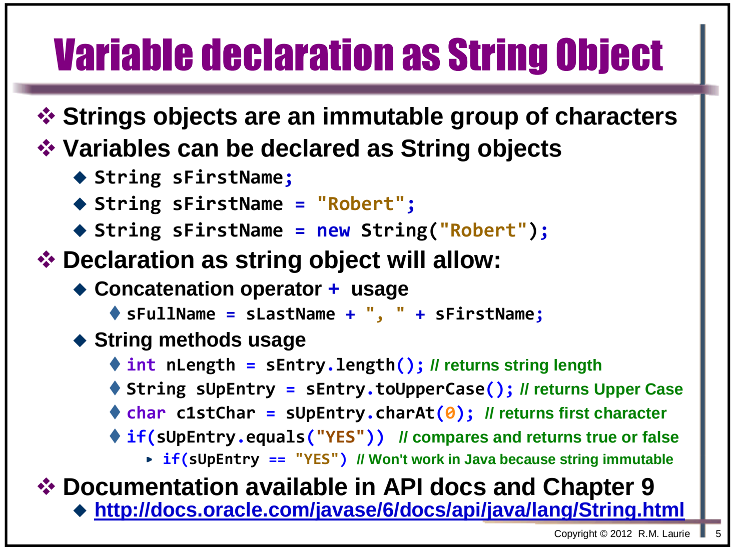
Method Summary

`void close()`	Closes this scanner.
`String nextLine()`	Returns line and advances this scanner past current line.
`String next()`	Finds and returns a sting before the next whitespace token.
`double nextDouble()`	Scans the next token of the input as a double.
`float nextFloat()`	Scans the next token of the input as a float.
`int nextInt()`	Scans the next token of the input as an int.
`int nextInt(int radix)`	Scans the next token of the input as an int of a number base.
`long nextLong()`	Scans the next token of the input as a long.



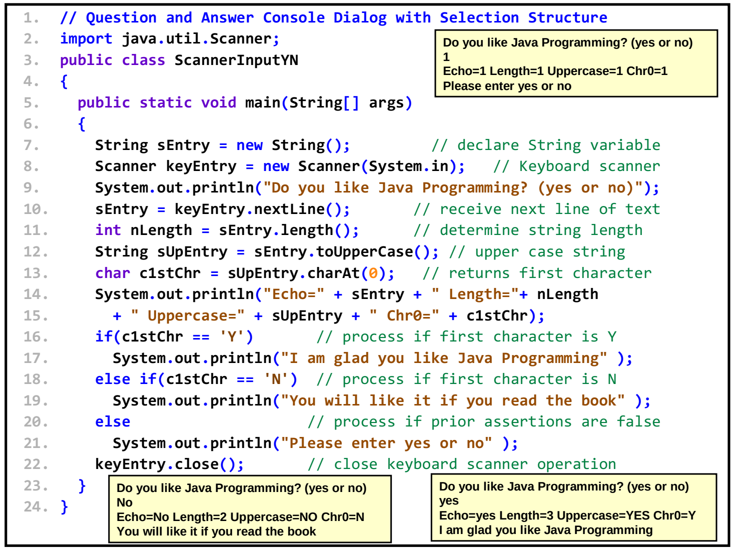
```
import java.util.Scanner; // Imports Scanner class
public class TicketSale
{
  public static void main(String[] args)
  {
    double dPrice;  // Variable declaration
    int nQty;       // Variable declaration
    Scanner kbdEntry = new Scanner(System.in);
    System.out.println("Please enter your first, middle, and last name:");
    String sFName = kbdEntry.next(); // Declare String Variables
    String sMName = kbdEntry.next(); // and use whitespace tokens
    String sLName = kbdEntry.next(); // to read words from keyboard
    System.out.print("How many tickets would you like to buy? ");
    nQty = kbdEntry.nextInt();
    System.out.print("What is the price of the ticket? $");
    dPrice = kbdEntry.nextDouble();
    System.out.println("Please pay $" + (dPrice * nQty)
      + " for the purchase of " + nQty +" tickets"
      + "\nIssued under the name of: "
      + sLName + ", " + sFName + ' ' + sMName);
    kbdEntry.close();
  }
}
```

```
Please enter your first, middle, and last name:
Robert Monroe Laurie
How many tickets would you like to buy? 7
What is the price of the ticket? $35
Please pay $245.0 for the purchase of 7 tickets
Issued under the name of: Laurie, Robert Monroe
```



Variable declaration as String Object

- ❖ Strings objects are an immutable group of characters
- ❖ Variables can be declared as String objects
  - `String sFirstName;`
  - `String sFirstName = "Robert";`
  - `String sFirstName = new String("Robert");`
- ❖ Declaration as string object will allow:
  - Concatenation operator + usage
    - `sFullName = sLastName + ", " + sFirstName;`
  - String methods usage
    - `int nLength = sEntry.length();` // returns string length
    - `String sUpEntry = sEntry.toUpperCase();` // returns Upper Case
    - `char c1stChar = sUpEntry.charAt(0);` // returns first character
    - `if(sUpEntry.equals("YES"))` // compares and returns true or false
      - `if(sUpEntry == "YES")` // Won't work in Java because string immutable
- ❖ Documentation available in API docs and Chapter 9
  - http://docs.oracle.com/javase/6/docs/api/java/lang/String.html

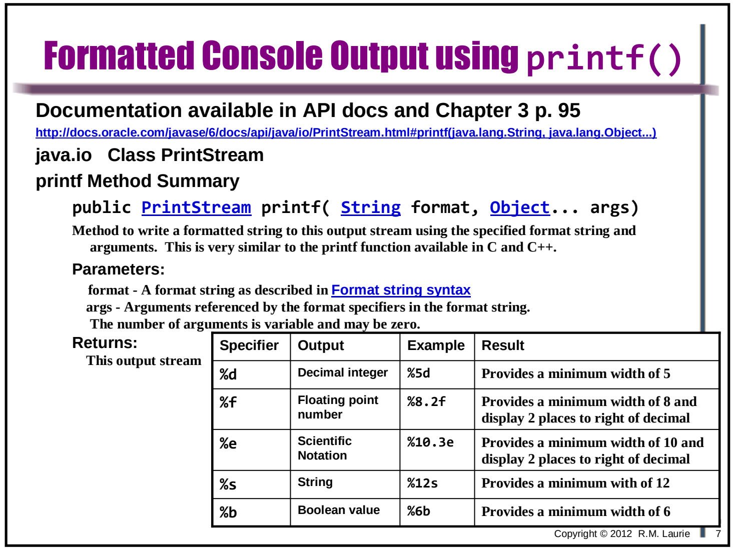


```
// Question and Answer Console Dialog with Selection Structure
import java.util.Scanner;
public class ScannerInputYN
{
  public static void main(String[] args)
  {
    String sEntry = new String();          // declare String variable
    Scanner keyEntry = new Scanner(System.in);   // Keyboard scanner
    System.out.println("Do you like Java Programming? (yes or no)");
    sEntry = keyEntry.nextLine();        // receive next line of text
    int nLength = sEntry.length();       // determine string length
    String sUpEntry = sEntry.toUpperCase(); // upper case string
    char c1stChr = sUpEntry.charAt(0);   // returns first character
    System.out.println("Echo=" + sEntry + " Length="+ nLength
      + " Uppercase=" + sUpEntry + " Chr0=" + c1stChr);
    if(c1stChr == 'Y')        // process if first character is Y
      System.out.println("I am glad you like Java Programming" );
    else if(c1stChr == 'N')  // process if first character is N
      System.out.println("You will like it if you read the book" );
    else                     // process if prior assertions are false
      System.out.println("Please enter yes or no" );
    keyEntry.close();        // close keyboard scanner operation
  }
}
```

```
Do you like Java Programming? (yes or no)
1
Echo=1 Length=1 Uppercase=1 Chr0=1
Please enter yes or no
```

```
Do you like Java Programming? (yes or no)
No
Echo=No Length=2 Uppercase=NO Chr0=N
You will like it if you read the book
```

```
Do you like Java Programming? (yes or no)
yes
Echo=yes Length=3 Uppercase=YES Chr0=Y
I am glad you like Java Programming
```



Formatted Console Output using `printf()`

Documentation available in API docs and Chapter 3 p. 95

http://docs.oracle.com/javase/6/docs/api/java/io/PrintStream.html#printf(java.lang.String, java.lang.Object...)

java.io Class PrintStream

printf Method Summary

`public PrintStream printf( String format, Object... args)`

Method to write a formatted string to this output stream using the specified format string and arguments. This is very similar to the printf function available in C and C++.

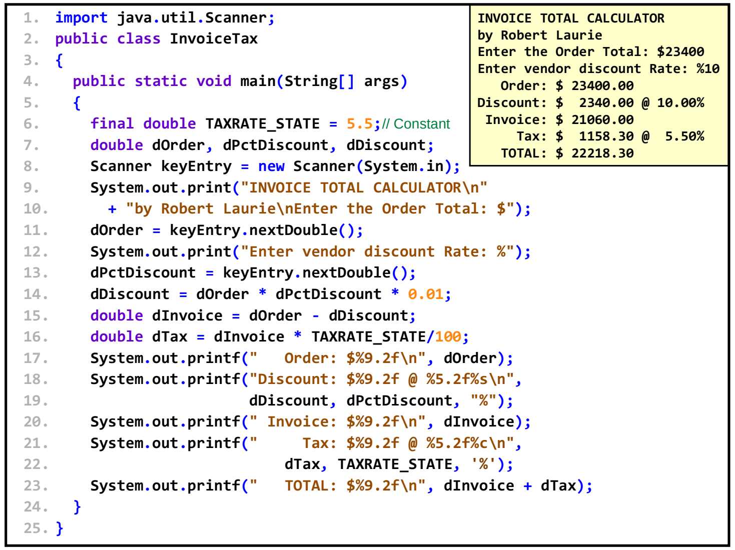
Parameters:

format - A format string as described in Format string syntax
args - Arguments referenced by the format specifiers in the format string.
The number of arguments is variable and may be zero.

Returns:

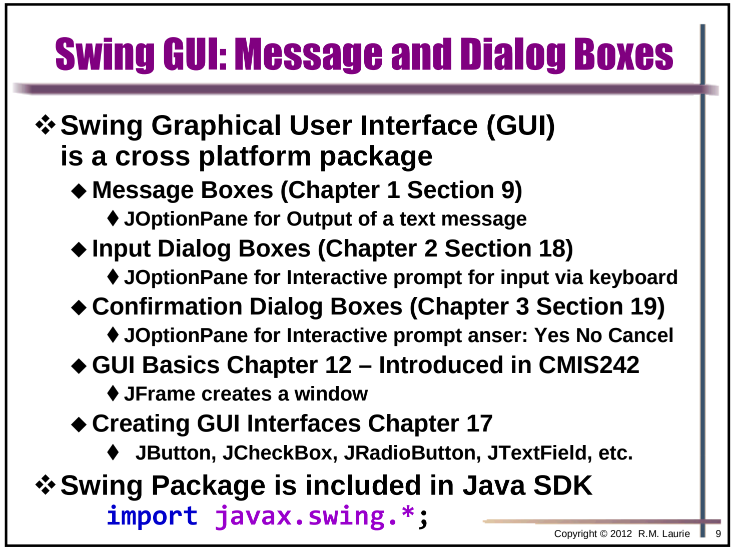
This output stream

Specifier	Output	Example	Result
%d	Decimal integer	%5d	Provides a minimum width of 5
%f	Floating point number	%8.2f	Provides a minimum width of 8 and display 2 places to right of decimal
%e	Scientific Notation	%10.3e	Provides a minimum width of 10 and display 2 places to right of decimal
%s	String	%12s	Provides a minimum with of 12
%b	Boolean value	%6b	Provides a minimum width of 6



```
import java.util.Scanner;
public class InvoiceTax
{
  public static void main(String[] args)
  {
    final double TAXRATE_STATE = 5.5;// Constant
    double dOrder, dPctDiscount, dDiscount;
    Scanner keyEntry = new Scanner(System.in);
    System.out.print("INVOICE TOTAL CALCULATOR\n"
      + "by Robert Laurie\nEnter the Order Total: $");
    dOrder = keyEntry.nextDouble();
    System.out.print("Enter vendor discount Rate: %");
    dPctDiscount = keyEntry.nextDouble();
    dDiscount = dOrder * dPctDiscount * 0.01;
    double dInvoice = dOrder - dDiscount;
    double dTax = dInvoice * TAXRATE_STATE/100;
    System.out.printf("   Order: $%9.2f\n", dOrder);
    System.out.printf("Discount: $%9.2f @ %5.2f%s\n",
                      dDiscount, dPctDiscount, "%");
    System.out.printf(" Invoice: $%9.2f\n", dInvoice);
    System.out.printf("     Tax: $%9.2f @ %5.2f%c\n",
                        dTax, TAXRATE_STATE, '%');
    System.out.printf("   TOTAL: $%9.2f\n", dInvoice + dTax);
  }
}
```

```
INVOICE TOTAL CALCULATOR
by Robert Laurie
Enter the Order Total: $23400
Enter vendor discount Rate: %10
   Order: $ 23400.00
Discount: $  2340.00 @ 10.00%
 Invoice: $ 21060.00
     Tax: $  1158.30 @  5.50%
   TOTAL: $ 22218.30
```

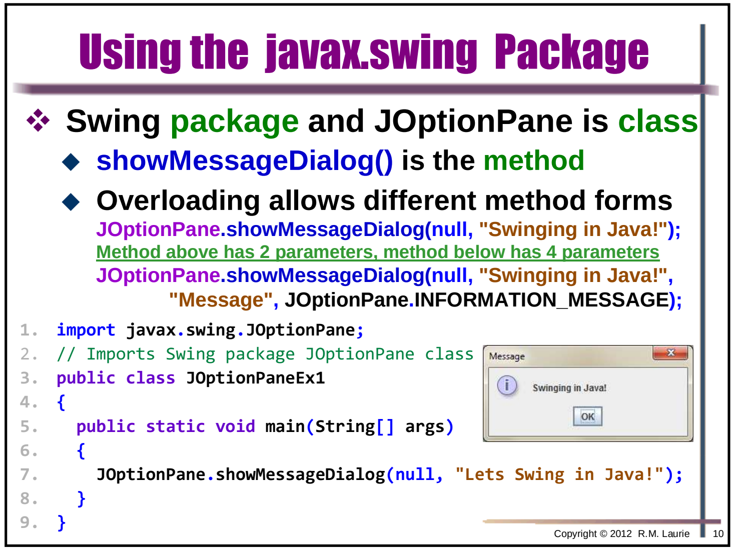


Swing GUI: Message and Dialog Boxes

- ❖ Swing Graphical User Interface (GUI) is a cross platform package
  - ◆ Message Boxes (Chapter 1 Section 9)
    - ♦ JOptionPane for Output of a text message
  - ◆ Input Dialog Boxes (Chapter 2 Section 18)
    - ♦ JOptionPane for Interactive prompt for input via keyboard
  - ◆ Confirmation Dialog Boxes (Chapter 3 Section 19)
    - ♦ JOptionPane for Interactive prompt anser: Yes No Cancel
  - ◆ GUI Basics Chapter 12 – Introduced in CMIS242
    - ♦ JFrame creates a window
  - ◆ Creating GUI Interfaces Chapter 17
    - ♦ JButton, JCheckBox, JRadioButton, JTextField, etc.
- ❖ Swing Package is included in Java SDK

```
import javax.swing.*;
```

Copyright © 2012 R.M. Laurie 9



Using the javax.swing Package

- ❖ Swing package and JOptionPane is class
  - ◆ showMessageDialog() is the method
  - ◆ Overloading allows different method forms

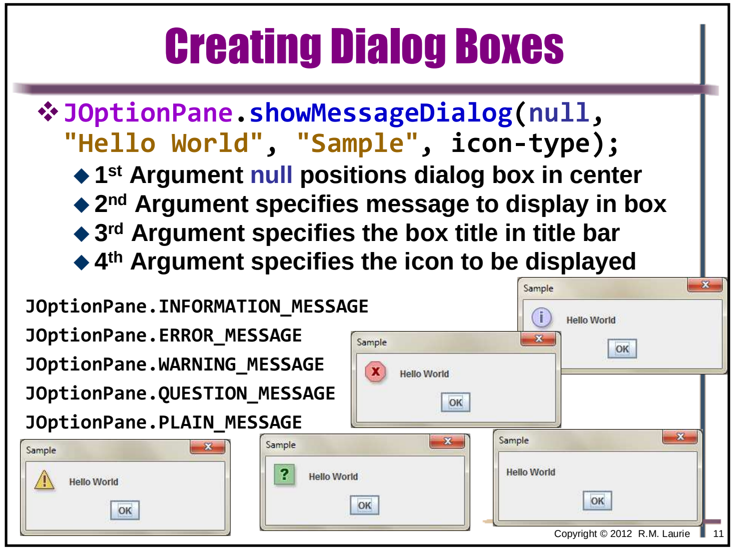
```
JOptionPane.showMessageDialog(null, "Swinging in Java!");
```

Method above has 2 parameters, method below has 4 parameters

```
JOptionPane.showMessageDialog(null, "Swinging in Java!",
        "Message", JOptionPane.INFORMATION_MESSAGE);
```

```
import javax.swing.JOptionPane;
// Imports Swing package JOptionPane class
public class JOptionPaneEx1
{
  public static void main(String[] args)
  {
    JOptionPane.showMessageDialog(null, "Lets Swing in Java!");
  }
}
```

Copyright © 2012 R.M. Laurie 10

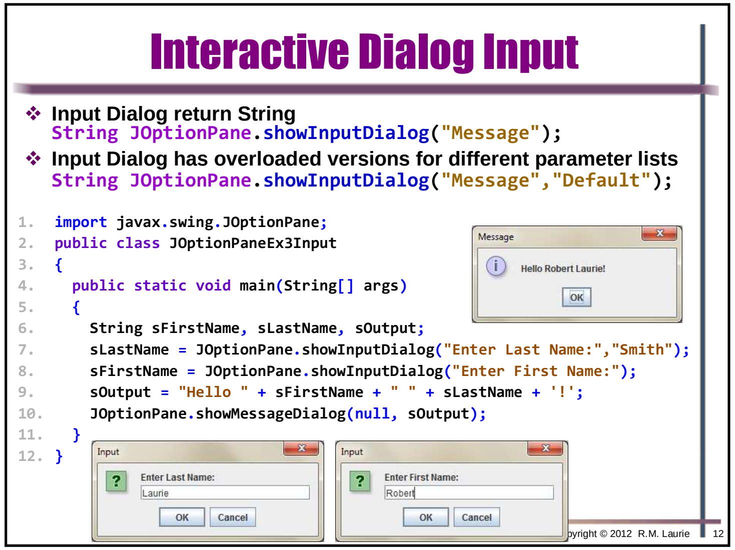


Creating Dialog Boxes

- ❖ `JOptionPane.showMessageDialog(null, "Hello World", "Sample", icon-type);`
  - ◆ 1st Argument null positions dialog box in center
  - ◆ 2nd Argument specifies message to display in box
  - ◆ 3rd Argument specifies the box title in title bar
  - ◆ 4th Argument specifies the icon to be displayed

```
JOptionPane.INFORMATION_MESSAGE
JOptionPane.ERROR_MESSAGE
JOptionPane.WARNING_MESSAGE
JOptionPane.QUESTION_MESSAGE
JOptionPane.PLAIN_MESSAGE
```

Copyright © 2012 R.M. Laurie 11



Interactive Dialog Input

- ❖ Input Dialog return String

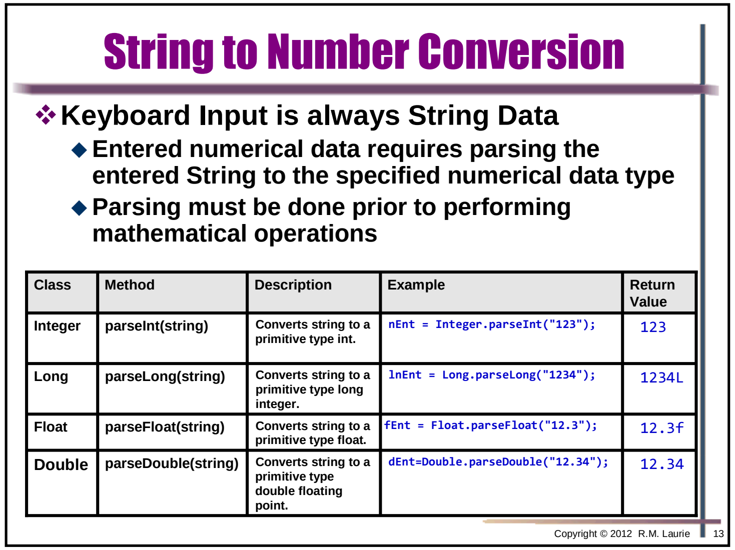
```
String JOptionPane.showInputDialog("Message");
```

- ❖ Input Dialog has overloaded versions for different parameter lists

```
String JOptionPane.showInputDialog("Message","Default");
```

```
import javax.swing.JOptionPane;
public class JOptionPaneEx3Input
{
  public static void main(String[] args)
  {
    String sFirstName, sLastName, sOutput;
    sLastName = JOptionPane.showInputDialog("Enter Last Name:","Smith");
    sFirstName = JOptionPane.showInputDialog("Enter First Name:");
    sOutput = "Hello " + sFirstName + " " + sLastName + '!';
    JOptionPane.showMessageDialog(null, sOutput);
  }
}
```

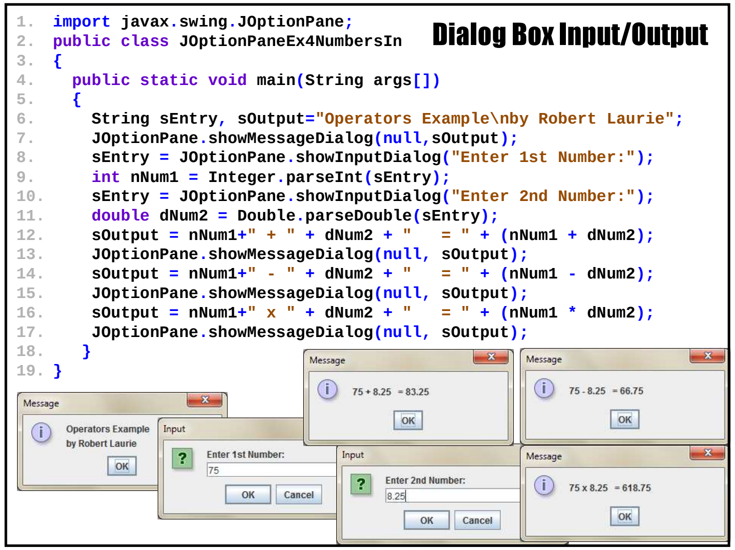
pyright © 2012 R.M. Laurie 12



String to Number Conversion

- ❖ Keyboard Input is always String Data
  - ◆ Entered numerical data requires parsing the entered String to the specified numerical data type
  - ◆ Parsing must be done prior to performing mathematical operations

Class	Method	Description	Example	Return Value
Integer	parseInt(string)	Converts string to a primitive type int.	`nEnt = Integer.parseInt("123");`	123
Long	parseLong(string)	Converts string to a primitive type long integer.	`lnEnt = Long.parseLong("1234");`	1234L
Float	parseFloat(string)	Converts string to a primitive type float.	`fEnt = Float.parseFloat("12.3");`	12.3f
Double	parseDouble(string)	Converts string to a primitive type double floating point.	`dEnt=Double.parseDouble("12.34");`	12.34



Dialog Box Input/Output

```
import javax.swing.JOptionPane;
public class JOptionPaneEx4NumbersIn
{
  public static void main(String args[])
  {
    String sEntry, sOutput="Operators Example\nby Robert Laurie";
    JOptionPane.showMessageDialog(null,sOutput);
    sEntry = JOptionPane.showInputDialog("Enter 1st Number:");
    int nNum1 = Integer.parseInt(sEntry);
    sEntry = JOptionPane.showInputDialog("Enter 2nd Number:");
    double dNum2 = Double.parseDouble(sEntry);
    sOutput = nNum1+" + " + dNum2 + "   = " + (nNum1 + dNum2);
    JOptionPane.showMessageDialog(null, sOutput);
    sOutput = nNum1+" - " + dNum2 + "   = " + (nNum1 - dNum2);
    JOptionPane.showMessageDialog(null, sOutput);
    sOutput = nNum1+" x " + dNum2 + "   = " + (nNum1 * dNum2);
    JOptionPane.showMessageDialog(null, sOutput);
  }
}
```

Message: Operators Example by Robert Laurie

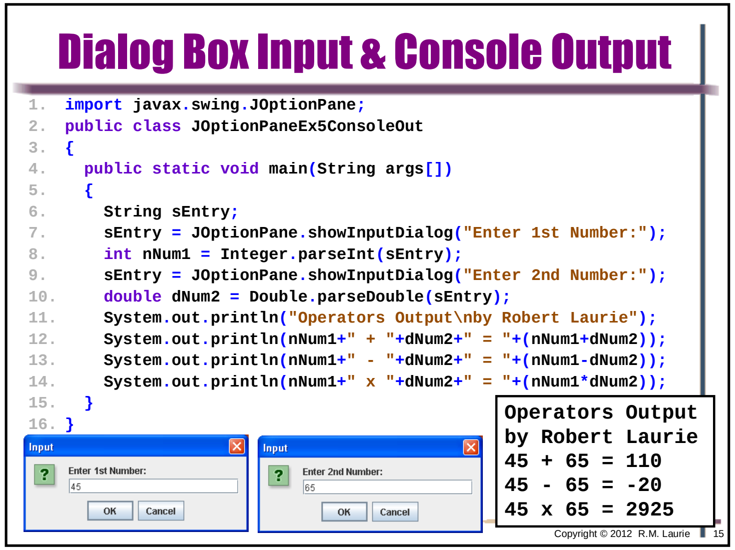
Input: Enter 1st Number: 75

Input: Enter 2nd Number: 8.25

Message: 75 + 8.25 = 83.25

Message: 75 - 8.25 = 66.75

Message: 75 x 8.25 = 618.75



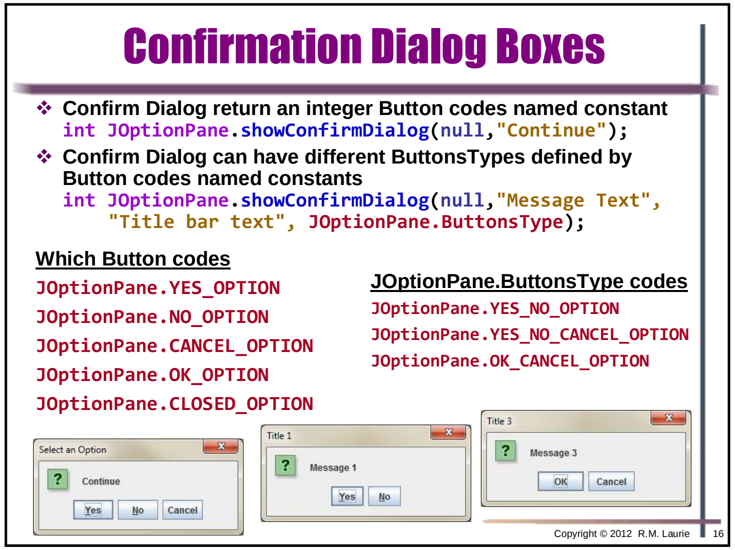
Dialog Box Input & Console Output

```
import javax.swing.JOptionPane;
public class JOptionPaneEx5ConsoleOut
{
  public static void main(String args[])
  {
    String sEntry;
    sEntry = JOptionPane.showInputDialog("Enter 1st Number:");
    int nNum1 = Integer.parseInt(sEntry);
    sEntry = JOptionPane.showInputDialog("Enter 2nd Number:");
    double dNum2 = Double.parseDouble(sEntry);
    System.out.println("Operators Output\nby Robert Laurie");
    System.out.println(nNum1+" + "+dNum2+" = "+(nNum1+dNum2));
    System.out.println(nNum1+" - "+dNum2+" = "+(nNum1-dNum2));
    System.out.println(nNum1+" x "+dNum2+" = "+(nNum1*dNum2));
  }
}
```

Input: Enter 1st Number: 45

Input: Enter 2nd Number: 65

```
Operators Output
by Robert Laurie
45 + 65 = 110
45 - 65 = -20
45 x 65 = 2925
```



Confirmation Dialog Boxes

- ❖ Confirm Dialog return an integer Button codes named constant
  `int JOptionPane.showConfirmDialog(null,"Continue");`
- ❖ Confirm Dialog can have different ButtonsTypes defined by Button codes named constants
  `int JOptionPane.showConfirmDialog(null,"Message Text", "Title bar text", JOptionPane.ButtonsType);`

Which Button codes

- JOptionPane.YES_OPTION
- JOptionPane.NO_OPTION
- JOptionPane.CANCEL_OPTION
- JOptionPane.OK_OPTION
- JOptionPane.CLOSED_OPTION

JOptionPane.ButtonsType codes

- JOptionPane.YES_NO_OPTION
- JOptionPane.YES_NO_CANCEL_OPTION
- JOptionPane.OK_CANCEL_OPTION

Select an Option: Continue

Title 1: Message 1

Title 3: Message 3