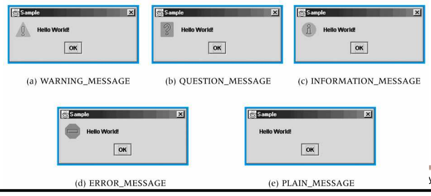
Using the javax.swing Package

- Classes in package provide means of specifying fully functional GUI with typical components such as:
  - Check boxes, text entry fields, and buttons
  - Dialog Boxes:
    - Modal = User must respond before continues
    - Modeless = No entry required
- Dialog Box class hierarchy
  - javax.swing.JComponet
    - javax.swing.JOptionPane
      (Derived class from javax.swing.JComponet)

Creating Dialog Boxes

- Use JOptionPane Class
  - Call showMessageDialog Method
- `JOptionPane.showMessageDialog(null, "message", "title", 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



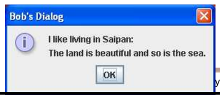
(a) WARNING_MESSAGE (b) QUESTION_MESSAGE (c) INFORMATION_MESSAGE

(d) ERROR_MESSAGE (e) PLAIN_MESSAGE

Import statement

- At beginning of program after package declaration
- Compiler searches for classes in import packages

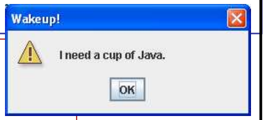
```
import javax.swing.*;
  public class DialogBoxExample1
  {
    public static void main(String[] args)
    {
      JOptionPane.showMessageDialog(null,
              "I like living in Saipan:"
             +"\nThe land is beautiful and so is the sea."
             ,"Bob\'s Dialog",
              JOptionPane.INFORMATION_MESSAGE);
      System.exit(0);
    }
  }
```



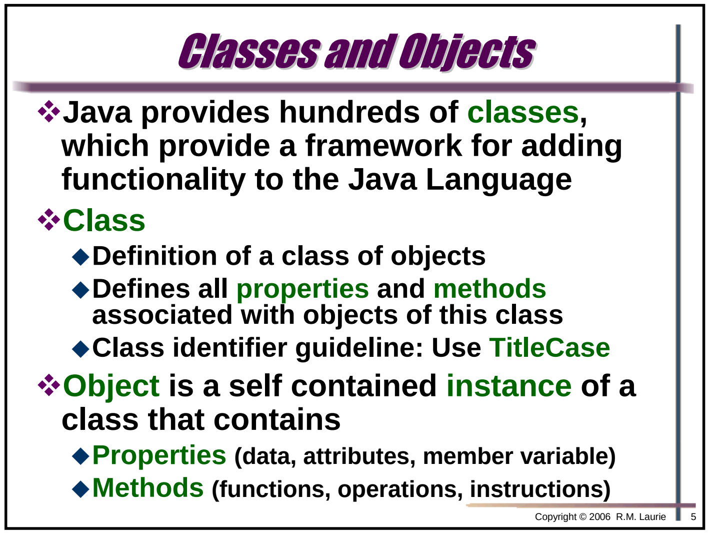
I like living in Saipan:
The land is beautiful and so is the sea.

```
import javax.swing.*;
public class ShowBoxCls
{
  private String sMessage;
  ShowBoxCls()
  {
    sMessage = "I need a cup of Java.";
  }
  public void displayMessage()
  {
    JOptionPane.showMessageDialog(null,sMessage,
   "Wakeup!",JOptionPane.WARNING_MESSAGE);
  }
}
```

```
public class ShowBoxPrg
{
  public static void main(String[] args)
  {
    // Create a variable of type ShowMessageCls
      ShowBoxCls oMessageOne;
    // Create an object of the ShowMessageCls
      oMessageOne = new ShowBoxCls();
    // Call the method for the object
      oMessageOne.displayMessage();
  }
}
```

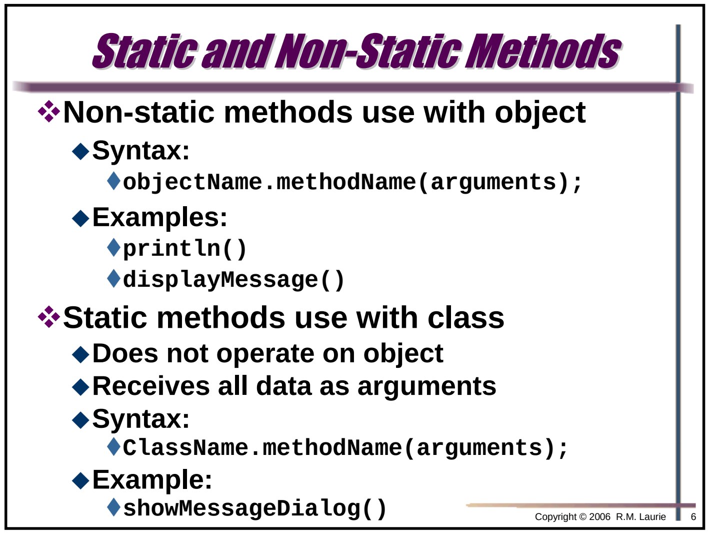


I need a cup of Java.



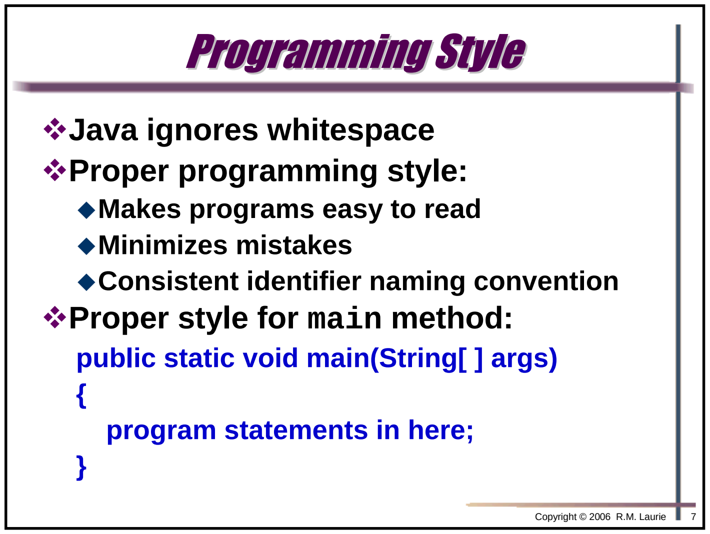
Classes and Objects

- Java provides hundreds of classes, which provide a framework for adding functionality to the Java Language
- Class
  - Definition of a class of objects
  - Defines all properties and methods associated with objects of this class
  - Class identifier guideline: Use TitleCase
- Object is a self contained instance of a class that contains
  - Properties (data, attributes, member variable)
  - Methods (functions, operations, instructions)



Static and Non-Static Methods

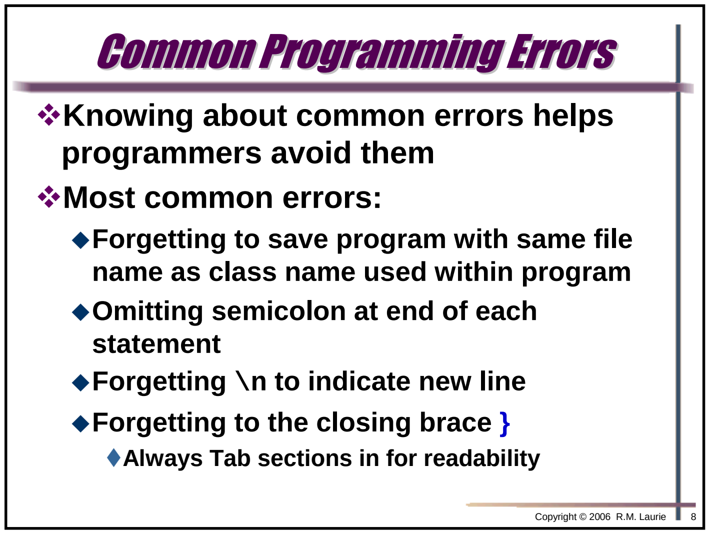
- Non-static methods use with object
  - Syntax:
    - `objectName.methodName(arguments);`
  - Examples:
    - `println()`
    - `displayMessage()`
- Static methods use with class
  - Does not operate on object
  - Receives all data as arguments
  - Syntax:
    - `ClassName.methodName(arguments);`
  - Example:
    - `showMessageDialog()`



Programming Style

- Java ignores whitespace
- Proper programming style:
  - Makes programs easy to read
  - Minimizes mistakes
  - Consistent identifier naming convention
- Proper style for `main` method:

```
public static void main(String[ ] args)
{
   program statements in here;
}
```



Common Programming Errors

- Knowing about common errors helps programmers avoid them
- Most common errors:
  - Forgetting to save program with same file name as class name used within program
  - Omitting semicolon at end of each statement
  - Forgetting `\n` to indicate new line
  - Forgetting to the closing brace }
    - Always Tab sections in for readability