Java Intro
Terminology

- Program – a collection of cooperating objects that accomplishes a particular task.
- object – an instance of a class, with its own copy of the state information.
- class – a description of a particular kind of object, including state information maintained by each instance and method definitions that describe the behavior of objects of this kind.
Example: Square

Square class

Attributes
  size
  color
  xPosition, yPosition

Methods
  changeSize
  moveHorizontal
  moveVertical

State

Behavior
• high-level language – computer language intended for use by humans in constructing programs
• machine language – the low level language implemented in a CPU's hardware. Unique to the CPU type.
• compiler – a program that translates a high-level language program into the machine language of a particular CPU type.
/**
 * Write a description of class TouchyButton here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */

import squint.*;
import javax.swing.*;

public class TouchyButton extends GUIManager {

    private final int WINDOW_WIDTH = 150;
    private final int WINDOW_HEIGHT = 300;

    public TouchyButton() {
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );
        contentPane.add( new JButton( "Click Here" ) );
    }

    public void buttonClicked() {
        contentPane.add( new JLabel( "I'm Touched\n!" ) );
    }

}

Libraries used by our Program
Class header
Squint class we will inherit from
Constructor
data fields
or attributes
Event Handler
/**
 * Write a description of class TouchyButton here.
 * 
 * @author (your name)
 * @version (a version number or a date)
 */

import squint.*;
import javax.swing.*;

public class TouchyButton extends GUIManager {

    private final int WINDOW_WIDTH = 150;
    private final int WINDOW_HEIGHT = 300;

    public TouchyButton() {
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );
        contentPane.add( new JButton( "Click Here" ) );
    }

    public void buttonClicked( ) {
        contentPane.add( new JLabel( "I'm Touched\n!" ) );
    }
}
Method's access. “public” means any other class can call this method.

Method’s “return type” – “void” means no value is produced.

parameter list – empty, so we do not need to provide any information to call this method.
Instance Variables

• If you need to use a value or an object in multiple places in your class, use an *instance variable* to “remember” it.
• Declare instance variables outside method declarations
• Each new object (instance) created from a class has its own copy of the instance variables
• Instance variables are accessible in all of the methods of the class.
• Local variables are like instance variables, but are declared and used within a single method.
import squint.*;
import javax.swing.*;

class AddressWindow extends GUIManager {

    private final int WINDOW_WIDTH = 470;
    private final int WINDOW_HEIGHT = 110;

    AddressWindow () {
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT);
    
        contentPane.add( new JLabel( "Street Addr:" ) );
        contentPane.add( new JTextField( 30 ) );

        contentPane.add( new JLabel( "City:" ) );
        contentPane.add( new JTextField( 15 ) );

        contentPane.add( new JLabel( "State:" ) );
        contentPane.add( new JTextField( 2 ) );

        contentPane.add( new JLabel( "Zip:" ) );
        contentPane.add( new JTextField( 5 ) );
    }
}
import squint.*;
import javax.swing.*;

public class PopUpMenu extends GUIManager {

    private final int WINDOW_WIDTH = 400;
    private final int WINDOW_HEIGHT = 110;

    public PopUpMenu () {
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT);
        contentPane.add( new JComboBox( new String[] {"Yes", "No", "Maybe So"} ) );
    }
}
import squint.*;
import javax.swing.*;

public class PasswordField extends GUIManager {

    private final int WINDOW_WIDTH = 300;
    private final int WINDOW_HEIGHT = 100;

    public PasswordField () {
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT);
        contentPane.add( new JLabel( "Password:" ) );
        contentPane.add( new JPasswordField( 10 ) );
    }
}

• class header – Java program line that starts a class definition; contains the name of the class and also tells if the class is derived from any existing class.

• constructor – method called to create a new object from a class definition. Has same name as the class.

• method definition – a sequence of instructions that can be executed by referring to the associated method name.
• field – a piece of data belonging to an object. A name is associated with each field. (Also: data member, attribute)

• state – the state of an object is the current value of all its fields.

• behavior – what an object can do, determined by what methods are defined in the class from which the object was created.