Guidelines for “Mini-project”

• During the remainder of the semester, we will be doing a small design (and implementation) project.
• Students may work alone or in groups of two.
• Students may choose their own partner, if desired.
  – You must notify me of your partnership via email no later than Friday, Nov. 12.
• We will attempt to facilitate partner arrangements for students who want a partner but have not found one on their own.
  – You must notify me via e-mail of your desire to be paired with a partner, no later than Friday, Nov. 12.

More about the Project

• The project will involve in iterative development of a drawing application.
• The project will make use of the Java component libraries--e.g. Swing
• The project will involve, both design and implementation in Java.
• Initial stages of the development will be done in class as an extended case study.
• This will provide the basic design framework and introduce the necessary Java constructs.
Still More About the Project

- **Timeline**
  - Monday, Nov. 15--Initial Assignment: Implementation of an early prototype
    - Due date: Monday, Nov. 22
    - Most of the design and code for this assignment will be developed in class.
  - Monday, Nov. 22--Project Assignment: Addition of features to the in-class case study.
    - Due date: Friday, Dec. 10

A Case Study of Object Oriented Design and Implementation

- **Objectives:**
  - To provide a comprehensive example of object oriented design.
  - To show effective use of inheritance and polymorphism.
  - To illustrate the practical use of patterns during design.
  - To show the use of standard component libraries
  - To demonstrate an iterative development process and the value of early prototyping.
Case Study--Problem Statement

**Problem:** to design and implement a simple drawing application. The application provides a drawing window, consisting of a drawing “canvas”, a menu, a toolbar and a control panel. The toolbar allows the user to select tools to draw various objects on the canvas. Tools are provided for: freehand forms, simple geometric shapes (lines, ovals, rectangles), and text. Ovals and rectangles may be filled or unfilled. A tool is also provided to “erase” *areas of the canvas*. Tools can be selected from a menu in addition to the toolbar. The control panel allows a user to clear the drawing canvas and select various pen colors. A help menu is also provided. The application is to be developed in Java, using Swing components. It should be runnable either as an application or an applet.

MiniDraw--Prototype User Interface

![Image of MiniDraw prototype user interface with tools and canvas showing a drawing of a fox]

The quick brown fox
MiniDraw--Conceptual Model

MiniDraw--Iterative Development

- Proposed design iterations:
  - First Iteration: Develop a simple prototype drawing window that provides only a drawing canvas (no toolbar, control panel, or menus) and only supports freehand drawing.
  - Second Iteration: Add the control panel with clear button and pen color selection.
  - Third iteration: Add toolbar, menu, and tools for lines, geometric shapes (unfilled), and erasing.
  - Fourth iteration: Add text tool and filled geometric shapes
MiniDraw--First Iteration

• Key design issues to be addressed
  – Use of Swing components
    • establishing a drawing canvas
    • understanding basics of Java/Swing graphics
  – Basic drawing functionality
    • coupling mouse movement to drawing operations
  – Application vs. Applet issues

Assignment for Friday

Read the introductory material on Swing at the following URL:

http://www.javasoft.com/products/jfc/tsc/what_is_swing/what_is_swing.html
Brief Overview of Java Swing

- Swing is a library of components (classes) for building interactive graphical user interfaces. It is part of the Java Foundation Classes (JFC).

- Swing consists of:
  - GUI widgets--frames, buttons, labels, checkboxes, etc
  - Layout Managers--control layout of widgets in windows
  - Events and event listeners--connect GUI events to actions
  - Graphics and imaging classes--allow widgets to draw their visual appearances and display graphics and images.
Swing Overview-Continued

Component

Container

The stuff on the previous slide

JPanel

JApplet

JComponent

Window

Frame

Dialog

Swing Overview: Layout Managers

Container

Layout Manager

BorderLayout

GridLayout

FlowLayout

CardLayout

LayoutManager2

GridLayout

Note: can also do custom layouts
java.awt.event Events

- ActionEvent
- ContainerEvent
- AdjustmentEvent
- FocusEvent
- ComponentEvent
- AbstractInputEvent
- ItemEvent
- PaintEvent
- TextEvent
- WindowEvent
- KeyEvent
- MouseEvent
java.awt.event Listener Interfaces

- ActionListener
- AdjustmentListener
- ComponentListener
- FocusListener
- ItemListener
- KeyListener
- MouseListener
- MouseMotionListener
- TextListener
- WindowListener

Java.awt. Graphics Class

- Provides an abstraction for drawing and representing graphical images.

  Graphics g = new Graphics( );

  Example methods:
  
g.setColor(Color color)
g.setFont(font)
g.drawString(String s, int x, int y)
g.drawLine(int x1, int y1, int x2, int y2)
g.drawRect(int x, int y, int w, int h)
g.drawOval(int x, int y, int w, int h)
Displaying Graphics

- Java Components have an associated graphics context (an object of class Graphics)
- The graphics context of a component can be accessed via the getGraphics() method of the component interface
- The visual display of a component can be updated by invoking the repaint() of the component interface.

MiniDraw--First Iteration

```plaintext
JComponent
   ↓
DrawingCanvas
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListener
   ↓
MiniDraw
   ↓
MouseListener
   ↓
DrawingCanvasListe
```
# MiniDraw--First Iteration Use Case

**UseCase Name:** Draw Freehand  
**Actor:** User  
**Typical Course of Events:**

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Use case begins when user positions mouse over canvas and depresses mouse button.</td>
<td>Begin drawing</td>
</tr>
<tr>
<td>3. User moves mouse with button depressed to sketch form on screen</td>
<td>Draw line on canvas that corresponds to mouse movement.</td>
</tr>
</tbody>
</table>