Solution to Section #7

Portions of this handout by Eric Roberts and Nick Troccoli

1. Using Interactors

```
* File: Interactors.java
 * This program allows the user to create a set of labels and then drag
 * them around in the window.
import acm.graphics.*;
import acm.program.*;
import java.util.*;
import java.awt.event.*;
import javax.swing.*;
public class Interactors extends GraphicsProgram {
  public void init() {
      contents = new HashMap<String,GLabel>();
      createController();
      addActionListeners();
      addMouseListeners();
   }
   /* Creates the control strip at the bottom of the window */
  private void createController() {
      nameField = new JTextField(MAX NAME);
      nameField.addActionListener(this);  // Detects ENTER key pressed
      addButton = new JButton("Add");
      removeButton = new JButton("Remove");
      clearButton = new JButton("Clear");
      add(new JLabel("Name"), SOUTH);
      add(nameField, SOUTH);
      add(addButton, SOUTH);
      add(removeButton, SOUTH);
      add(clearButton, SOUTH);
   }
   /* Adds a label with the given name at the center of the window */
  private void addLabel(String name) {
      GLabel label = new GLabel(name);
      double labelX = getWidth() / 2.0 - label.getWidth() / 2.0;
      double labelY = getHeight() / 2 + label.getAscent() / 2.0;
      add(label, labelX, labelY);
      contents.put(name, label);
   }
```

```
/* Removes the label with the given name */
private void removeLabel(String name) {
   GLabel obj = contents.get(name);
   if (obj != null) {
      remove(obj);
}
/* Removes all labels in the contents table */
private void removeContents() {
   for (String labelName : contents.keySet()) {
      removeLabel(labelName);
                        // Clear all entries in the hashmap
   contents.clear();
}
/* Called in response to button actions */
public void actionPerformed(ActionEvent e) {
   Object source = e.getSource();
   // Detect both clicks and ENTER for adding a new label
   if (source == addButton || source == nameField) {
      addLabel(nameField.getText());
   } else if (source == removeButton) {
      removeLabel(nameField.getText());
   } else if (source == clearButton) {
      removeContents();
}
/* Called on mouse press to record the coordinates of the click */
public void mousePressed(MouseEvent e) {
   last = new GPoint(e.getPoint());
   currentLabel = (GLabel)getElementAt(last);
}
/* Called on mouse drag to reposition the object */
public void mouseDragged(MouseEvent e) {
   if (currentLabel != null) {
      currentLabel.move(e.getX() - last.getX(),
         e.getY() - last.getY());
      last = new GPoint(e.getPoint());
   }
}
/* Private constants */
private static final int MAX NAME = 25;
/* Private instance variables */
private HashMap<String,GLabel> contents;
private JTextField nameField;
private JButton addButton;
private JButton removeButton;
private JButton clearButton;
private GLabel currentLabel;
private GPoint last;
```

2. The Employee Class

```
* File: Employee.java
 * Class which describes the Employee variable type.
 * An Employee has the following information:
         - title
         - salary
 * They may be given a promotion, which adds the word "Senior"
 * to their job title and doubles their salary.
public class Employee {
  public Employee(String newName, String newTitle) {
      name = newName;
      title = newTitle;
   public String getTitle() {
      return title;
  public void setTitle(String title) {
      this.title = title;
  public int getSalary() {
      return salary;
   public void setSalary(int salary) {
      this.salary = salary;
  public String getName() {
      return name;
   // Adds "Senior" to the front of our job title, and doubles our salary
  public void promote() {
      title = "Senior " + title;
      salary *= 2;
   }
   /* Employee instance variables */
   private String name;
  private String title;
  private int salary;
```

2. Subclassing GCanvas

```
* File: RandomCirclesCanvas.java
* This GCanvas subclass adds the ability to also draw random circles.
* Each circle has a randomly chosen color, a randomly chosen
\star radius between 5 and 50 pixels, and a randomly chosen
\star position on the canvas, subject to the condition that
* the entire circle must fit inside the canvas without
* extending past the edge.
import acm.graphics.*;
import acm.util.*;
public class RandomCirclesCanvas extends GCanvas {
   /** Minimum radius */
  private static final double MIN_RADIUS = 5;
   /** Maximum radius */
  private static final double MAX RADIUS = 50;
  public void drawRandomCircle() {
      double r = rgen.nextDouble(MIN RADIUS, MAX RADIUS);
      double x = rgen.nextDouble(0, getWidth() - 2 * r);
      double y = rgen.nextDouble(0, getHeight() - 2 * r);
      GOval circle = new GOval(x, y, 2 * r, 2 * r);
      circle.setFilled(true);
      circle.setColor(rgen.nextColor());
      add(circle); // adds it to ourself!
   }
   /* Private instance variable */
   private RandomGenerator rgen = RandomGenerator.getInstance();
```