Mehran Sahami
CS 106A

Handout #38A
November 14, 2007

Solution to Section #7

Based on a handout by Eric Roberts

```
// File: BoxDiagram.java
// This program allows the user to create a set of boxes with 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 BoxDiagram extends GraphicsProgram {

/* Initializes the program */
   public void init() {
      contents = new HashMap<String,GObject>();
      createController();
      addActionListeners();
      addMouseListeners();
   }

/* Creates the control strip at the bottom of the window */
   private void createController() {
      nameField = new JTextField(MAX_NAME);
      nameField.addActionListener(this);
      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 box with the given name at the center of the window */
   private void addBox(String name) {
      GCompound box = new GCompound();
      GRect outline = new GRect(BOX_WIDTH, BOX_HEIGHT);
      GLabel label = new GLabel(name);
      box.add(outline, -BOX_WIDTH / 2, -BOX_HEIGHT / 2);
      box.add(label, -label.getWidth() / 2, label.getAscent() / 2);
      add(box, getWidth() / 2, getHeight() / 2);
      contents.put(name, box);
   }

/* Removes the box with the given name */
   private void removeBox(String name) {
      GObject obj = contents.get(name);
      if (obj != null) {
         remove(obj);
      }
   }
```

```
/* Removes all boxes in the contents table */
   private void removeContents() {
      Iterator<String> iterator = contents.keySet().iterator();
      while (iterator.hasNext()) {
         removeBox(iterator.next());
      }
      contents.clear();    // Clear all entries in the hashmap
   }

/* Called in response to button actions */
   public void actionPerformed(ActionEvent e) {
      Object source = e.getSource();
      if (source == addButton || source == nameField) {
         addBox(nameField.getText());
      } else if (source == removeButton) {
         removeBox(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());
      currentObject = getElementAt(last);
   }

/* Called on mouse drag to reposition the object */
   public void mouseDragged(MouseEvent e) {
      if (currentObject != null) {
         currentObject.move(e.getX() - last.getX(),
                            e.getY() - last.getY());
         last = new GPoint(e.getPoint());
      }
   }

/* Called on mouse click to move this object to the front */
   public void mouseClicked(MouseEvent e) {
      if (currentObject != null) currentObject.sendToFront();
   }

/* Private constants */
   private static final int MAX_NAME = 25;
   private static final double BOX_WIDTH = 120;
   private static final double BOX_HEIGHT = 50;

/* Private instance variables */
   private HashMap<String,GObject> contents;
   private JTextField nameField;
   private JButton addButton;
   private JButton removeButton;
   private JButton clearButton;
   private GObject currentObject;
   private GPoint last;

}
```