Carlos Guisao

Homework 4

a) What is the purpose of a design pattern?

The purpose of a design pattern is to provide general solutions to design problems that occur frequently in OOD.

b) When do you apply the Observer pattern?

The observer pattern should be used when a change of a state in one object must be reflected in another object without keeping the objects tightly coupled. While leaving the door open to future enhancements like adding more observers with minimal change.

c) You review a design written by somebody else for an application and you find these:

- an interface Shape with a method draw()
- a class Circle that implements Shape
- a class Rectangle that implements Shape
- a class CompoundShape that:
    - implements interface Shape
    - aggregates 0 or more Shape objects,
    - has an extra method called add(Shape sh)
    - for implementing method draw() calls the draw() method for all aggregated Shape objects.

You assume that a CompoundShape object is made of multiple shapes.

What design pattern is at work in this application? Explain your answer.

The composite design pattern is being used here in this example. The intent of a composite pattern is to group components into a whole which is what the CompoundShape class is trying to achieve. Example add(Shape sh) method.

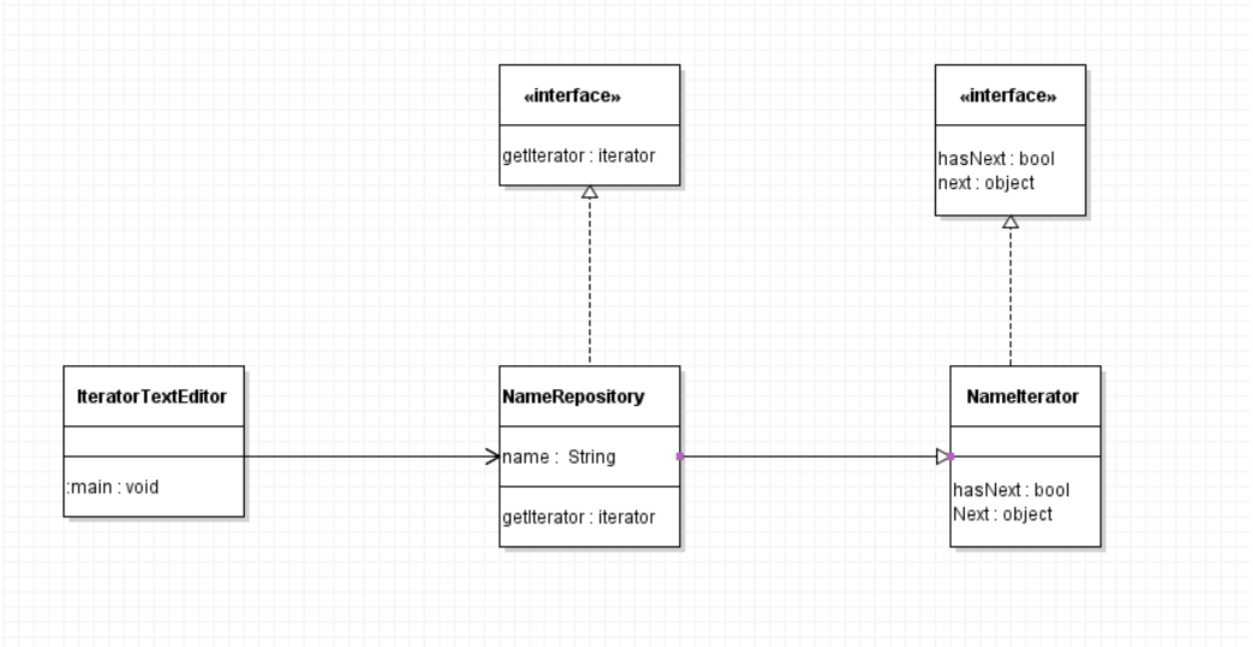
d) The TitledBorder class can give a title to a border. Consider the code

The pattern at work here in the strategy pattern because the method is being invoked with instances of other classes without declaration. In this example the context class Title Border calls the selected method of the strategy object EtchedBorder.

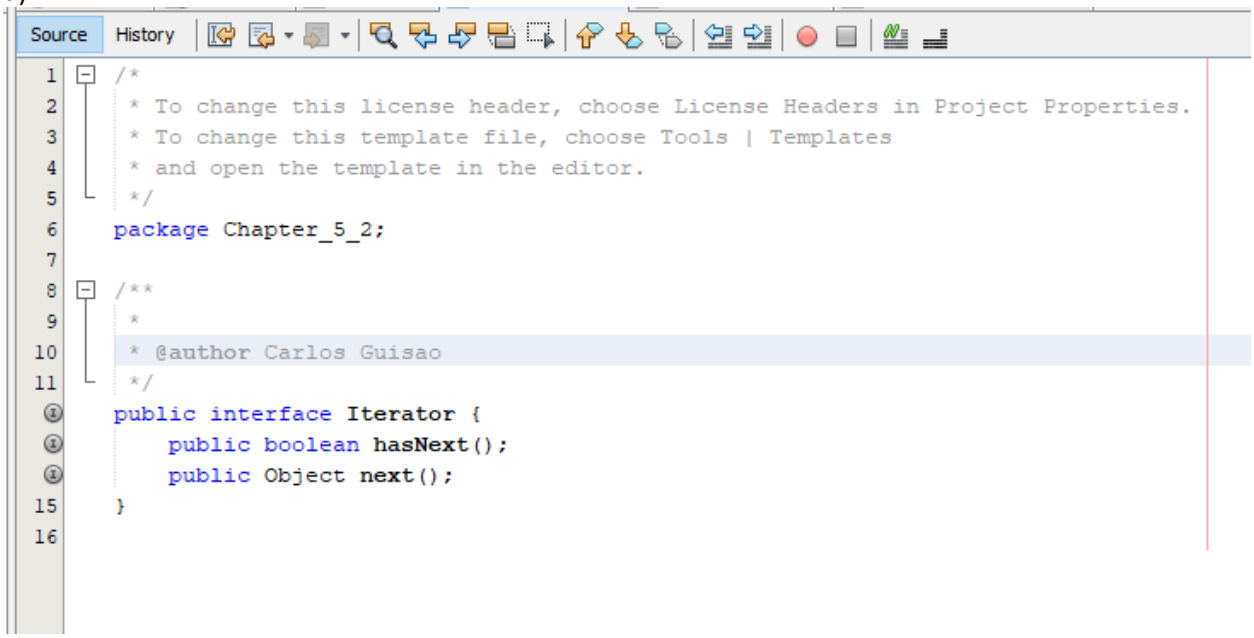
5.2.

a) I would use an iterator pattern because this example requires different versions of a spell-checking algorithm to be considered. This pattern makes the most sense for code that is maintainable over a period.

b)



d)

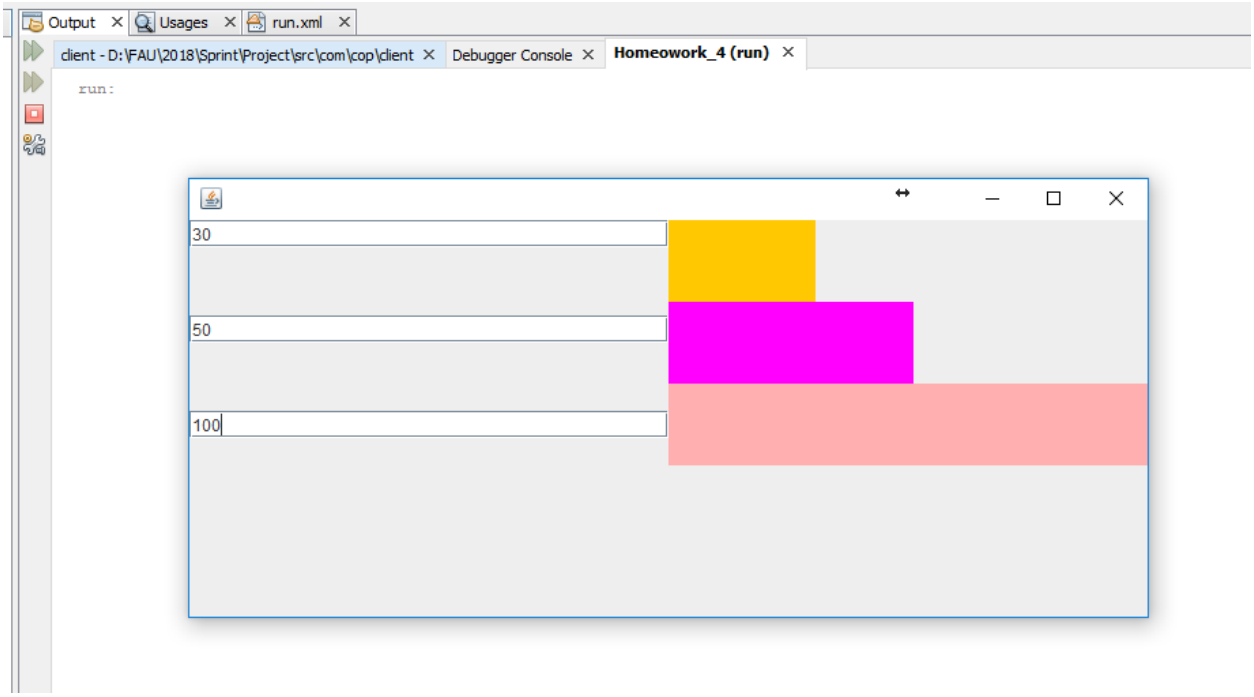


```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_5_2;

/**
 *
 * @author Carlos Guisao
 */
public interface Iterator {
    public boolean hasNext();
    public Object next();
}
```

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_5_2;

/**
 *
 * @author Carlos Guisao
 */
public interface Container {
    public Iterator getIterator();
}
```

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_5_2;

/**
 *
 * @author Carlos Guisao
 */
public class NameRepository implements Container {

    @Override
    public Iterator getIterator() {
        throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
    }

    private class NameIterator implements Iterator
    {

        @Override
        public boolean hasNext() {
            throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
        }

        @Override
        public Object next() {
            throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
        }

    }

}
```

5.3



Output | Usages | run.xml | Main.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_5;

/**
 *
 * @author Carlos Guisao
 */
public class Main {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        ControllerOld model = new ControllerOld();
        View view = new View(model);
        model.addObserver(view);
    }

}
```

Output | Usages | run.xml | Main.java | View.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_5;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.GridLayout;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.util.Observable;
import java.util.Observer;
import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.WindowConstants;

/**
 *
 * @author Carlos Guisao
 */
public class View implements Observer {
    private final ControllerOld controller;
    private JTextField textField_1, textField_2, textField_3;
    private JPanel textPanel, graphPanel;
    private Rectangle progressiveBar_1, progressiveBar_2, progressiveBar_3;
    private JLabel label_1, label_2, label_3;

    public View(ControllerOld m) {
        controller = m;
        Bars();
        Fields();
        Panels();
        Frames();
    }

```

Output | Usages | run.xml | Main.java | View.java

Source | History

```java
    private void Bars() {
        progressiveBar_1 = new Rectangle(Color.ORANGE);
        progressiveBar_2 = new Rectangle(Color.MAGENTA);
        progressiveBar_3 = new Rectangle(Color.pink);
        label_1 = new JLabel(progressiveBar_1);
        label_2 = new JLabel(progressiveBar_2);
        label_3 = new JLabel(progressiveBar_3);
    }

    private void ModelRefactor(int barIndex, String value) {
        try {
            controller.SetBar(barIndex, value);
        } catch (IllegalArgumentException e) {
            System.out.println("Invalid value: '" + value + "' " + e);
        }
    }

    private void Fields() {
        textField_1 = Controller(()->ModelRefactor(0, textField_1.getText()));
        textField_2 = Controller(()->ModelRefactor(1, textField_2.getText()));
        textField_3 = Controller(()->ModelRefactor(2, textField_3.getText()));

    }

    private void Panels() {
        textPanel = new JPanel(); graphPanel = new JPanel();
        textPanel.setLayout(new BoxLayout(textPanel, BoxLayout.PAGE_AXIS));
        graphPanel.setLayout(new BoxLayout(graphPanel, BoxLayout.PAGE_AXIS));
        textPanel.add(textField_1);
        textPanel.add(Box.createRigidArea(new Dimension(0, 50)));
        textPanel.add(textField_2);
        textPanel.add(Box.createRigidArea(new Dimension(0, 50)));
        textPanel.add(textField_3);
        graphPanel.add(label_1);
        graphPanel.add(label_2);
        graphPanel.add(label_3);
    }
```

```java
    private void Frames() {
        JFrame frame = new JFrame();
        frame.setPreferredSize(new Dimension(720, 330));
        frame.setLayout(new GridLayout(1, 2));
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        frame.add(textPanel);
        frame.add(graphPanel);
        frame.pack();
        frame.setVisible(true);
    }

    @Override
    public void update(Observable o, Object ol) {
        int[] values = controller.GetBar();
        progressiveBar_1.SetWidth(values[0]);
        progressiveBar_2.SetWidth(values[1]);
        progressiveBar_3.SetWidth(values[2]);
        label_1.revalidate(); label_1.repaint();
        label_2.revalidate(); label_2.repaint();
        label_3.revalidate(); label_3.repaint();
    }

    private JTextField Controller(final Runnable function) {
        JTextField newField = new JTextField();
        newField.setMaximumSize(new Dimension(Integer.MAX_VALUE, newField.getPreferredSize().height));
        newField.addKeyListener(new KeyListener() {
            @Override
            public void keyTyped(KeyEvent e) {}

            @Override
            public void keyPressed(KeyEvent e) {}

            @Override
            public void keyReleased(KeyEvent e) {
                function.run();
            }
        });
        return newField;
    }
}
```

Chapter 6

6.1

a) Explain the purpose of abstract classes in no more than 15 lines.

The purpose of an abstract class is to define the basic functionality while leaving certain methods or parts undefined. By doing this abstract classes function as the base class. All undefined areas are left up to the person extending the class to define. Abstract classes are also used in situations where you want to design a class but do not want to allow anyone to make an object of that class. By making a class abstract you achieve this feature.

b) Give an example for a situation when an abstract class cannot be used in a Java program and an interface is the only choice.

An example of when you cannot use an abstract class in a java program is when you want to take advantage of multiple inheritance type. Like the HashMap class in JDK that implements several interfaces like Serializable and Cloneable.

c) GeneralPath collects shapes and is itself a shape. What design pattern does it implement? Explain.

The design pattern here at work is a composite pattern. GernalPath collects shapes and is also a shape this is a component/container problem discussed in class. Therefore, the pattern working here is the composite pattern.

6.2

```
run:
Manager Information
Full Name: Carlos Guisao
Salary: 200

BUILD SUCCESSFUL (total time: 0 seconds)
```

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_1;

/**
 *
 * @author Carlos Guisao
 */
public class main {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {

        //Test implemented here.
        Manager manager = new Manager();

        manager.addEmployee("Carlos", "Guisao");
        manager.addSalary(150);
        manager.addBonus(50);
        System.out.println(manager.toString());

    }
}
```

Output | Usages | run.xml | main.java | TemplateEmployee.java | Manager.java

Source | History

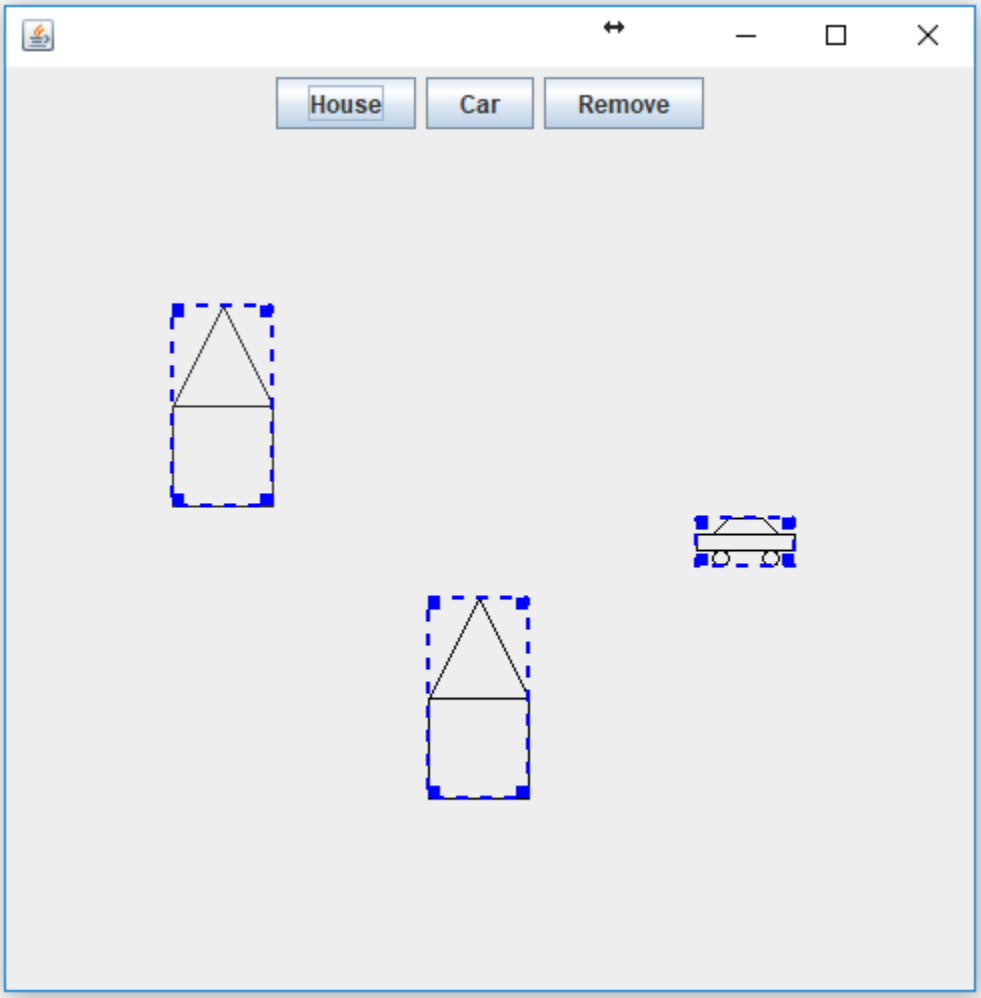
```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_1;

/**
 *
 * @author Carlos Guisao
 */
public class Manager extends TemplateEmployee {

    @Override
    public void addEmployee(String Name, String LastName) {
        name = Name;
        lastName = LastName;
    }

    @Override
    public void addSalary(int Salary) {
        salary = Salary;
    }

    @Override
    public void addBonus(int Bonus) {
        bonus = Bonus;
    }

    @Override
    public String toString() {
        return ("Manager Information \n" +
                "Full Name: " + name + " " + lastName + "\n" +
                "Salary: " + getSalary() + "\n");
    }

    @Override
    public int getSalary() {
        return salary + bonus;
    }
}
```

Output | Usages | run.xml | main.java | TemplateEmployee.java | Manager.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_1;

/**
 *
 * @author Carlos Guisao
 */
public abstract class TemplateEmployee {

    protected String name;
    protected String lastName;
    protected int salary;
    protected int bonus;

    public final void createEmployee()
    {
        addEmployee(name, lastName);
        addSalary(salary);
        addBonus(bonus);
        toString();
        getSalary();
    }
    public abstract void addEmployee(String name, String lastName);
    public abstract void addSalary(int salary);
    public abstract void addBonus(int bonus);
    public abstract int getSalary();
    public abstract String toString();
}
```

6.2



Output | Usages | run.xml | SceneEditor.java

Source | History

```java

/**
 *
 * @author Carlos Guisao
 */
public class SceneEditor
{
    public static void main(String[] args)
    {
        JFrame frame = new JFrame();
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

        final SceneComponent scene = new SceneComponent();

        JButton houseButton = new JButton("House");
        houseButton.addActionListener((ActionEvent event) -> {
            scene.add(new HouseShape(20, 20, 50));
        });

        JButton carButton = new JButton("Car");
        carButton.addActionListener((ActionEvent event) -> {
            scene.add(new CarShape(20, 20, 50));
        });

        JButton removeButton = new JButton("Remove");
        removeButton.addActionListener((ActionEvent event) -> {
            scene.removeSelected();
        });

        JPanel buttons = new JPanel();
        buttons.add(houseButton);
        buttons.add(carButton);
        buttons.add(removeButton);

        frame.add(scene, BorderLayout.CENTER);
        frame.add(buttons, BorderLayout.NORTH);

        frame.setSize(500, 500);
        frame.setVisible(true);
    }
}
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_2;

import java.awt.geom.Ellipse2D;
import java.awt.geom.Line2D;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;

/**
 *
 * @author Carlos Guisao
 */
public class CarShape extends CompoundShape
{
    /**
        Constructs a car shape.
        @param x the left of the bounding rectangle
        @param y the top of the bounding rectangle
        @param width the width of the bounding rectangle
    */
    public CarShape(int x, int y, int width)
    {
        Rectangle2D.Double body
            = new Rectangle2D.Double(x, y + width / 6,
                width - 1, width / 6);
        Ellipse2D.Double frontTire
            = new Ellipse2D.Double(x + width / 6, y + width / 3,
                width / 6, width / 6);
        Ellipse2D.Double rearTire
            = new Ellipse2D.Double(x + width * 2 / 3,
                y + width / 3,
                width / 6, width / 6);

```

Output | Usages | run.xml | SceneEditor.java | CarShape.java

Source | History

```
37
38          // The bottom of the front windshield
39          Point2D.Double r1
40              = new Point2D.Double(x + width / 6, y + width / 6);
41          // The front of the roof
42          Point2D.Double r2
43              = new Point2D.Double(x + width / 3, y);
44          // The rear of the roof
45          Point2D.Double r3
46              = new Point2D.Double(x + width * 2 / 3, y);
47          // The bottom of the rear windshield
48          Point2D.Double r4
49              = new Point2D.Double(x + width * 5 / 6, y + width / 6);
50          Line2D.Double frontWindshield
51              = new Line2D.Double(r1, r2);
52          Line2D.Double roofTop
53              = new Line2D.Double(r2, r3);
54          Line2D.Double rearWindshield
55              = new Line2D.Double(r3, r4);
56
            add(body);
            add(frontTire);
            add(rearTire);
            add(frontWindshield);
            add(roofTop);
            add(rearWindshield);
63      }
64  }
65
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_2;

import java.awt.BasicStroke;
import java.awt.Color;
import java.awt.Graphics2D;
import java.awt.Shape;
import java.awt.Stroke;
import java.awt.geom.AffineTransform;
import java.awt.geom.GeneralPath;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;

/**
 *
 * @author Carlos Guisao
 */
public abstract class CompoundShape extends SelectableShape
{
    public CompoundShape()
    {
      path = new GeneralPath();
    }

    protected void add(Shape s)
    {
      path.append(s, false);
    }

    @Override
    public boolean contains(Point2D aPoint)
    {
      return path.contains(aPoint);
    }

```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java

Source | History

```java
40      @Override
        public void translate(int dx, int dy)
42      {
43        path.transform(AffineTransform.getTranslateInstance(dx, dy));
44      }
45
46      @Override
        public void draw(Graphics2D g2)
48      {
49        g2.draw(path);
50      }
51
52      /**
53       * Draws a blue dashed border around a selected shape,
54       * and draws blue squares in the corners of the border.
55       * @param g2 The Graphics object used to draw the shapes.
56       * @see #drawSelectedBorder(Graphics2D)
57       * @see #drawSelectedCorners(Graphics2D)
58       */
59      @Override
        public void drawSelection(Graphics2D g2) {
61          drawSelectedBorder(g2);
62          drawSelectedCorners(g2);
63      }
64
65      /**
66       * Draws a blue dashed border around the shape.
67       * @param g2 the Graphics object used to draw the shapes.
68       */
69      private void drawSelectedBorder(Graphics2D g2) {
70          Rectangle2D bounds = path.getBounds();
71          double x = bounds.getX(), y = bounds.getY(), width = bounds.getMaxX()-x, height = bounds.getMaxY()-y;
72          Stroke dashed = new BasicStroke(2, BasicStroke.CAP_BUTT, BasicStroke.JOIN_BEVEL, 0, new float[]{6}, 0);
73          Rectangle2D border = new Rectangle2D.Double(x, y, width, height);
74
75          g2.setStroke(dashed);
76          g2.setColor(Color.BLUE);
77          g2.draw(border);
78          g2.setStroke(new BasicStroke()); // reset so other shapes aren't affected
79          g2.setColor(Color.BLACK);
80      }
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java

Source | History

```java
        g2.setColor(Color.BLACK);
    }

    /**
     * Draws blue squares in the corners of the shape's
     * border drawn by drawSelectedBorder()
     * @param g2 the Graphics object used to draw the shapes.
     * @see #drawSelectedBorder(Graphics2D)
     */
    private void drawSelectedCorners(Graphics2D g2) {
        Rectangle2D bounds = path.getBounds();
        double x = bounds.getX(), y = bounds.getY(), width = bounds.getMaxX()-x, height = bounds.getMaxY()-y;
        Rectangle2D.Double[] corners = {
                new Rectangle2D.Double(x, y, 6, 6),
                new Rectangle2D.Double((x+width)-6, y, 6, 6),
                new Rectangle2D.Double(x, (y+height)-6, 6, 6),
                new Rectangle2D.Double((x+width)-6, (y+height)-6, 6, 6)
        };
        g2.setPaint(Color.BLUE);
        for (Rectangle2D.Double corner : corners) {
            g2.fill(corner);
        }
        g2.setPaint(Color.BLACK); // reset so other shapes aren't affected
    }

    private final GeneralPath path;
}
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java | HouseShape.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_2;

import java.awt.geom.Line2D;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;

/**
 *
 * @author Carlos Guisao
 */
public class HouseShape extends CompoundShape
{
    /**
        Constructs a house shape.
        @param x the left of the bounding rectangle
        @param y the top of the bounding rectangle
        @param width the width of the bounding rectangle
    */
    public HouseShape(int x, int y, int width)
    {
        Rectangle2D.Double base
            = new Rectangle2D.Double(x, y + width, width, width);

        // The left bottom of the roof
        Point2D.Double r1
            = new Point2D.Double(x, y + width);
        // The top of the roof
        Point2D.Double r2
            = new Point2D.Double(x + width / 2, y);
        // The right bottom of the roof
        Point2D.Double r3
            = new Point2D.Double(x + width, y + width);

```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java | HouseSha

Source | History

```
22          @param width the width of the bounding rectangle
23      */
24      public HouseShape(int x, int y, int width)
25      {
26          Rectangle2D.Double base
27              = new Rectangle2D.Double(x, y + width, width, width);
28
29          // The left bottom of the roof
30          Point2D.Double r1
31              = new Point2D.Double(x, y + width);
32          // The top of the roof
33          Point2D.Double r2
34              = new Point2D.Double(x + width / 2, y);
35          // The right bottom of the roof
36          Point2D.Double r3
37              = new Point2D.Double(x + width, y + width);
38
39          Line2D.Double roofLeft
40              = new Line2D.Double(r1, r2);
41          Line2D.Double roofRight
42              = new Line2D.Double(r2, r3);
43
            add(base);
            add(roofLeft);
            add(roofRight);
47      }
48  }
49
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java | HouseShape.java | SceneCom

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_2;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Point;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.MouseMotionAdapter;
import java.util.ArrayList;
import javax.swing.JComponent;

/**
 *
 * @author Carlos Guisao
 */
public class SceneComponent extends JComponent
{
    public SceneComponent()
    {
        shapes = new ArrayList<>();

        addMouseListener(new
            MouseAdapter()
            {
                @Override
                public void mousePressed(MouseEvent event)
                {
                    mousePoint = event.getPoint();
                    shapes.stream().filter((s) -> (s.contains(mousePoint))).forEachOrdered((s) -> {
                        s.setSelected(!s.isSelected());
                    });
                    repaint();
                }
            });
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java | HouseShape.java | Scen

Source | History

```
40
41          addMouseMotionListener(new
42           MouseMotionAdapter()
43           {
44              @Override
45              public void mouseDragged(MouseEvent event)
46              {
47                 Point lastMousePoint = mousePoint;
48                 mousePoint = event.getPoint();
49                 shapes.stream().filter((s) -> (s.isSelected())).forEachOrdered((s) -> {
50                     double dx = mousePoint.getX() - lastMousePoint.getX();
51                     double dy = mousePoint.getY() - lastMousePoint.getY();
52                     s.translate((int) dx, (int) dy);
53                  });
54                 repaint();
55              }
56           });
57       }
58
59       /**
60          Adds an shape to the scene.
61          @param s the shape to add
62       */
63       public void add(SceneShape s)
64       {
65          shapes.add(s);
66          repaint();
67       }
68
69       /**
70          Removes all selected shapes from the scene.
71       */
72       public void removeSelected()
73       {
74          for (int i = shapes.size() - 1; i >= 0; i--)
75          {
76             SceneShape s = shapes.get(i);
77             if (s.isSelected()) shapes.remove(i);
78          }
79          repaint();
80       }
```

```java

    @Override
    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        Graphics2D g2 = (Graphics2D) g;
        shapes.stream().map((s) -> {
            s.draw(g2);
             return s;
         }).filter((s) -> (s.isSelected())).forEachOrdered((s) -> {
             s.drawSelection(g2);
         });
    }

    private ArrayList<SceneShape> shapes;
    private Point mousePoint;
}
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java | HouseShape.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_2;

import java.awt.Graphics2D;
import java.awt.geom.Point2D;

/**
 *
 * @author Carlos Guisao
 */
public interface SceneShape
{
    /**
        Draws this item.
        @param g2 the graphics context
    */
    void draw(Graphics2D g2);
    /**
        Draws the selection adornment of this item.
        @param g2 the graphics context
    */
    void drawSelection(Graphics2D g2);
    /**
        Sets the selection state of this item.
        @param b true if this item is selected
    */
    void setSelected(boolean b);
    /**
        Gets the selection state of this item.
        @return true if this item is selected
    */
    boolean isSelected();
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java | Ho

Source | History

```
23          Draws the selection adornment of this item.
24          @param g2 the graphics context
25      */
        void drawSelection(Graphics2D g2);
27      /**
28          Sets the selection state of this item.
29          @param b true if this item is selected
30      */
        void setSelected(boolean b);
32      /**
33          Gets the selection state of this item.
34          @return true if this item is selected
35      */
        boolean isSelected();
37      /**
38          Translates this item by a given amount.
39          @param dx the amount to translate in x-direction
40          @param dy the amount to translate in y-direction
41      */
        void translate(int dx, int dy);
43      /**
44          Tests whether this item contains a given point.
45          @param p a point
46          @return true if this item contains p
47      */
        boolean contains(Point2D p);
49  }
50
```

Output | Usages | run.xml | SceneEditor.java | CarShape.java | CompoundShape.java | HouseShape.jav

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Chapter_6_2;

import java.awt.Graphics2D;

/**
 *
 * @author Carlos Guisao
 */
public abstract class SelectableShape implements SceneShape
{
    @Override
    public void setSelected(boolean b)
    {
        selected = b;
    }

    @Override
    public boolean isSelected()
    {
        return selected;
    }

    @Override
    public abstract void drawSelection(Graphics2D g2);

    private boolean selected;
}
```