Carshape.java

/\*\*

A car that can be moved around.

**@author** Text Book Sample

\*/

**public** **class** CarShape **implements** CompositeShape

{

/\*\*

Constructs a car item.

**@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)

{

**this**.x = x;

**this**.y = y;

**this**.width = width;

}

**public** **void** translate(**int** dx, **int** dy)

{

x += dx;

y += dy;

}

**public** **void** draw(Graphics2D g2)

{

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);

// The bottom of the front windshield

Point2D.Double r1

= **new** Point2D.Double(x + width / 6, y + width / 6);

// The front of the roof

Point2D.Double r2

= **new** Point2D.Double(x + width / 3, y);

// The rear of the roof

Point2D.Double r3

= **new** Point2D.Double(x + width \* 2 / 3, y);

// The bottom of the rear windshield

Point2D.Double r4

= **new** Point2D.Double(x + width \* 5 / 6, y + width / 6);

Line2D.Double frontWindshield

= **new** Line2D.Double(r1, r2);

Line2D.Double roofTop

= **new** Line2D.Double(r2, r3);

Line2D.Double rearWindshield

= **new** Line2D.Double(r3, r4);

g2.draw(body);

g2.draw(frontTire);

g2.draw(rearTire);

g2.draw(frontWindshield);

g2.draw(roofTop);

g2.draw(rearWindshield);

}

**public** CompositeShape getBounds() {

**return** **this**;

}

**public** **int** getWidth() {

**return** width;

}

**public** **int** getHeight() {

**return** 30;

}

**private** **int** x;

**private** **int** y;

**private** **int** width;

}

compositeShape.java

**import** java.awt.Graphics2D;

**public** **interface** CompositeShape {

**public** **void** draw(Graphics2D g2);

**public** CompositeShape getBounds();

**public** **int** getWidth();

**public** **int** getHeight();

}

CompositeShapeInt.java

/\*\*

A shape that is composed of several individual shapes.

\*/

**public** **interface** CompositeShapeInt

{

/\*\*

Add a shape to this composite shape.

**@param** aShape the shape to add

\*/

**void** add(Shape aShape);

/\*\*

Returns an integer Rectangle that completely encloses the Shape.

Note that there is no guarantee that the returned Rectangle is the

smallest bounding box that encloses the Shape, only that the Shape

lies entirely within the indicated Rectangle.

\*/

Rectangle getBounds();

/\*\*

Draws this CompositeShape at the relative position to the top left corner of the bounding box.

**@param** g2 the graphics context

\*/

**void** draw (Graphics2D g);

}

MyShape

/\*\*

A car that can be moved around.

**@author** Text Book Sample

\*/

**public** **class** MyShape **implements** CompositeShape

{

/\*\*

Constructs a car item.

**@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** MyShape(**int** x, **int** y, **int** width)

{

**this**.x = x;

**this**.y = y;

**this**.width = width;

}

**public** **void** translate(**int** dx, **int** dy)

{

x += dx;

y += dy;

}

**public** **void** draw(Graphics2D g2)

{

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);

g2.draw(body);

g2.draw(frontTire);

g2.draw(rearTire);

}

**public** CompositeShape getBounds() {

**return** **this**;

}

**public** **int** getWidth() {

**return** width;

}

**public** **int** getHeight() {

**return** 30;

}

**private** **int** x;

**private** **int** y;

**private** **int** width;

}

ShapeDisplayer.java

**public** **class** ShapeDisplayer

{

**public** **static** **void** main(String[] args)

{

ShapeFrame frame = **new** ShapeFrame();

frame.addShape(**new** SnowMan(0, 0, 20));

frame.addShape(**new** CarShape(0, 0, 50));

frame.addShape(**new** MyShape(0, 0, 50));

frame.setSize(300, 400);

frame.setTitle("Shape Displayer");

frame.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

frame.setVisible(**true**);

}

}

ShapeFrame.java

**public** **class** ShapeFrame **extends** JFrame{

**private** JPanel btnPanel;

**private** JPanel viewPanel;

**private** **int** mode;

**public** ShapeFrame() {

**super**();

setLayout(**new** BorderLayout());

mode = 1;

btnPanel = **new** JPanel();

viewPanel = **new** JPanel();

btnPanel.setLayout(**new** BoxLayout(btnPanel, BoxLayout.***X\_AXIS***));

viewPanel.setLayout(**new** OverlayLayout(viewPanel));

viewPanel.addMouseListener(**new** MouseAdapter() {

**public** **void** mouseClicked(MouseEvent e) {

**if** (mode == 1) {

CarShape car = **new** CarShape(e.getX(), e.getY(), 50);

ShapeIcon icon = **new** ShapeIcon(car, 300, 400);

JLabel label = **new** JLabel();

label.setIcon(icon);

viewPanel.add(label);

viewPanel.revalidate();

viewPanel.repaint();

System.***out***.println("clicked x is:"+e.getX()+" Y is:"+e.getY());

} **if** (mode == 2) {

SnowMan snow = **new** SnowMan(e.getX(), e.getY(), 50);

ShapeIcon icon = **new** ShapeIcon(snow, 300, 400);

JLabel label = **new** JLabel();

label.setIcon(icon);

viewPanel.add(label);

viewPanel.revalidate();

viewPanel.repaint();

System.***out***.println("clicked x is:"+e.getX()+" Y is:"+e.getY());

} **else** {

MyShape m = **new** MyShape(e.getX(),e.getY(), 50);

ShapeIcon icon = **new** ShapeIcon(m, 300, 400);

JLabel label = **new** JLabel();

label.setIcon(icon);

viewPanel.add(label);

viewPanel.revalidate();

viewPanel.repaint();

System.***out***.println("clicked x is:"+e.getX()+" Y is:"+e.getY());

}

}

});

add(btnPanel, BorderLayout.***NORTH***);

add(viewPanel, BorderLayout.***CENTER***);

}

ShapeIcon.java

/\*\*

An icon that is painted by drawing a Shape

\*/

**public** **class** ShapeIcon **implements** Icon

{ **private** CompositeShape shape;

**private** **int** width;

**private** **int** height;

/\*\*

Construct a ShapeIcon.

**@param** aShape the shape to use when painting

**@param** aWidth the width of the icon

**@param** aHeight the height of the icon

\*/

**public** ShapeIcon(CompositeShape aShape, **int** aWidth, **int** aHeight)

{

shape = aShape;

width = aWidth;

height = aHeight;

}

**public** **int** getIconWidth()

{

**return** width;

}

**public** **int** getIconHeight()

{

**return** height;

}

**public** **void** paintIcon(Component c, Graphics g, **int** x, **int** y)

{

**double** shapeWidth = shape.getBounds().getWidth();

**double** shapeHeight = shape.getBounds().getHeight();

**double** scaleX = Math.*max*(1, shapeWidth / width);

**double** scaleY = Math.*max*(1, shapeHeight / height);

**double** scale = 1 / Math.*max*(scaleX, scaleY);

Graphics2D g2 = (Graphics2D)g;

AffineTransform oldTransform = g2.getTransform();

g2.translate(x, y);

g2.scale(scale, scale);

g2.setColor(Color.***black***);

shape.draw(g2);

g2.setTransform(oldTransform);

}

}

SnowMan.java

/\*\*

An icon that is painted by drawing a Shape

\*/

**public** **class** ShapeIcon **implements** Icon

{ **private** CompositeShape shape;

**private** **int** width;

**private** **int** height;

/\*\*

Construct a ShapeIcon.

**@param** aShape the shape to use when painting

**@param** aWidth the width of the icon

**@param** aHeight the height of the icon

\*/

**public** ShapeIcon(CompositeShape aShape, **int** aWidth, **int** aHeight)

{

shape = aShape;

width = aWidth;

height = aHeight;

}

**public** **int** getIconWidth()

{

**return** width;

}

**public** **int** getIconHeight()

{

**return** height;

}

**public** **void** paintIcon(Component c, Graphics g, **int** x, **int** y)

{

**double** shapeWidth = shape.getBounds().getWidth();

**double** shapeHeight = shape.getBounds().getHeight();

**double** scaleX = Math.*max*(1, shapeWidth / width);

**double** scaleY = Math.*max*(1, shapeHeight / height);

**double** scale = 1 / Math.*max*(scaleX, scaleY);

Graphics2D g2 = (Graphics2D)g;

AffineTransform oldTransform = g2.getTransform();

g2.translate(x, y);

g2.scale(scale, scale);

g2.setColor(Color.***black***);

shape.draw(g2);

g2.setTransform(oldTransform);

}

}