

5

*Drawing Pictures*

## Self-review Questions

**5.1** Use a Web browser to find the documentation for classes: Line2D, Line2D.Double, Graphics2D, BasicStroke and Color.

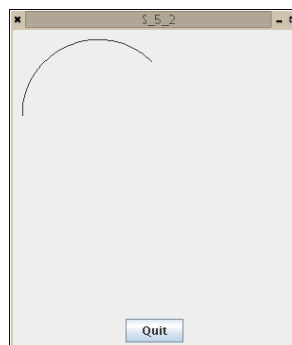
This is more of a 'did you try', type question, so trying to provide an answer is a bit fatuous.

**5.2** Find class Arc2D.Double in the JDK documentation and work out how to draw arcs.

Another 'did you try' question, but this time there is a concrete deliverable. We should write a program to draw an arc. Basing things on the line drawing program on page 148, we came up with:

```
import java.awt.Graphics ;
import java.awt.Graphics2D ;
import java.awt.geom.Arc2D ;
import java.awt.geom.Arc2D.Double ;
public class S_5_2 extends DrawPanel {
    public void paint ( final Graphics g ) {
        final Graphics2D g2d = (Graphics2D) g ;
        final Arc2D aLine = new Arc2D.Double ( 10 , 10 , 160 , 160 , 45 , 135 , Arc2D.OPEN ) ;
        g2d.draw ( aLine ) ;
    }
    public static void main ( final String[] args ) {
        DrawFrame.display ( "S_5_2" , new S_5_2 ( ) ) ;
    }
}
```

When executed we got the frame:



**5.3** What needs to be done to rename class Drawing to class MyPicture?

Edit the source code to change the 'Drawing' to 'MyPicture' wherever it occurs. This can be done quickly by using the Find and Replace command in your editor.

#### 5.4 How can the size of the drawing grid be changed?

Drawing programs are created by copying the **class Drawing** template and editing it. The template defines a **main** method, which looks like this:

```
DrawFrame.display ( "Drawing" , new Drawing ( ) );
```

The expression **new Drawing ( )** creates a new drawing with the default size of 300 by 300 pixels, as the parameter list following **Drawing** is empty. If two size parameter values are added to the parameter list then a drawing can be created with specified size. For example, this:

```
DrawFrame.display ( "Drawing" , new Drawing ( 500 , 400 ) );
```

will create a window with a grid 500 pixels in width and 400 pixels in height. You need to copy the entire **class Drawing** template for this to work correctly.

#### 5.5 Where is the origin (position (0, 0)) of a drawing grid?

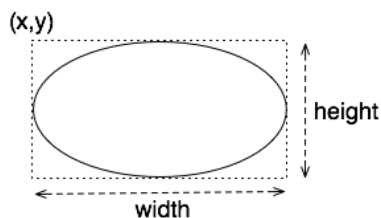
The origin is at the top left corner of a grid or panel. This follows from the way that the Java 2D graphics system defines its coordinate system.

#### 5.6 How is the size and shape of an ellipse specified?

An ellipse is drawn using the **Ellipse2D** class from the Java class libraries. The size and position of an ellipse is specified by giving the rectangular boundary that it should be drawn in, with the width and height of the ellipse determined by the size of the rectangle. For example:

```
Ellipse2D circle = new Ellipse2D.Double ( x , y , width , height ) ;
```

This displays an ellipse with a rectangular boundary of width by height in size, with the top left corner of the boundary at (x,y).



Note that `Ellipse2D` declares two nested subclasses `Ellipse2D.Float` and `Ellipse2D.Double`, which are used to instantiate ellipse objects, as `Ellipse2D` itself is abstract.

Once an ellipse object has been created it is drawn by calling methods like `draw` or `fill` from class `Graphics2D`.

### 5.7 How can you draw a circle?

A circle should be seen as a kind of ellipse and drawn using the `Ellipse2D` class (see previous question). If a square is specified as the boundary, the ellipse will actually be a circle. For example:

```
Ellipse2D circle = new Ellipse2D.Double ( 10 , 10 , 25 , 25 );
```

Here, 10, 10 specifies the top left corner of the square boundary of the circle, with 25, 25 giving the width and height to give the size of the square (both should, of course, be equal).

### 5.8 What is the path of a dodecahedron?

## Programming Exercises

### 5.1 Write a program to draw a hexagon using lines.

```
import java.awt.Graphics ;
import java.awt.Graphics2D ;
import java.awt.geom.Line2D ;
import java.awt.geom.Line2D.Double ;
public class E_5_1 extends DrawPanel {
    public void paint ( final Graphics g ) {
        final Graphics2D g2d = (Graphics2D) g ;
        Line2D aLine = new Line2D.Double ( 100 , 100 , 200 , 100 ) ;
        g2d.draw ( aLine ) ;
        aLine = new Line2D.Double ( 200 , 100 , 250 , 150 ) ;
        g2d.draw ( aLine ) ;
        aLine = new Line2D.Double ( 250 , 150 , 200 , 200 ) ;
        g2d.draw ( aLine ) ;
        aLine = new Line2D.Double ( 200 , 200 , 100 , 200 ) ;
        g2d.draw ( aLine ) ;
        aLine = new Line2D.Double ( 100 , 200 , 50 , 150 ) ;
        g2d.draw ( aLine ) ;
        aLine = new Line2D.Double ( 50 , 150 , 100 , 100 ) ;
        g2d.draw ( aLine ) ;
    }
    public static void main ( final String[] args ) {
        DrawFrame.display ( "E_5_1" , new E_5_1 ( ) ) ;
    }
}
```

5.2 Write a program to draw a hexagon using class *GeneralPath*.

```
import java.awt.Graphics ;
import java.awt.Graphics2D ;
import java.awt.geom.GeneralPath ;
public class E_5_2 extends DrawPanel {
    public void paint ( final Graphics g ) {
        final Graphics2D g2d = (Graphics2D) g ;
        GeneralPath path = new GeneralPath ( ) ;
        path.moveTo ( 100, 100 ) ;
        path.lineTo ( 200, 100 ) ;
        path.lineTo ( 250, 150 ) ;
        path.lineTo ( 200, 200 ) ;
        path.lineTo ( 100, 200 ) ;
        path.lineTo ( 50, 150 ) ;
        path.closePath ( ) ;
        g2d.draw ( path ) ;
    }
    public static void main ( final String[] args ) {
        DrawFrame.display ( "E_5_2", new E_5_2 ( ) ) ;
    }
}
```

This seems a whole lot more sensible than using lines. In particular, we do not have to repeat the coordinates of points that are end points of one line and start points of another line. But then that is the whole point of a path.

5.3 Write a program to draw a collection of hexagons of different sizes and in different positions, using the *GeneralPath* you created in the last question.

The code:

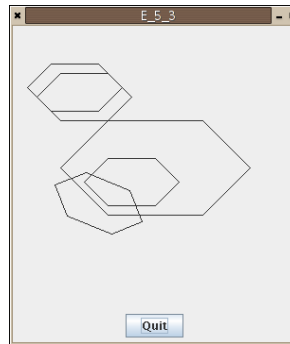
```
import java.awt.Graphics ;
import java.awt.Graphics2D ;
import java.awt.geom.GeneralPath ;
public class E_5_3 extends DrawPanel {
    public void paint ( final Graphics g ) {
        final Graphics2D g2d = (Graphics2D) g ;
        GeneralPath path = new GeneralPath ( ) ;
        path.moveTo ( 100, 100 ) ;
        path.lineTo ( 200, 100 ) ;
        path.lineTo ( 250, 150 ) ;
        path.lineTo ( 200, 200 ) ;
        path.lineTo ( 100, 200 ) ;
        path.lineTo ( 50, 150 ) ;
        path.closePath ( ) ;
        g2d.draw ( path ) ;
        g2d.scale ( 0.5, 0.5 ) ;
        g2d.draw ( path ) ;
        g2d.translate ( -20, -20 ) ;
        g2d.draw ( path ) ;
        g2d.translate ( 120, 200 ) ;
```

```

g2d.draw ( path ) ;
g2d.rotate ( Math.PI / 8 ) ;
g2d.draw ( path ) ;
}
public static void main ( final String[] args ) {
    DrawFrame.display ( "E_5_3" , new E_5_3 ( ) ) ;
}
}

```

creates the drawing:



#### 5.4 Write a program to draw a picture of a house.

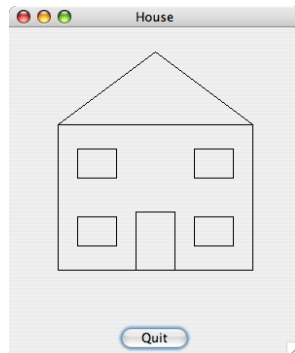
A basic house can be drawn using mostly rectangles with two lines to form the roof. This is primarily an exercise in determining the correct coordinates for drawing everything.

```

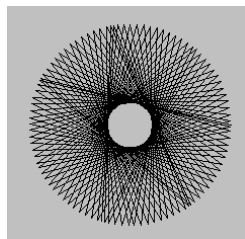
import java.awt.Graphics ;
import java.awt.Graphics2D ;
import java.awt.geom.Line2D ;
import java.awt.geom.Rectangle2D ;
public class E_5_4 extends DrawPanel {
    public void paint ( final Graphics g ) {
        final Graphics2D g2d = (Graphics2D) g ;
        g2d.draw ( new Line2D.Double ( 50 , 100 , 150 , 25 ) ) ;
        g2d.draw ( new Line2D.Double ( 150 , 25 , 250 , 100 ) ) ;
        g2d.draw ( new Rectangle2D.Double ( 130 , 190 , 40 , 60 ) ) ;
        g2d.draw ( new Rectangle2D.Double ( 50 , 100 , 200 , 150 ) ) ;
        g2d.draw ( new Rectangle2D.Double ( 70 , 195 , 40 , 30 ) ) ;
        g2d.draw ( new Rectangle2D.Double ( 190 , 195 , 40 , 30 ) ) ;
        g2d.draw ( new Rectangle2D.Double ( 70 , 125 , 40 , 30 ) ) ;
        g2d.draw ( new Rectangle2D.Double ( 190 , 125 , 40 , 30 ) ) ;
    }
    public static void main ( final String[] args ) {
        DrawFrame.display ( "House" , new E_5_4 ( ) ) ;
    }
}

```

The house looks like this:



- 5.5 Write a program that uses `drawString` to display your name and a message, using a large font.
- 5.6 Write a program to draw a sine wave.
- 5.7 As suggested in Section 5.2.4, page 147, write a method called `createTriangle` that can be used to create triangles of any size at any position. Use the method in a program to demonstrate that triangle drawing works correctly.
- 5.8 Repeat the last question but additionally allow the fill colour and outline width of triangles to be specified as well. (This will require some thinking about what actually goes in the `createTriangle` method.)
- 5.9 Rewrite program Chessboard so that it can display a board of any size in any position, as suggested in the design review section.
- 5.10 Write a program to display a shape like this one:



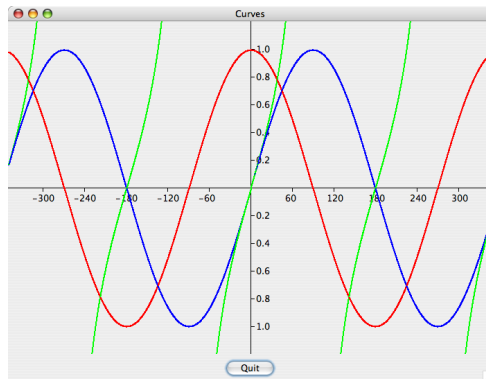
- 5.11 Write a program that displays a graph showing the curves  $y = \sin x$ ,  $y = \cos x$  and  $y = \tan x$ .

```

import java.awt.Color ;
import java.awt.Graphics ;
import java.awt.Graphics2D ;
import java.awt.geom.Line2D ;
import java.awt.geom.Rectangle2D ;
public class E_5_11 extends DrawPanel {
    public E_5_11 ( final int w , final int h ) { super ( w , h ) ; }
    public void paint ( final Graphics g ) {
        final Graphics2D g2d = ( Graphics2D ) g ;
        final int width = getWidth ( ) ;
        final int height = getHeight ( ) ;
        final int x_zero = width / 2 ;
        final int y_zero = height / 2 ;
        final double y_max = 1.2 ;
        // Draw and label the x axis.
        g2d.draw ( new Line2D.Double ( 0 , y_zero , width , y_zero ) ) ;
        for ( int x = 60 ; x < x_zero ; x += 60 ) {
            g2d.draw ( new Line2D.Double ( x_zero + x , y_zero , x_zero + x , y_zero + 5 ) ) ;
            g2d.draw ( new Line2D.Double ( x_zero - x , y_zero , x_zero - x , y_zero + 5 ) ) ;
            g2d.drawString( "" + x , x_zero + x - 10 , y_zero + 20 ) ;
            g2d.drawString( "-" + x , x_zero - x - 15 , y_zero + 20 ) ;
        }
        // Draw the and label the y axis.
        g.drawLine ( x_zero , 0 , x_zero , height ) ;
        for ( double y = 0.2 ; y < y_max ; y += 0.2 ) {
            final int y_pixel = (int) ( y * ( y_zero / y_max ) ) ;
            g2d.draw ( new Line2D.Double ( x_zero , y_pixel , x_zero + 5 , y_pixel ) ) ;
            g2d.draw ( new Line2D.Double ( x_zero , y_pixel + y_zero , x_zero + 5 , y_pixel + y_zero ) ) ;
            String label = ( "" + y ).substring ( 0 , 3 ) ;
            g.drawString( label , x_zero + 8 , y_zero - y_pixel + 5 ) ;
            g.drawString( label , x_zero + 8 , y_zero + y_pixel + 5 ) ;
        }
        // Draw the curves.
        for ( double x = -x_zero ; x < x_zero ; x += 0.2 ) {
            g2d.setColor ( Color.red ) ;
            double y = Math.cos ( Math.toRadians ( x ) ) ;
            int y_pixel = y_zero - (int) ( y * ( y_zero / y_max ) ) ;
            g2d.draw ( new Rectangle2D.Double ( (int) x + x_zero , y_pixel , 1 , 1 ) ) ;
            g2d.setColor ( Color.blue ) ;
            y = Math.sin ( Math.toRadians ( x ) ) ;
            y_pixel = y_zero - (int) ( y * ( y_zero / y_max ) ) ;
            g2d.draw ( new Rectangle2D.Double ( (int) x + x_zero , y_pixel , 1 , 1 ) ) ;
            g2d.setColor ( Color.green ) ;
            y = Math.tan ( Math.toRadians ( x ) ) ;
            y_pixel = y_zero - (int) ( y * ( y_zero / y_max ) ) ;
            g2d.draw ( new Rectangle2D.Double ( (int) x + x_zero , y_pixel , 1 , 1 ) ) ;
        }
    }
    public static void main ( final String[] args ) {
        DrawFrame.display ( "Curves" , new E_5_11 ( 700 , 480 ) ) ;
    }
}

```

This program displays the following:



Most of the detail in this program is about getting the visual presentation neat and tidy, so that the displayed graph fills the window whatever size the window is set to. The size specified in the main method to 700 by 400, but this can be changed to any sensible size.

**5.12** *Extend the program Graph allowing the graph to be placed in any position and to be of any size, with multiple lines given data from multiple files.*

## Challenges

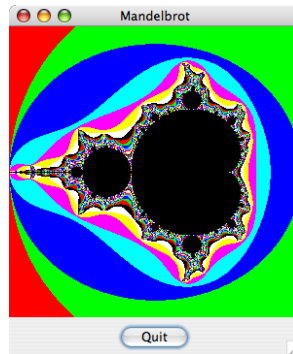
**5.1** *Write a program to draw a random collection of shapes, with different fills and outline widths.*

*Hint: The JDK has a class that generates random numbers, which can be employed to determine the properties of the shapes drawn.*

**5.2** *Write a program that draws a series of curves using the classes `CubicCurve2D.Double` and `QuadCurve2D.Double`.*

**5.3** *Draw a picture of a Mandelbrot set.*

The Mandelbrot set is a kind of fractal that has an appealing visual representation. The drawing program listed below displays this basic Mandelbrot:



```

import java.awt.Color ;
import java.awt.Graphics ;
import java.awt.Graphics2D ;
import java.awt.geom.Line2D ;
import java.awt.geom.Line2D.Double ;

public class C_5_3 extends DrawPanel {
    public C_5_3 ( ) { }
    public C_5_3 ( final int w , final int h ) { super ( w , h ) ; }
    @Override
    public void paint ( final Graphics g ) {
        final Graphics2D g2d = ( Graphics2D ) g ;
        int numberOfColours = 8 ;
        Color colours[] = new Color[] {
            Color.black , Color.red , Color.green , Color.blue ,
            Color.cyan , Color.magenta , Color.yellow , Color.white
        } ;
        int columns = getWidth ( ) ;
        int rows = getHeight ( ) ;
        int iterations = 256 ;
        int size = 4 ;

        for ( int column = 0 ; column < columns ; column++ ) {
            for ( int row = 0 ; row < rows ; row++ ) {
                double x = 0.0 ;
                double y = 0.0 ;
                int colour ;
                for ( colour = 0 ; colour < iterations ; colour++ ) {
                    if ( ( x * x + y * y ) > size ) { break ; }
                    double newy = 2 * x * y + ( ( row * 0.01 ) - 1.5 ) ;
                    x = x * x - y * y + ( ( column * 0.01 ) - 2.0 ) ;
                    y = newy ;
                }
                g2d.setColor ( colours [ colour % numberOfColours ] ) ;
                final Line2D point = new Line2D.Double ( column , row , column , row ) ;
                g2d.draw ( point ) ;
            }
        }
    }
    public static void main ( final String[] args ) {
        DrawFrame.display ( "Mandelbrot" , new C_5_3 ( ) ) ;
    }
}

```

```
}  
}
```

See [http://en.wikipedia.org/wiki/Mandelbrot\\_set](http://en.wikipedia.org/wiki/Mandelbrot_set) or <http://mathworld.wolfram.com/MandelbrotSet.html> for more information about the Mandelbrot set.

**5.4** Write a program to create an array of squares where every square is a different colour.