Kyle McClintick, David Caggiano, Thomas D'Agostino

Dr. Delvin Defoe

CSSE 221-01

Sources:

http://docs.oracle.com/javase/6/docs/api/java/awt/Shape.html

Big Java

Shape Classes

The Shape interface is used to make multiple graphics at once by using a different class for each shape that sets their size and position. The interface is good for drawing simple graphics that can be composed of shapes such as rectangles, squares, circles, ovals, lines, and parabolas. A class implementing the Shape interface has its position determined on an x-y coordinate plane with the top left corner of the terminal representing (0,0). Some frequently used classes are the geom super class, the Graphics and Graphics2D classes, which draw basic shapes like rectangles and circles.

Each class implementing the Shape interface provides callbacks to get the class' geometry, determine whether points lie partly or entirely within the interior of the Shape, and retrieve a PathIterator object that describes the trajectory path of the Shape outline.

A Shape's geometry is determined by the interface drawing rectangles around the Shape attempting to get the rectangle's boundrys to match that of the Shape with varying degrees of precision depending on the method called.

Shape's determine if a point is in the Shape in order to improve drawing and locating the Shape. A point must fit one of the following criteria in order to be considered inside the Shape:

- it lies completely inside the Shape boundary *or*
- it lies exactly on the Shape boundary *and* the space immediately adjacent to the point in the increasing X direction is entirely inside the boundary *or*
- it lies exactly on a horizontal boundary segment **and** the space immediately adjacent to the point in the increasing Y direction is inside the boundary.

Shape classes provide information and access to the class' geometry to the user by returning a PathIterator.

Several classes implement the Shape interface at a given time. Some of the important methods that Shape uses are:

- contains: Returns a boolean if a pair of x and y coordinates, point, or rectangle is inside the shape
- getBounds: Retuns an integer Rectangle that completely encloses the shape

- intersects: Tests if the interior of the Shape intersects the interior of a specified rectangular area

It is important to note that Shapes can overlap each other.

To determine the bounds of Shape, Shape draws a Rectangle that encloses the Shape with different methods to vary precision.

To begin drawing, you must create a class named, for instance, AirPlaneComponent, that extends JComponent and passes in a Graphics object in a void method. Then use the Graphics object to create a Graphics2D object like so: Graphics2D g2 = (Graphics2D) g. Then use the Graphics2D object to draw your graphic. For example, if you created an airplane with an Airplane Class that passed in g2 as a parameter, you'd draw it by typing airplane.draw(g2). Within the Airplane Class, you'd draw each shape of the Airplane. Using perhaps an Ellipse for the body and triangles for the wings; each shape class has parameters for size and position. After creating all of the shape objects within the Airplane Class, you'd draw them all by typing g2.draw(rWing), g2.draw(lWing), g2.draw(body) etc. Finally, you'd create a JFrame in the main method to create an AirPlaneComponent object.

Both the airplane and the components need coordinates.