The Shape interface is an interface that is used for objects which are some sort of geometric shape. The methods are all methods that are useful when dealing with shapes, naturally.

The following methods are a part of any class that implements the Shape interface:

- contains - tests if a particular point or rectangle lies within the shape.
- getBounds and getBounds2D - return a rectangle that encloses the shape entirely
- getPathIterator - provides a way for a shape to describe its boundary geometry through an iterator one segment at a time (using line segments or curves)
- intersects - determines if a give rectangular area intersects with the shape.

Insideness – whether or not a point lies within a shape.

A point is considered to be within a shape if one of the following is true:

- The point lies entirely within the bounds of the shape
- The point lies on a boundary of the shape and the adjacent area in the increasing x direction also lies within the shape.
- The point lies on a horizontal boundary of the shape, as does the adjacent area in the increasing y direction

Some classes which implement the Shape interface:

- Arc2D *
- CubicCurve2D *
- Ellipse2D *
- Line2D *
- Path2D *
- QuadCurve2D *
- Rectangle2D *
- RoundRectangle2D *
- Area
- Polygon
- Rectangle
- RectangularShape

Examples of creating shapes:

- Rectangle(10, 30, 25, 45) – a rectangle with height 45 and width 25 at coordinates (10, 30)
- int[] x = new int[] {1,4,2}, y = new int[] {1,2,3};
  Polygon(x, y, 3) – a polygon with three vertices at (1, 1), (4, 2), and (2, 3)