Implementing a Basic Rectangle Class

Learning Objectives

1. Implementing a Class that Describes an Object,
2. Implementing Accessors, Mutators and Constructors, and
3. Writing a Program to Test the Implementation of a Class

DEFINITION 1. A **Rectangle** is a quadrilateral (a four-sided polygon) whose interior angles are 90° each.

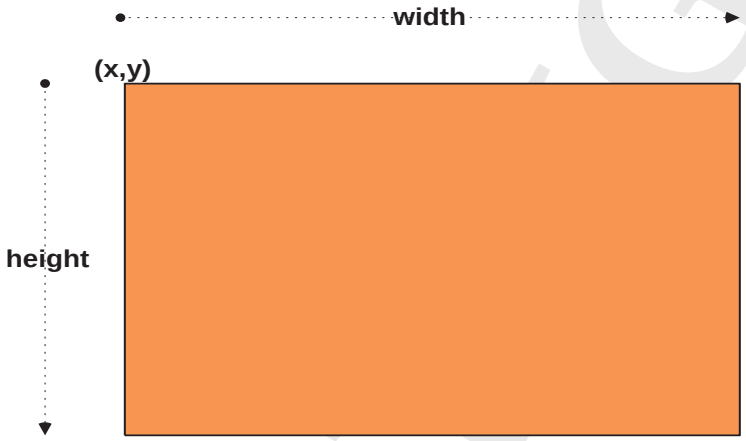

Figure 1: Representation of a Rectangle

The perimeter and area of a rectangle are:

$$perimeter = 2 \times (width + height) \quad \text{(eqn 1)}$$

$$area = width \times height \quad \text{(eqn 2)}$$

In the lab exercise this week, you will specify the rectangle in screen coordinates rather than Cartesian coordinates. The coordinates of the top-left corner of the screen is (0,0). The y axis increases with depth; that is, as we move downward on the screen, the y-coordinate increases and as we move upward on the screen it decreases. The x axis increases rightward; that is, as we move rightward on the screen, the x-coordinate increases and as we move leftward on the screen it decreases. In this project, we will specify a rectangle by the x-coordinate (x) and y-coordinate (y) of the top-left corner of the screen (location) and the *width* and *height* of the rectangle.

The Rectangle2D Class

Implement the class *Rectangle2D* that models the *behavior* of a rectangle in a plane. See the API documentation for the class on Moodle. The class will consist of four instance variables, *x*, *y*, *width* and *height*, all of which are doubles. Be sure to document the class, fields and methods using the standard javadoc style documentation.

The Rectang2DTester Class

Write a class called *Rectangle2DTester* to test the implementation of your *Rectangle2D* class. Using the Scanner class, your program will obtain input values for the location of a rectangle, its width and height. This class should contain no local variables other than the two *Rectangle2D* references, one for the big rectangle and the other for the small one, one for the Point2D.Double location and a reference to a *Scanner* object used to input data from the keyboard. Your program then does the following:

1. Calls the constructor that has an arity of three to create a rectangle using these values. **These values should not be stored to local variables in the main - they should be scanned and used directly as arguments to the constructor**.
2. Calls the constructor that has an arity of three to create a second, bigger, rectangle such that the both rectangles are concentric (have the same center) and there is a path of uniform width, 15 pixels, between the bigger and smaller rectangles as shown in figure 2.

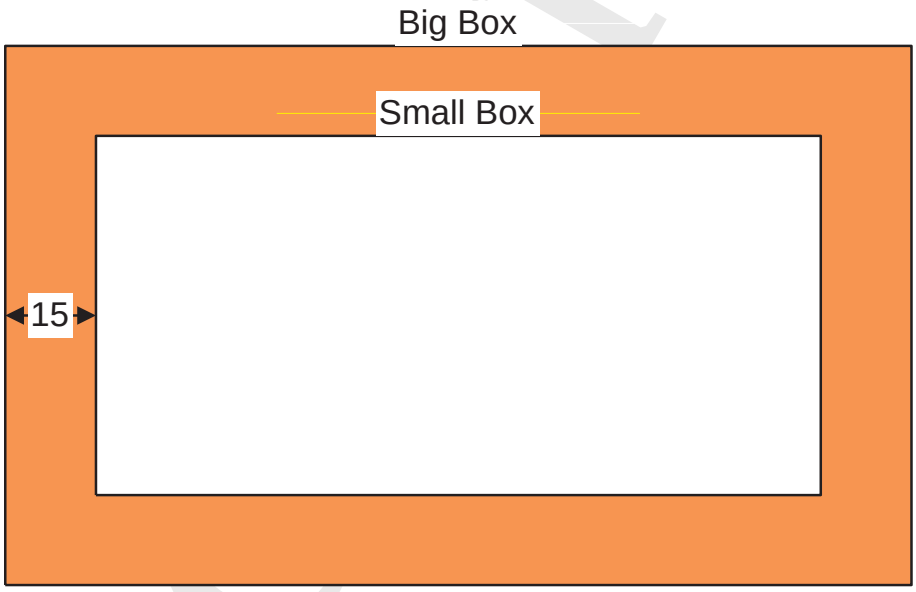

Figure 2: Two Concentric Rectangles

3. Your program then displays the following details, invoking appropriate methods from the *Rectangle2D* class, where the question mark denotes a numeric value to the nearest hundredths:

   (a) Big rectangle: [x = ?; y = ?; w = ?; h = ?]

   (b) Small rectangle: [x = ?; y = ?; w = ?; h = ?]

   (c) The perimeter of the big rectangle is ?.

   (d) The perimeter of the small rectangle is ?.

   (e) The area of the shaded region between the rectangles is ?.

4. Calling the appropriate method, it moves the small rectangle 15 pixels to the left and 20 pixels upward.

5. Your program again displays the following details, invoking appropriate methods from the *Rectangle2D* class, where the question mark denotes a numeric value to the nearest hundredths:

   (a) Big rectangle: [x = ?; y = ?; w = ?; h = ?]

   (b) Small rectangle: [x = ?; y = ?; w = ?; h = ?]

   (c) The perimeter of the big rectangle is ?.

   (d) The perimeter of the small rectangle is ?.

   (e) The area of the big rectangle is ?.

   (f) The area of the small rectangle is ?.

6. Calling the appropriate method from the *Rectangle2D* class, it doubles the width and halves the height of the bigger rectangle while preserving its location.

7. Your program again displays the following details, invoking appropriate methods from the *Rectangle* class, where the question mark denotes a numeric value to the nearest hundredths:

   (a) Big rectangle: [x = ?; y = ?; w = ?; h = ?]

   (b) Small rectangle: [x = ?; y = ?; w = ?; h = ?]

   (c) The perimeter of the big rectangle is ?.

   (d) The perimeter of the small rectangle is ?.

   (e) The area of the big rectangle is ?.

   (f) The area of the small rectangle is ?.

   (g) Big rectangle contains (32.0,32.0). ?

   (h) Small rectangle contains (32.0,32.0). ?

Other Requirements

Remove all Netbeans auto-generated comments. Your program's interactivity must be as shown in the sample runs. Include the Javadoc documentation for each method in *Rectangle2D*, as given in the API documentation handout. Write header comments for both *Rectangle2D* and *Rectangle2DTester* using this Javadoc documentation template:

```
/**
 * Explain the purpose of this class; what it does <br>
 * CSC 1350 Lab # 9
 * @author YOUR NAME
 * @since DATE THE CLASS WAS WRITTEN
 * @see Rectangle2D (ADD THIS TAG ONLY IN THE BankAccountTester Javadoc HEADER COMMENTS)
 */
```

Here are sample program interactions:

Listing 1: Sample Run

```
Enter the x- and y-coordinates of the top-left corner -> 40 50
Enter the width and height of the rectangle -> 80 100

After creating the two rectangles:
Big rectangle: [x = 25.00; y = 35.00; w = 110.00; h = 130.00]
Small rectangle: [x = 40.00; y = 50.00; w = 80.00; h = 100.00]
The perimeter of the big rectangle is 480.00.
The perimeter of the small rectangle is 360.00.
The area of the shaded region is 6300.00.

After moving the small rectangle:
Big rectangle: [x = 25.00; y = 35.00; w = 110.00; h = 130.00]
Small rectangle: [x = 25.00; y = 30.00; w = 80.00; h = 100.00]
The perimeter of the big rectangle is 480.00.
The perimeter of the small rectangle is 360.00.
The area of the big rectangle is 14300.00.
The area of the small rectangle is 8000.00.

After resizing the big rectangle:
Big rectangle: [x = 25.00; y = 35.00; w = 220.00; h = 65.00]
Small rectangle: [x = 25.00; y = 30.00; w = 80.00; h = 100.00]
The perimeter of the big rectangle is 570.00.
The perimeter of the small rectangle is 360.00.
The area of the big rectangle is 14300.00.
The area of the small rectangle is 8000.00.
Big rectangle contains (32.0,32.0). false
Small rectangle contains (32.0,32.0). true
```

Submitting Your Work for Grading

Navigate your way to the ...\NetBeansProjects\Rectangle2DTester\src\rectangle2dtester folder using Windows file explorer. You should find *Rectangle2D.java* and *Rectangle2DTester.java*, files containing your source code for the program. Click one of the files and then hold down the shift key and click the other so that both files are selected. Right-click the selected files and create a compressed (zipped) folder containing a copy of each file. Rename the zip file *PAWSID_lab09.zip*, where *PAWSID* is the prefix of your LSU/Tiger email address - the characters left of the @ sign. Double-click the zip file to verify that both *Rectangle2D.java* and *Rectangle2DTester.java* are included the zip file. If the zip file does not contain both files, delete the zip file and repeat the steps. Upload the zip file to the digital drop box on Moodle.