Keshav N Gainda 216891319 EECS3311A   Project Name : Display Shapes TA : Kazi Mridul

PART - I

In this software project, we are to develop an application which displays an interface with two buttons : one to load six shapes which can be either of circles, rectangles or squares; and a second button to sort them on the basis of their surfaces.

The challenges associated with this project include instantiating the six shapes on the interface of the application - which is initially blank with two buttons, and then sorting them on the basis of their surfaces, and one of the challenges is also to learn new java elements including JFrame, JPanesl, Color, AWT, etc.

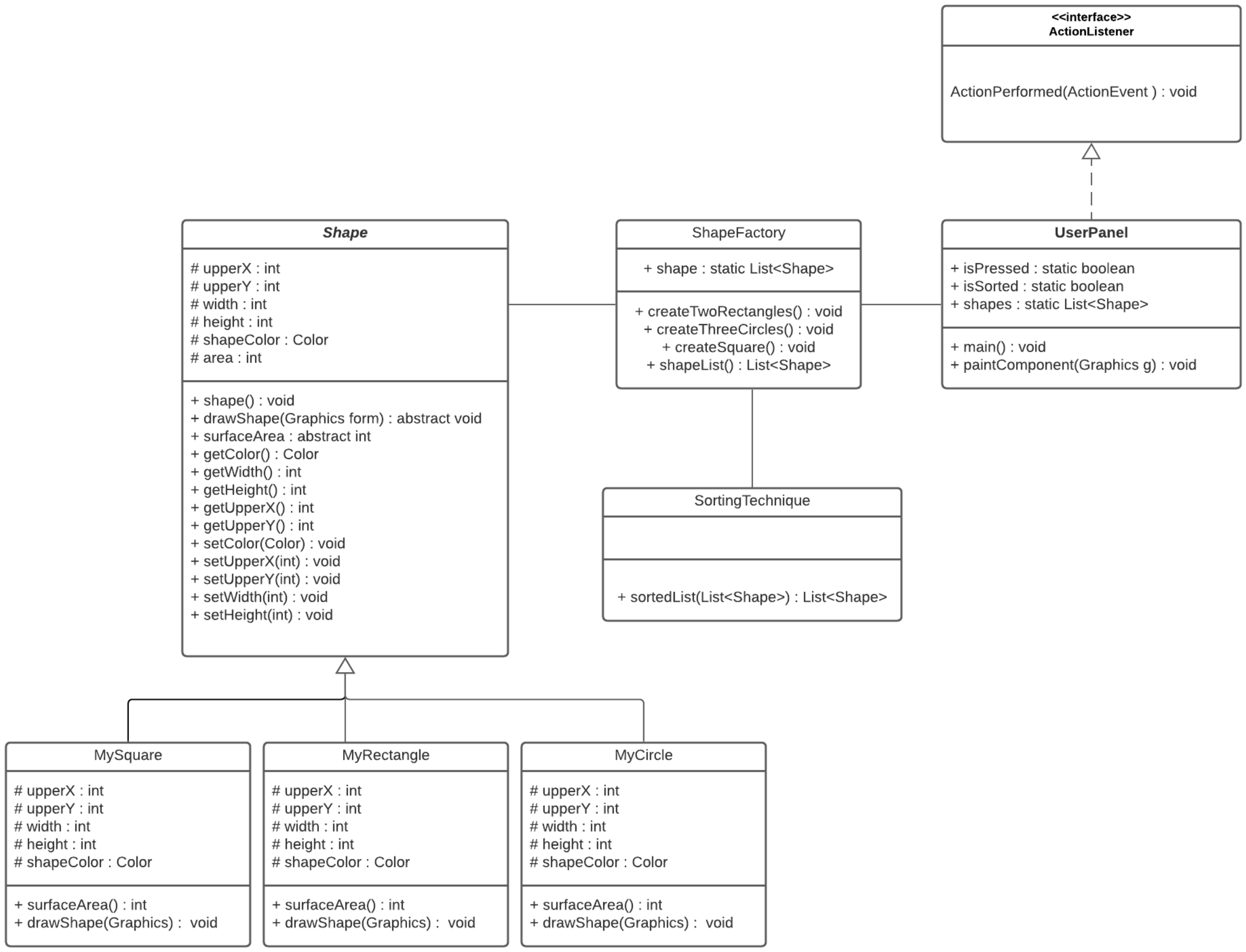
In this project, the concept of Object Oriented Design(OOD) will be implemented. OOD is a programming paradigm, which is used by high-level programming languages such as Java, C++, etc. The main OOD principles that will be used in this application are Abstraction - hiding the internal details of one class from other classes; Encapsulation - keeping the objects and states of a method encapsulated, i.e. inside the class; Inheritance - making a child class by reusing the methods of the parent class without changing them.

This project will follow a combination of the Singleton and the Factory design patterns mostly throughout the code.

The report consists of a brief introduction of the project, its goals and challenges, its design, followed by the design of the solution through a UML class diagram, followed by a description of the implementation of the solution and will end with a conclusion.
It is divided in four parts - Introduction, Class Diagrams, Implementation Description and Conclusion.

Keshav N Gainda 216891319 EECS3311A Project Name : Display Shapes TA : Kazi Mridul

PART - II



PART - III

In my SortingTechnique class, I have used the algorithm of Bubble Sort to sort the shapes on the basis of their surfaces. I have declared a boolean variable named 'sorted' and initialized it to false for the list is unsorted in the beginning. An unsorted list has been added as a parameter of the sorting method which has all the shapes in the random form in which they have been loaded. A while loop has been created which runs until the entire list of shapes has been sorted in ascending order on the basis of their areas.
Using the technique of bubble sort, each shape's area is being compared to the area of the next shape in the list, and the upper left or the origin coordinates as well as the position of the shape in the list are swapped if the area of the current shape is greater than the area of the shape immediately next to the current shape. This continues until the entire list of all the shapes have been sorted.

I have implemented the first class diagram for the solution of this problem. In this solution, I have made seven classes namely, UserPanel, ShapeFactory, Shape, MyRectangle, MyCircle, MySquare, and SortingTechnique. The UserPanel class is extending the JPanel class and is implementing the ActionListener interface in Java, and is being used to create the JFrame and JPanel objects to create the interface for loading and sorting the shapes and buttons on the screen. This class has the main() method, which is responsible for instantiating the entire application - to display the interface containing the buttons and the space for loading the shapes. Another method in this class is the paintComponent() method, which is using the repaint() method from the java AWT package. This method uses an object of the ShapeFactory class to instantiate six shapes and then using the repaint() and drawshape() methods, draws the desired shapes in the desired order when necessary i.e. either when load or sort buttons are pressed.

The second class is the Shape class, which is an abstract class defining the generic attributes of any shape including its height, width, colour, and the upper left coordinates where the shape is to be formed on the screen. It also consists of the generic methods for drawing all the shapes as well as getting their surface areas for sorting them later. It also consists of the accessor and mutator functions i.e. getters and setters for all those attributes for which they are necessary.

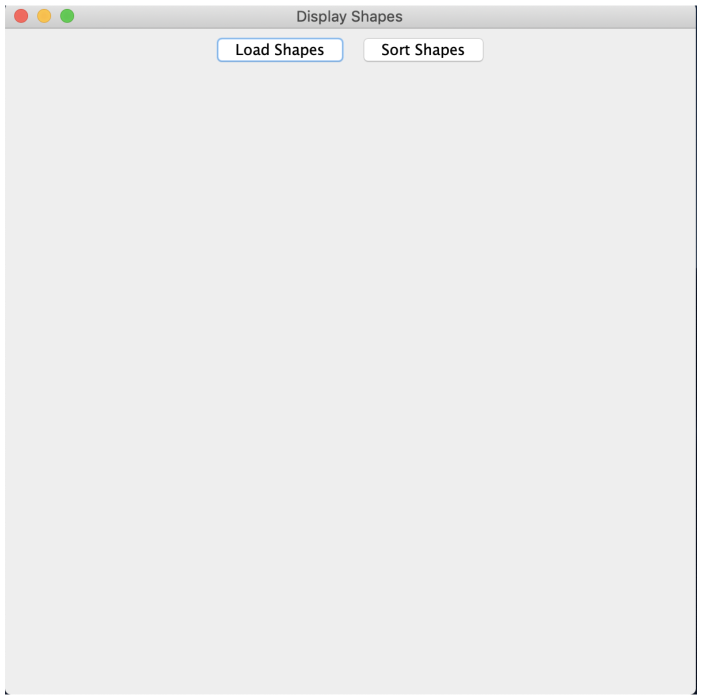
The third class is the ShapeFactory class, which instantiates the six shapes that are to be drawn when the load button is pressed. Also, in this class, after the instantiation of each shape, the surfaceArea() method from the Shape class is called to compute the surface area of each shape for sorting them later. In this class, a list of Shape type is being created which is storing all the shapes that are being instantiated - for this list will be used for sorting the shapes. The shapes in this class are instantiated from objects of different classes named MyRectangle - for rectangles, MyCircle - for circles and MySquare - for square.

The next three classes named MyRectangle, MyCircle and MySquare are all child classes for the parent class Shape. These classes get all their generic properties from the parent class, with the differences in the drawShape() and the surfaceArea() methods for each of them. Since, each shape has a different surface or surface area, the surfaceArea() method in each class is overridden depending on the shape. Also, since all the shapes have a unique way to be drawn, the drawShape() method has been overridden in each class depending on the shape.

The last class is the SortingTechnique class. I have created a method called sortedList() which takes a list of Shape type, which consists of all the shapes that have been created when the load button is pressed in a random order. The method uses the technique of Bubble Sort, in which the surface area of each shape in the list is compared to the surface area of the shape next to the current shape. This all is added in a while loop which runs till the entire list is sorted in ascending order on the basis of surface areas of the shapes in the list. The method returns the list in which all the shapes are sorted on the basis of their surface.

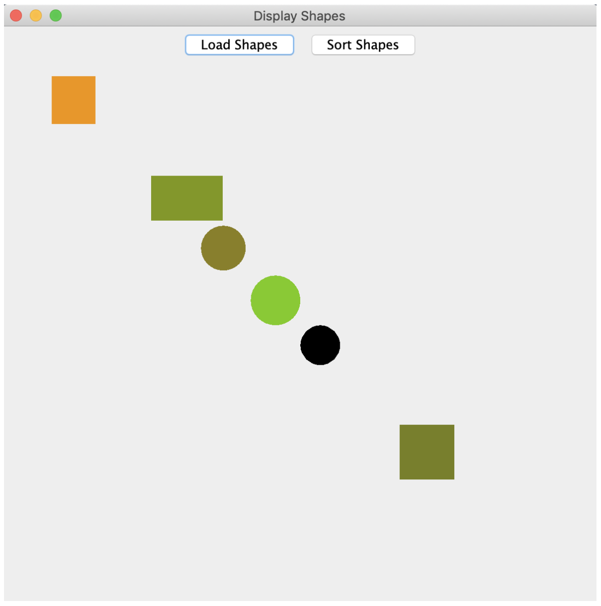
I have used the Eclipse IDE for the development of this software project with the 2019-12 (4.14.0) version and the java version used is Java17 or JDK17.

Below are the snapshots of the interface when different actions occur as described below each image :

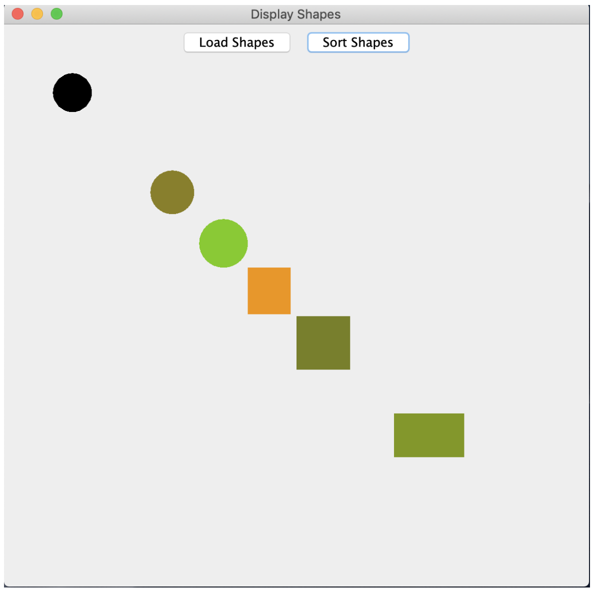


(Image 1) This image is a snapshot of the time instance when the project is initially started, i.e. the blank interface with two butto

Keshav N Gainda 216891319 EECS3311A Project Name : Display Shapes TA : Kazi Mridul



(Image 2) This image is of the time when the Load Shapes button is pressed, and as can be seen, 6 shapes are loaded on the interface.



(Image 3) This image is of the time when the Sort Shapes button is pressed, and as can be seen, the shapes are sorted on the basis of their surfaces, with the smallest one on the top and increasing surfaces as going down.

PART IV

This software project was a challenge that I tried to overcome by giving my best efforts. This project was really interesting as it provided a basis and an opportunity to learn about Object Oriented Design and its patterns and principles.

The sorting problem was a challenge according to me, and the part where I initially went wrong was not swapping the coordinates of the shapes while sorting them, because of which no change was observed when the Sort Shapes button was pressed.

The project was a great learning practice as well as a great practice of effective and efficient software development. This project provided a greater understanding of the various patterns and principles of Object Oriented Design. Also, this project provided me with a new experience of learning about some new java components such as the AWT and Color component of Java.

My Top three recommendations for the ease of completion of this project would be as follows :

1. A bit more knowledge about the components of Java that were used including the API of AWT.
2. Deeper knowledge of UML class diagrams.
3. More practice of OOD principles in application in in-class topics and discussions.