FIRST SOFTWARE PROJECT- EECS 3311

Name: Sagarkumar Patel

Student ID : 217637604

Course : EECS 3311, Section B

INTRODUCTION:

This Software project is about creating an application which sorts different shapes according to its Surface area. It has 2 buttons with different functionalities. First button loads the shapes when clicked and other sorts the shapes. Everytime user clicks load button, the images of shapes loads up on the interface.

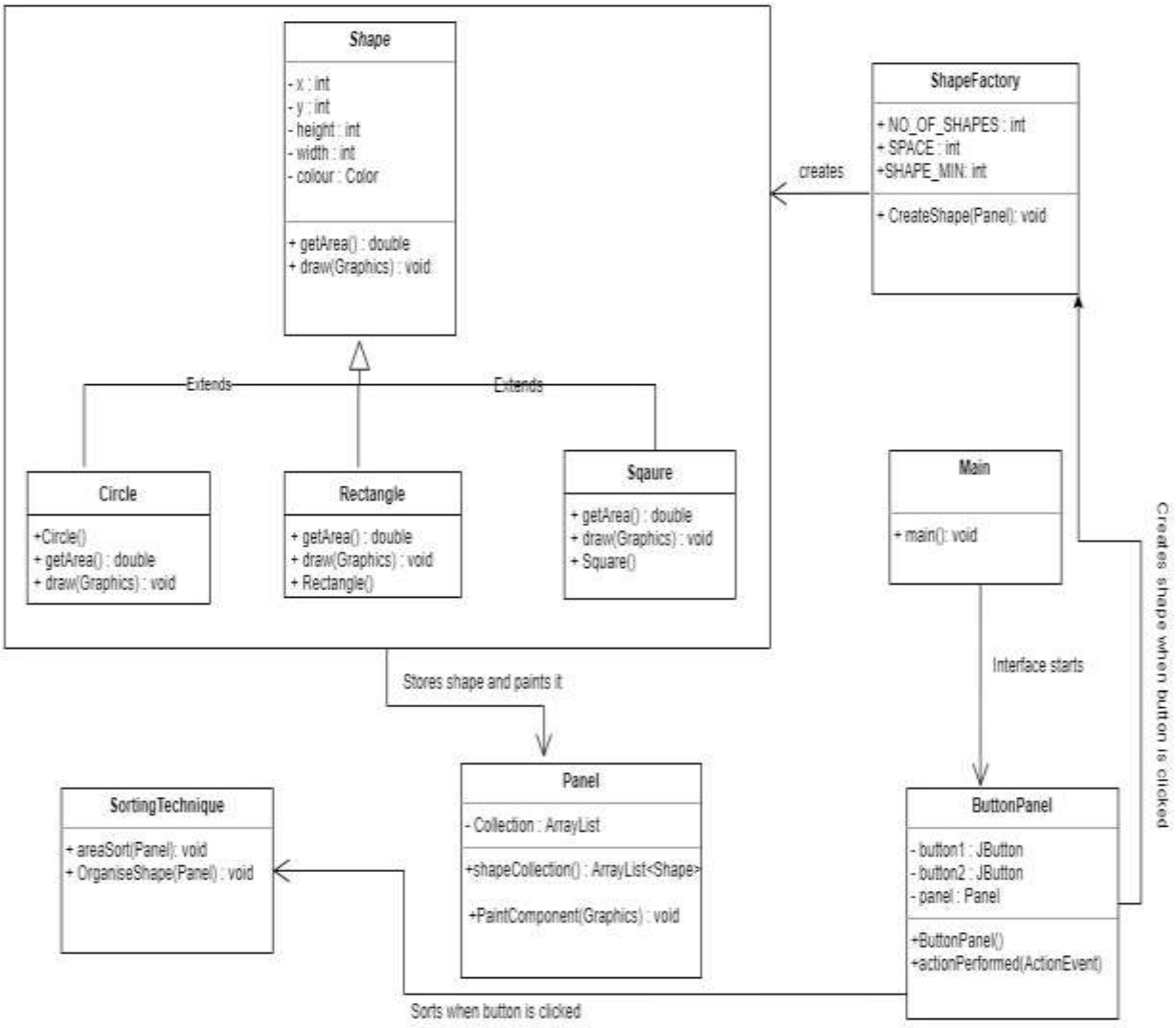
The challenges associated with this software projects are mainly implementing various classes with OO principle of design (OOD). Also working hands on OOPs concept of java, Implementing UML etc… Firstly creating a parent class which inherits all the shapes.Also making shapefactory class which creates all the shapes. The biggest challenge was creating frame and spacing between shapes when loaded.

This Software project uses several concepts like the main one is OOD(Object Oriented Design). Its responsible for shape,coloring and he whole design of an interface. Also getting familiar to swing framework and using its properties for frame,panel and coloring. Moreover bubblesort concept is also used in my first software project.

Design pattern used for my project is Factory pattern. There is a class named ShapeFactory which creates various shapes. Hence abstract factory pattern uses shapeFactory class with method called Createshape(), also it maintains the spacing between shapes when loaded. Hence it's a Use case for my software project which associates ButtonPanel as an actor.

My report is going to be structured as Introduction, followed by a UML which I have implemented to run my application. Defining the class relationships and making class diagrams by specifying details of each class. Also attaching some snapshots and workflow.

DESIGN :


Shape
- x : int
- y : int
- height : int
- width : int
- colour : Color
+ getArea() : double
+ draw(Graphics) : void
ShapeFactory
+ NO_OF_SHAPES : int
+ SPACE : int
+SHAPE_MIN: int
+ CreateShape(Panel): void
creates
Extends
Extends
Circle
+Circle()
+ getArea() : double
+ draw(Graphics) : void
Rectangle
+ getArea() : double
+ draw(Graphics) : void
+ Rectangle()
Sqaure
+ getArea() : double
+ draw(Graphics) : void
+ Square()
Main
+ main(): void
Creates shape when button is clicked
Interface starts
Stores shape and paints it
Panel
- Collection : ArrayList
+shapeCollection() : ArrayList<Shape>
+PaintComponent(Graphics) : void
SortingTechnique
+ areaSort(Panel): void
+ OrganiseShape(Panel) : void
ButtonPanel
- button1 : JButton
- button2 : JButton
- panel : Panel
+ButtonPanel()
+actionPerformed(ActionEvent)
Sorts when button is clicked

Elements of the above UML diagram are :

1. ACTOR : **(ButtonPanel.java)**
   In the above UML diagram, the class which interacts with the use cases, and which can be controlled by user or an external system.

2. USE CASES : **(SortingTechnique.java , ShapeFactory.java)**
   Here Sorting and loading shapes are user-visible functions performed by user (actor). These functions fulfill the user requirements as needed by mapping actors to the use cases.

3. ASSOCIATIONS **: (Buttonpanel to ShapeFactory, ButtonPanel to SortingTechnique)**
   It is an interacting connection between actors and use cases.
   So basically there are several communication links between them.

Class Relationships in my UML:

1) **Shape** : Shape is a parent class which has functions to create the shapes by using attrributes of specific shape. It has fields like size of a shape according to width and height. Color of the shape which is generated according to RGB. It has 2 methods of calculating area and uses draw method with parameter graphics which creates shape.

2) **Circle,Rectangle,Square** : These classes are childrens of above class shape. It inherits the attributes of Shape class and uses specific attributes to help creating shapes. It also has the same methods to calculate area of specific shape and draw the shape.

3) **ShapeFactory**: This class functions the spacing of 6 shapes which are generated eveytime when one clicks load shapes button. Also it has a method called Createshape() with parameter of panel class's object. It generates a random number 1,2 and 3 according to shapes to be generated and uses switch case to create shape using its parameters like dimension, color etc.

4) **SortingTechnique** : This class functions to sort the shapes when sorting button is clicked. It uses attributes of Panel with having 2 methods.First one is sorting according to area. I have used bubble sort algorithm to sort shapes according to the values of area. Second method functions to organise the shapes which uses shapeCollection() method of panel whose return type is arraylist. It organises shape when sorting is done with perfect spacing between different shapes.

5) **Panel :** This class has an arraylist with collection of shapes. It has 2 methods mainly, first one just return the collection of shape and second method is paintComponent() with parameter graphics, It colors the shape from shapeCollection()

6) **ButtonPanel:** This class extends Jpanel to inherit all the attributes which basically implements ActionListener.It basically implemets the interface which loads button and has certain action performed which leads to specific tasks. For example there are 2 actions performed where 1st is load shapes, which calls method createShape() from ShapeFactory and 2nd one is Sort shapes, which calls method areasort() from SortingTechnique.

7) **Main:** It just has a main method which runs the GUI.It generates 2 buttons on the interface initially.

Class diagrams:

Shape class:

- Shape is an abstract class and a superclass too.
- Shape has attributes like x,y,width and height with return type int and color with return type colour. They are privately declared.
- 2 methods named getarea() with return type double and draw() with return type void.

Square,Rectangle,Circle:

- All are subclasses of shape class. It has inheritance relaationship.
- All methods and attributes are same like shape class.

ButtonPanel :

- It as an interface class and a superclass too.
- It inherits Jpanel and implements ActionListener.
- 2 private declared buttons with type Jbutton and a Panel with type panel.
- It also has a method named Actionperformed with return type void.

SortingTechnique:

- It has 2 methods namely areasort() and Organiseshape() with return type void.
- Its declared publically.

ShapeFactory:

- It has 3 public static final attributes named NO_OF_SHAPE, SPACE, SHAPE_MIN with type int.
- Moreover has method Createshape with return type void.

The Design Principles that are used in the project:

Inheritance:

The concept of Inheritance has been used mainly in the modeled section of the project. Inheritance includes a base class which has all the common attributes and functionalities being used by the subclasses in the later part of the project. Here Shape.java is the parent class I designed while Circle.java, Square.java and Rectangle.java are sub-classes that inherit the Shape class respectively. Also ButtonPanel inherits Jpanel.

Polymorphism:

There is polymorphism used in my project.The object made "new" is used in many forms. For an example panel is used in shapefactory to create an object of its type.Hence objects of various classes are used in eachtoher. Also Graphics g has been used in various shapes, hence while using paintcomponent(), specific method is called.

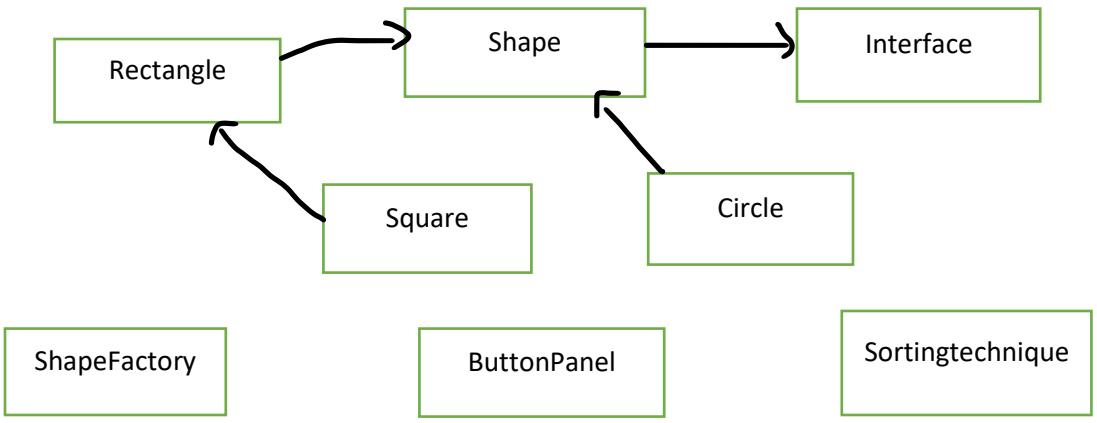
Encapsulation:

There is Encapsulation used in my project. In class shape where, its hidden from a user that how shapes are created by making its attributes private. Also ButtonPanel has certain private fields which acts as an actor in my software project.

Abstraction:

Abstraction is used mainly everywhere. Shape class implements comparable. Also Buttonpanel implemets ActionListener which generates specific functions using method calls like loading and sorting. Hence there are abstract classes used in my project.

Alternate UML:

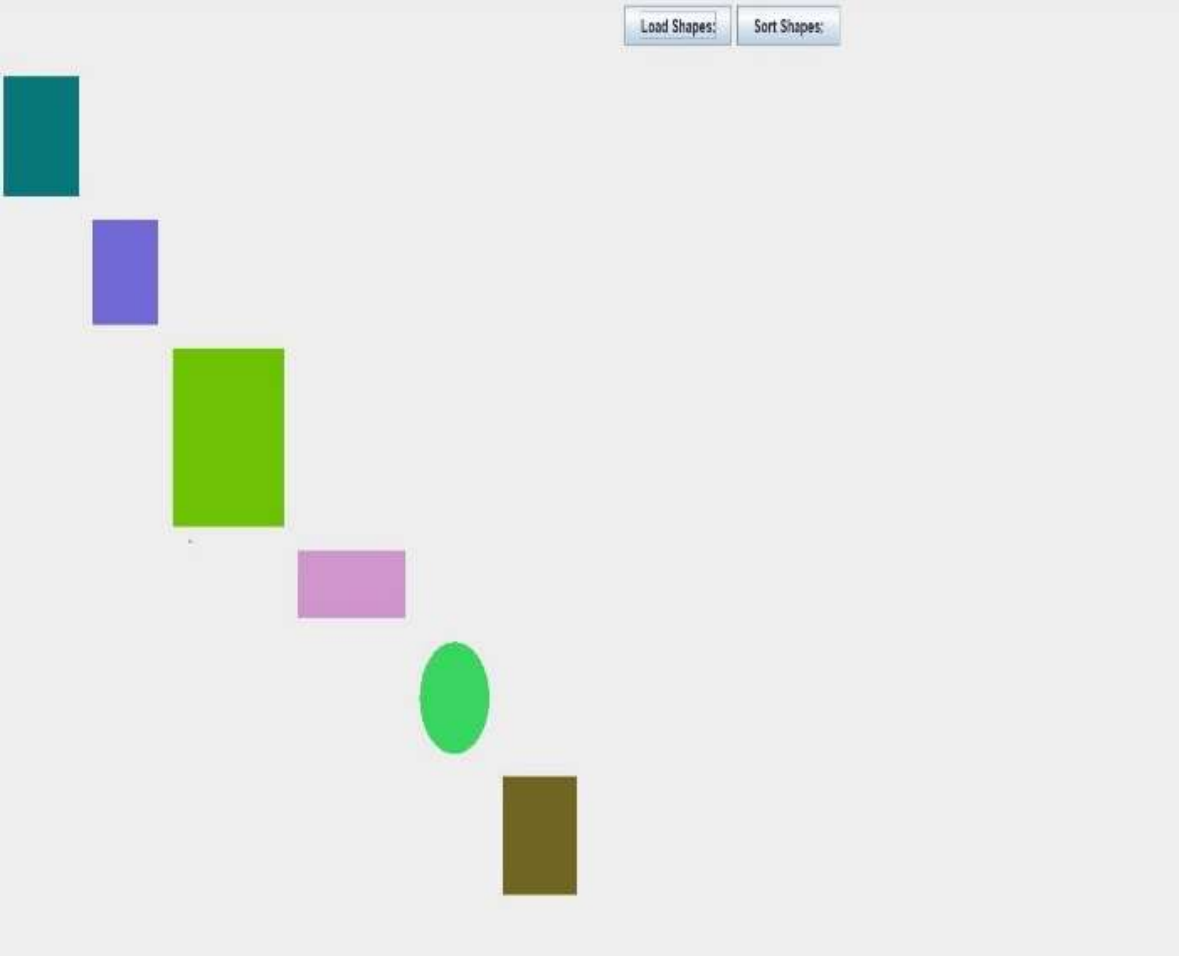


Here Square class is inherited from rectangle which basically extends Shape. Circle extends Shape and shape is defined as an Interface. This will reduce some coding part and would be implemented more easily.

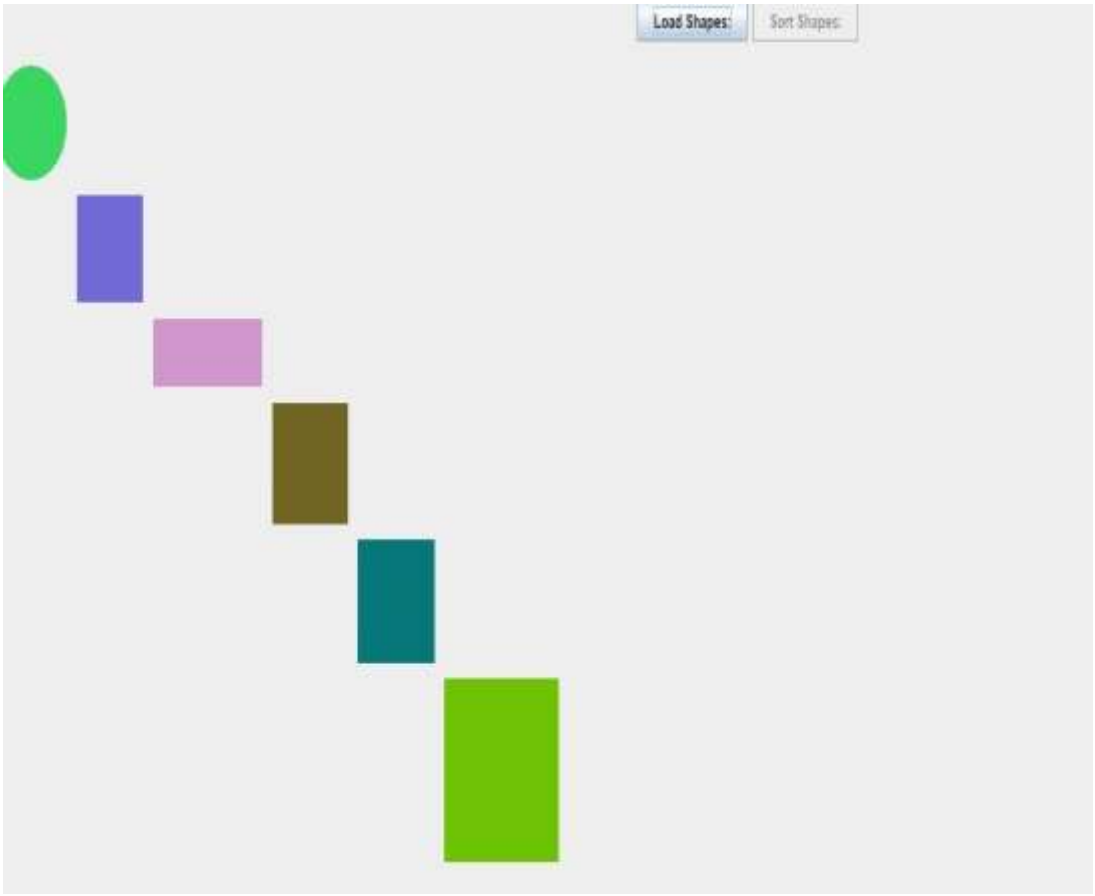
IMPLEMENTATION:

Sortingtechnique: Here I have used simple bubble sort algorithm to sort the shapes. It basically compares 2 adjacent shapes, takes the run time of O(n^2). It basically has a for loop which runs till total number of shapes. It checks with alternate shape, if the shape is larger then swaps it with the smaller one. This smaller shape again sorts with adjacent shape. If the shape is larger it goes back. In the same way we get the largest shape at the end.Hence we repeat this step for every shape except the last one. Hence this method sorts the shapes when button is clicked according to its area.

My whole work is done in Eclipse IDE, version of JDK is 13.



LOAD SHAPE SNAPSHOT



SORT SHAPE SNAPSHOT

Video can be found on Github.

CONCLUSION:

This was my First Software project. Firstly getting familiar with swing framework was a challenging part. Using OOD principles and creating UML are some of the things which went well in my software project. Also using various design pattern was something new I learnt.

Talking about challenging part, using all 4 OOD principles was challenging part. Creating shapeFactory class was difficult because it required lots of debugging to manage space between shapes, reorganising shapes and generating random number to create shape etc.

As I mentioned above, I have learned new design patterns, implemented it in eclipse by creating UML. So by this project I learned how we can design a project when requirements are given. Also I learned how to push,pull and commit while working on eclipse. Got familiar with swing framework.

My three reccomendations would be :

1) To have working hands on OOPs concept and OOD in java.
2) Also reading various books on design pattern and how to implement it from scratch.
3) Try to learn new things everytime by using various online resources like youtube,stackoverflow.