Name : Hassan Hassan

Student ID : 215216930

Course : LE EECS 3311

Section : A

Software Project : SoftwareProject1

Part 1 : Introduction

This software project is meant to develop a program that can load various shapes (e.g. Rectangle , Square , Circle etc.) on clicking a button and then should be able to sort all those shapes on the basis of their surface areas.

The goal behind the development of this project is to able to figure out :

* How the various shapes are drawn using the Graphics class
* How the shapes are arranged on a container using x and y coordinates.
* How the size (width and height) of shapes are managed.
* How the surface of a shape , is calculated.
* How the various shapes are sorted , based on their surfaces.
* How the shapes are rearranged after performing their sorting.

The following challenges associated to the software project :

* Figure out what classes are need to be defined to achieve the goal.
* Figure out how all the classes are integrated to meet the requirements.
* Determining how the GUI is designed.
* Determining how to draw various shapes and what should be their sizes and locations.
* Selecting the best sorting technique to sort all the shapes.
* Choosing the event to trigger the sorting of the shapes.
* How to paint the shapes.
* How to repaint the shapes after sorting.

The following concepts of OOD is used :

* Encapsulation

All the classes are designed using the concept of encapsulation , like they have their own data sets that is needed to design an object of that particular class. For example : Each object of Rectangle class will have its own set of x , y , width , height and colour properties.

* Abstraction

All the instance variables of the classes are made private and we have provided setter and getter methods to access or modify the properties of an object indirectly.

* Inheritance

Inheritance is a tool to reuse a software component as parent / child relation.Here we have made use of inheritance rigorously like the top level class Shape is inherited to the Rectangle class since a Rectangle is a Shape , the Square class extends the Rectangle class since the Square is a Rectangle with equal width and height and the Circle class extends the Square class since a Circle is essentially a circle drawn touching the inner boundaries of a Square.

* Polymorphism

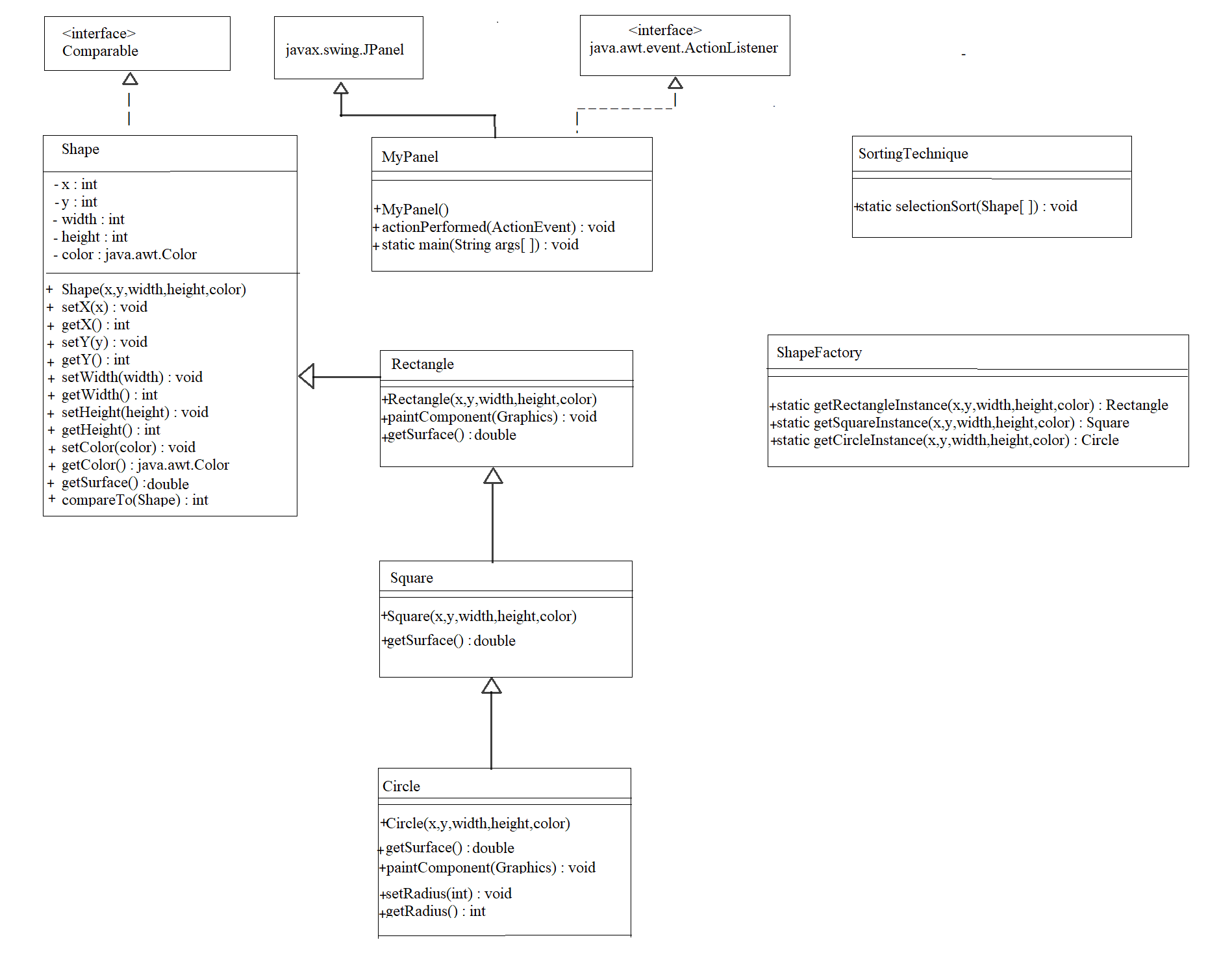
“One name many forms”

In this project , the getSurface() of Shape class is made abstract that is meant to be defined by all of its subclasses personally.

So whenever we create an object of any shape and calls the getSurface() , then it is automatically linked to the respective class , the object belongs to.

Part 2 : Design of the solution

UML class diagram



Basic components of a UML class diagram

* Upper section: Contains the name of the class.
* Middle section: Contains the properties or attributes of the class.
* Bottom section: Includes class operations (methods).

Access attributes are specified as

+ public access

* private access

# protected access

------------ represents that the class or interface implements an interface.

\_\_\_\_\_\_\_\_ represents that the class extends another class.

Basic elements of a UML class diagram

Class : set of objects having identical responsibilities. A class embodies a concept which encapsulates state(attributes) and behavior(operations).

Interface : set of operations specifying the responsibility of a class.

Object Oriented Design principles

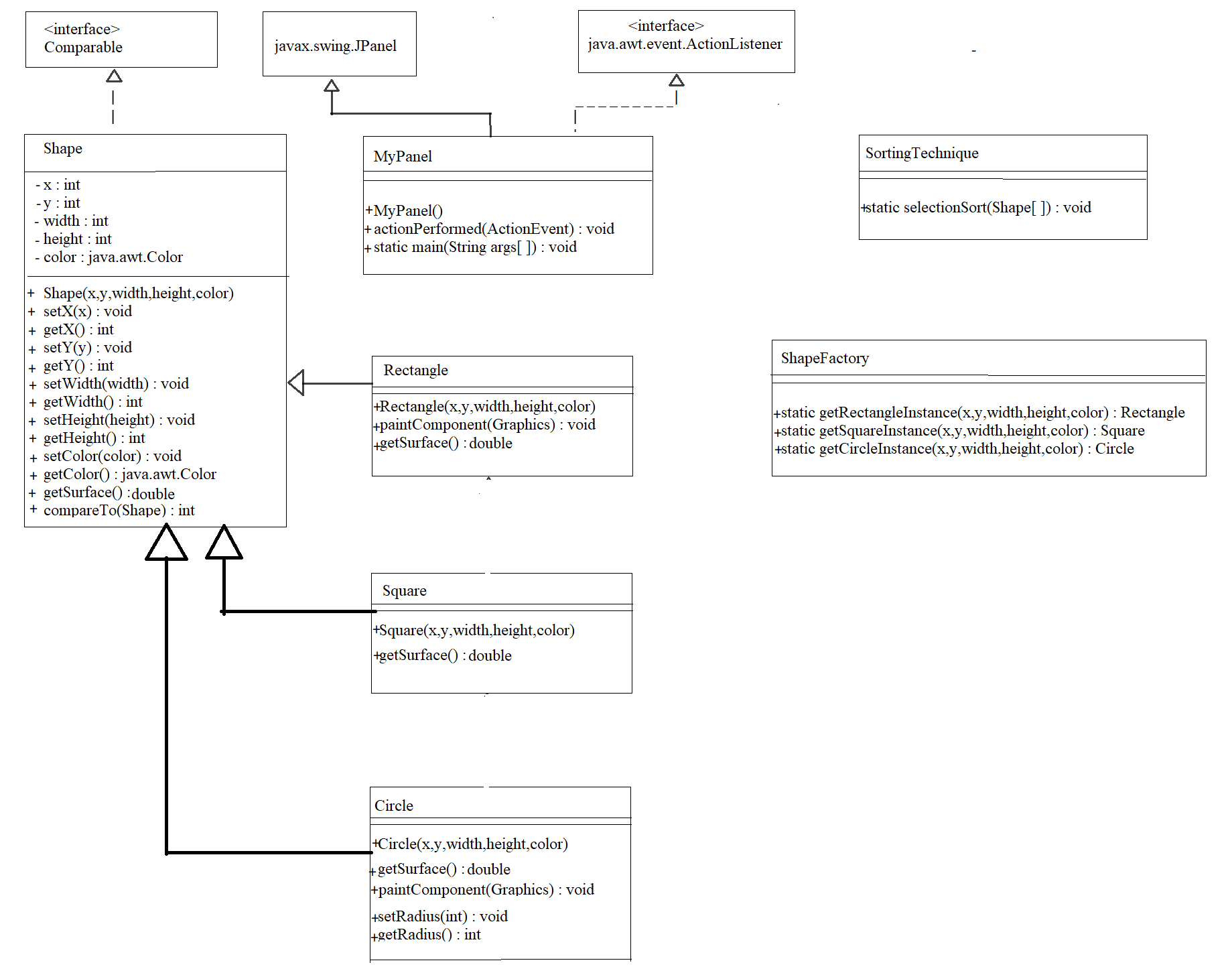
Encapsulation : Shape class contains the location coordinates named x and y , size of the shape named width and height , and color of the shape as the attributes of the class , the setter and getter methods , compareTo() and getSurface() as mechanism , all at one place.

Abstraction : All the attributes in the Shape class has made private so that they can not be used outside the class directly , but can be used indirectly using the setter and getter methods.

Inheritence : Shape is the parent class that extends the JPanel and implements the Comparable interface.Rectangle extends the Shape as a Rectangle is a Shape , Square extends the Rectangle as a Square is essentially a Rectangle with equal width and height , and the Circle extends the Square as a Circle is drawn by touching the inner boundaries of a Square shape.

Polymorphism : getSurface() is declared as an abstract method in Shape class , so all of its direct or indirect subclasses define that method accordingly and it is invocation is decided during runtime , depending upon the object automatically.

Design alternative UML class diagram



Part 3 : Implementation of the solution

Sorting algorithm : Selection sort

**Step 1** − Set MIN to location 0

**Step 2** − Search the minimum element in the list

**Step 3** − Swap with value at location MIN

**Step 4** − Increment MIN to point to next element

**Step 5** − Repeat until list is sorted

Here i have implemented first UML class diagram.

Shape class

This class extends the javax.swing.JPanel class and implements the Comparable interface.It has encapsulated the location coordinates x and y , size dimensions as width and height and color of the shape.It has setter and getter methods of all the attributes of the class in addition with the getSurface() that returns the surface area of the shape.The getSurface() is an abstract method that is intended to be designed by its subclasses actually.This class has a public constructor that takes location , diemnsion and color as argument and creates an object of the Shape accordingly.

Rectangle class

This class is subclass of Shape class so has rights to access all the properties of Shape class.It provides a parameterized constructor that simply transfers the call to the Shape class parameterized constructor.It has a paintComponent() that draws a Rectangle shape for the given width and hight at the given x and y coordinates with the given color.It has overridden the getSurface() method of Shape class and returns the surface area of the Rectangle object.

Square class

This class is subclass of Rectangle class so has rights to access all the properties of Rectangle class.It provides a parameterized constructor that simply transfers the call to the Rectangle class parameterized constructor.It uses the paintComponent() method of Rectangle class that draws a Sqaure as a Rectangle with equal width and height at the given x and y coordinates with the given color.It has overridden the getSurface() method of Rectangle class and returns the surface area of the Square object.

Circle class

This class is subclass of Square class so has rights to access all the properties of Square class.It provides a parameterized constructor that simply transfers the call to the Square class parameterized constructor.It defines the setRadius() and getRadius() methods to set or get the radius as width/2 of the circle object.It defines the paintComponent() method of Shape class (indirect parent class) that draws an Oval as a Square with equal width and height at the given x and y coordinates with the given color.It has overridden the getSurface() method of Sqaure class and returns the surface area of the Circle object.

SortingTechnique class

This class provides a static method named selectionSort() that takes an array of Shape objects and sort all those Shape objects in the increasing order of their surface area using the selection sort mechanism.

ShapeFactory class

This class provides three static methods named getRectangleInstance() , getSquareInstance() and getCircleInstance() that returns the corresponding Shape objects for the given coordinates as x and y , size dimensions as width and height and given color.

MyPanel class

This is the main class that creates the UI.It has a constructor that creates a UI having a JPanel that contains two buttons with label “Load shapes” and “Sort shapes”.It implements ActionListener interface to manage the ActionEvent of the ose two buttons.When user clicks on the load button , it creates six shapes for the random sizes and places those onto a panel that is displayed on the UI.When user clicks on the sort button , all those shapes are sorted using the selectionSort() of SortingTechnique class and then rearranges all the shapes onto the panel that is displayed on the UI.

All these classes have been placed in a package named “package1”.

All the classes are compiled and executed on Eclipse IDE.

Eclipse IDE for Java Developers

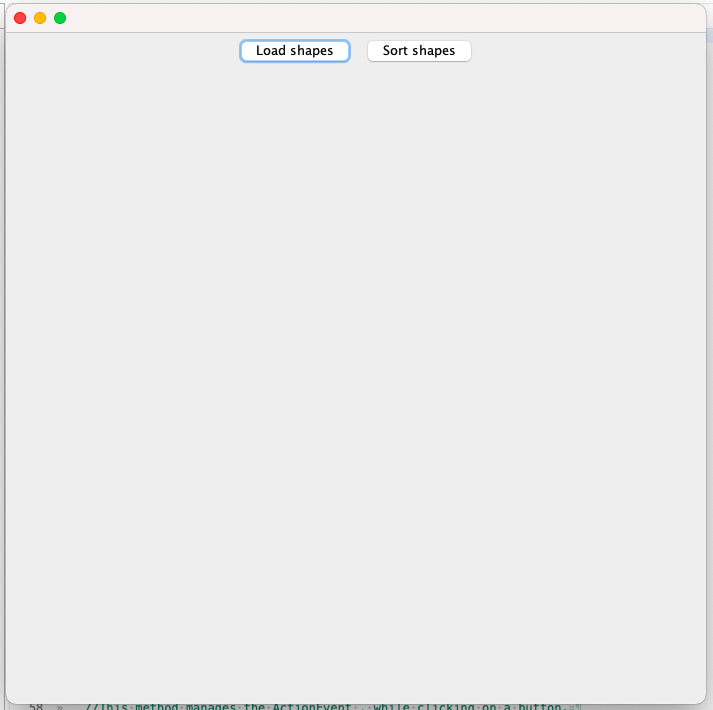
Version: 2020-12 (4.18.0)

Build id: 20201210-1552

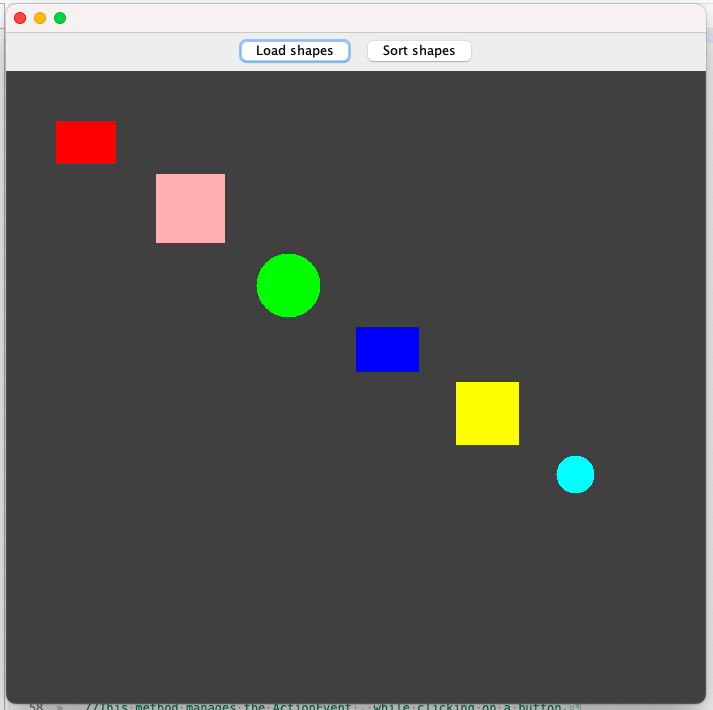
OS: macOS 11.1

Java version: 13.0.2

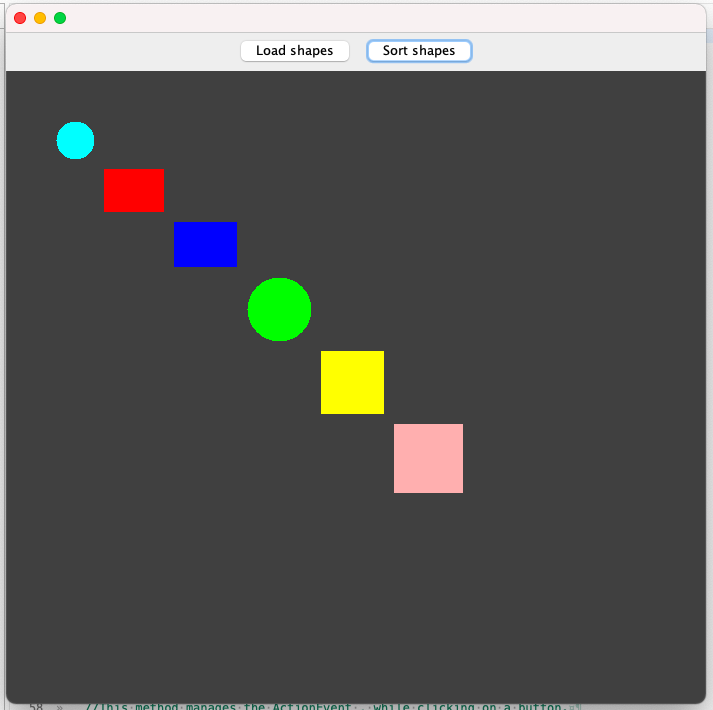
Initial startup of the application



While clicking on “Load shapes” button



While clicking on “Sort shapes” button



Part 4 : Conclusion

The inheritence and polymorphism principles of OOD helped here a lot.I created the classes using the principal of inheritence so that Circle class inherited the Square , The Square class inherited the Rectangle and the Rectangle class inherited the Shape class.Shape class provided an abstract method getSurface() that is defined by all of those subclasses according to that particular shape and while creating an oject of that particular Shape , the respective getSurface() is choosen automatically due to the polymorphism.

Initailly I tried placing the shape components onto the JPanel using a layout manager , but later I found that it is not possible to place the components in the desired order using the layout manager , so I choose not to use any layout manager and placed all the shape components by providing manual locations using x and y coordinates.

I learned how to use the inheritence and polymorphism in a real project and how to develop a gui and sort various objects using the Comparable interface.

I recommend to use the inheritence principle along with the polymorphsim to reduce the coding and achieve the integrity.