ThePainter V1.0

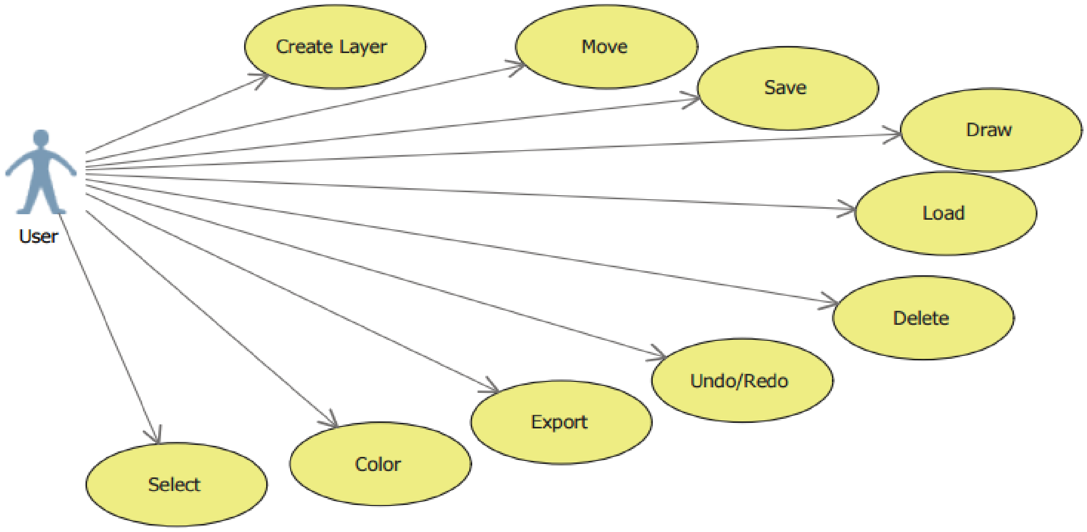
Software Documentation for technical users

Introduction

ThePainter V1.0 is an- Object Oriented model- application based, with advanced GUI and 2D Graphics capabilities, that helps users to draw, color and edit a number of built in, extensible set of geometric shapes.

UML

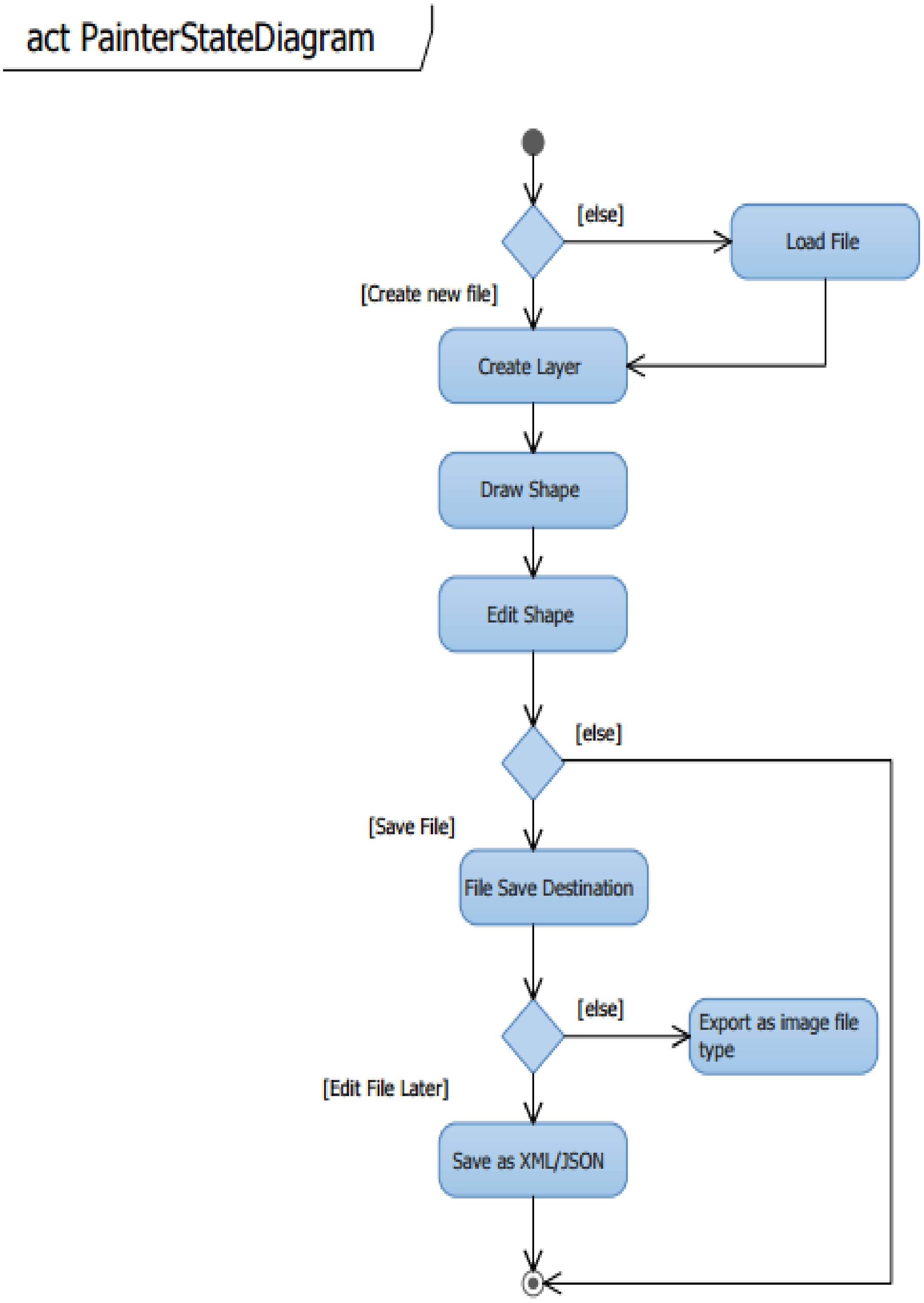
Use Case Diagram



Use Case Description

Use Case Name	Draw
Participating Actors	Paint user
Flow of Events	1-User creates new canvas 2-User chooses shape 3- User chooses color 4- User Draws shapes 5-User saves image 6-User exits program
Alternative Flows	4.a User select shape 4.b User Move shape 4.c User resize shape 4.d User undo and redo changes
Exit conditions	User finishes drawing and save image

State Diagram



Classes

Design Pattern

ThePainter uses Model-View-Controller (MVC) design pattern for dividing the application into three interconnected parts. Each component of the MVC (whether Model, View, or controller) has its own package with inside classes hierarchy; managing the data, logic and rules of the application.

Model

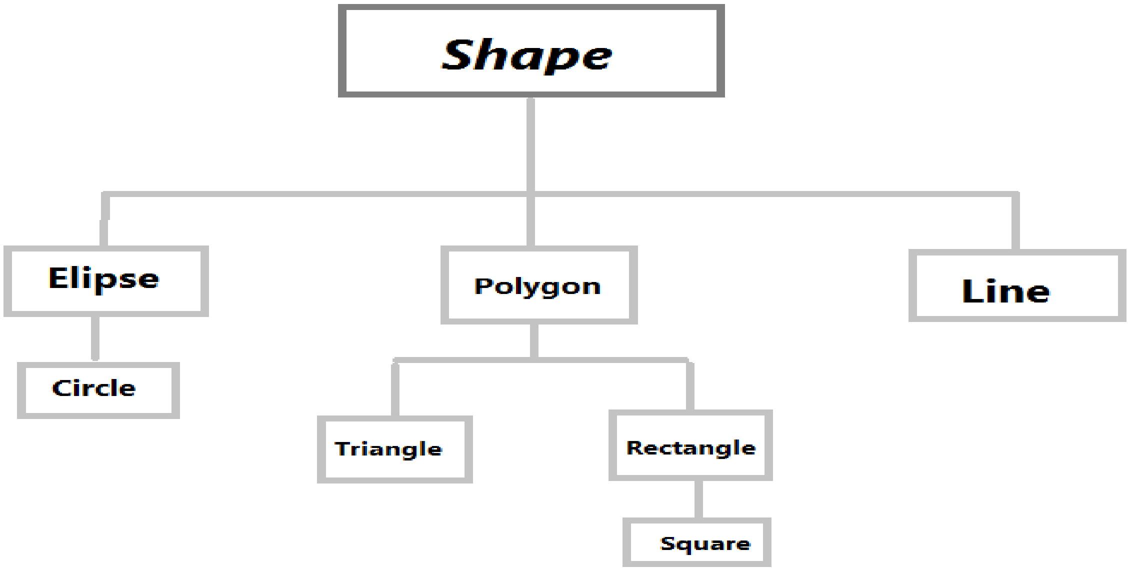
Consists of two packages:

1- Model.Shapes

2-Model.Layers

- 1.Model.Shapes classes in the MVC pattern
  a) *Abstract* class Shape.
  b) Public class Ellipse extends Shape
  c) Public class Polygon extends Shape
  d) Public class Line extends Shape
  e) Public class Circle extends Ellipse
  f) Public class Rectangle extends Polygon
  g) Public class Triangle extends Polygon
  h) Public class Square extends Rectangle

- Model Classes hierarchy



- 2.Model.Layers classes in the MVC pattern
  - Public class Painter
  - Public class Layer extends JPanel implements MouseListener
  - Public class Canvas extends JPanel

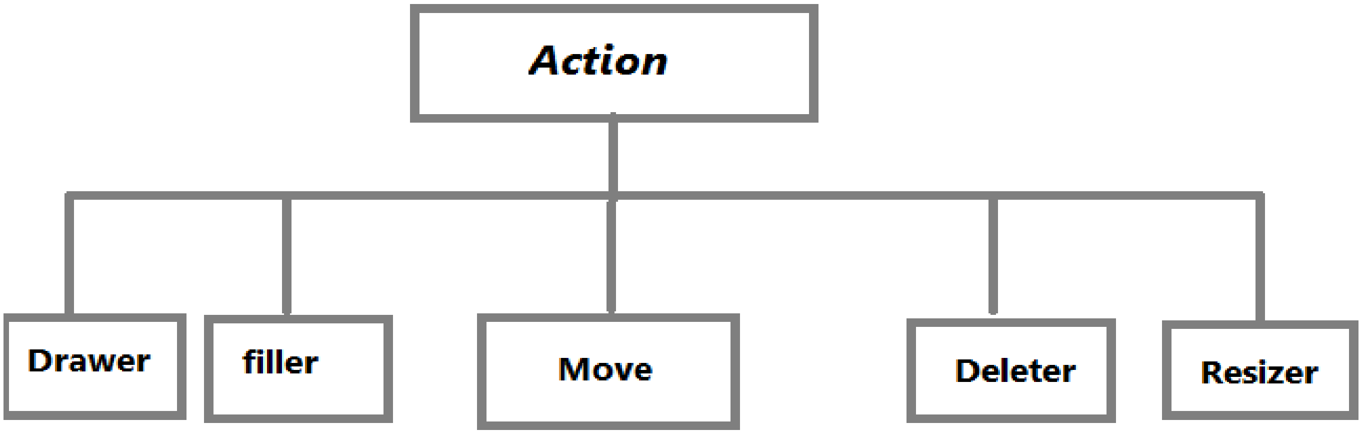
View

- View classes in the MVC pattern
  1. Public class LayerStyle extends JPanel
  2. Public final class ColorChooser. Extends JColorChooser
  3. Public class ButtonStyle.
  4. Public class GUI extends JFrame

Controller

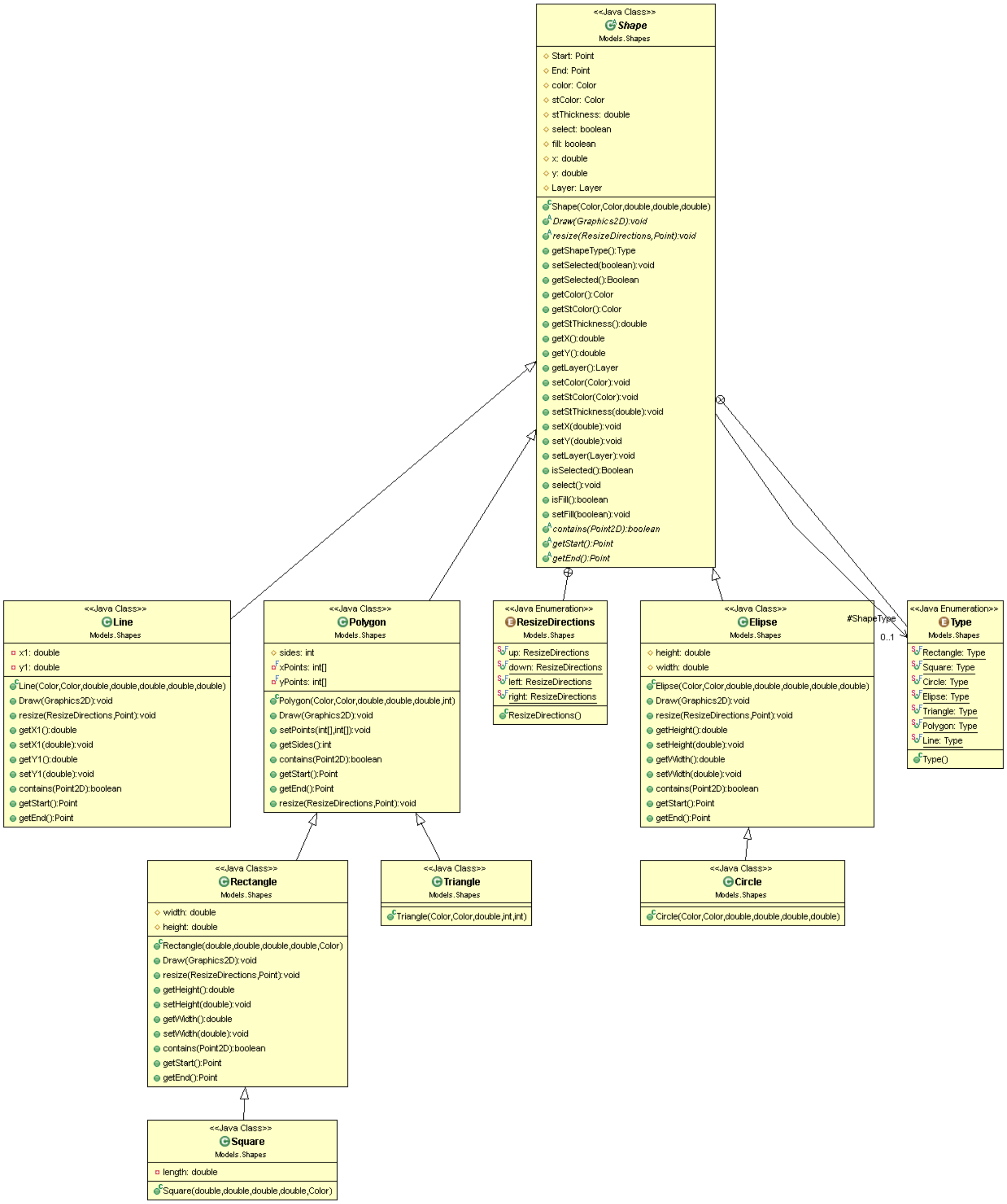
- Controller classes in the MVC pattern
    1. Public class ThePainter.
    2. Public abstract class Action.
    3. Public class Drawer extends Action
    4. Public class Filler extends Action
    5. Public class Deleter extends Action
    6. Public class Resizer extends Action
    7. Public class UndoRedo.
    8. public class Move extends Action

- View Classes hierarchy

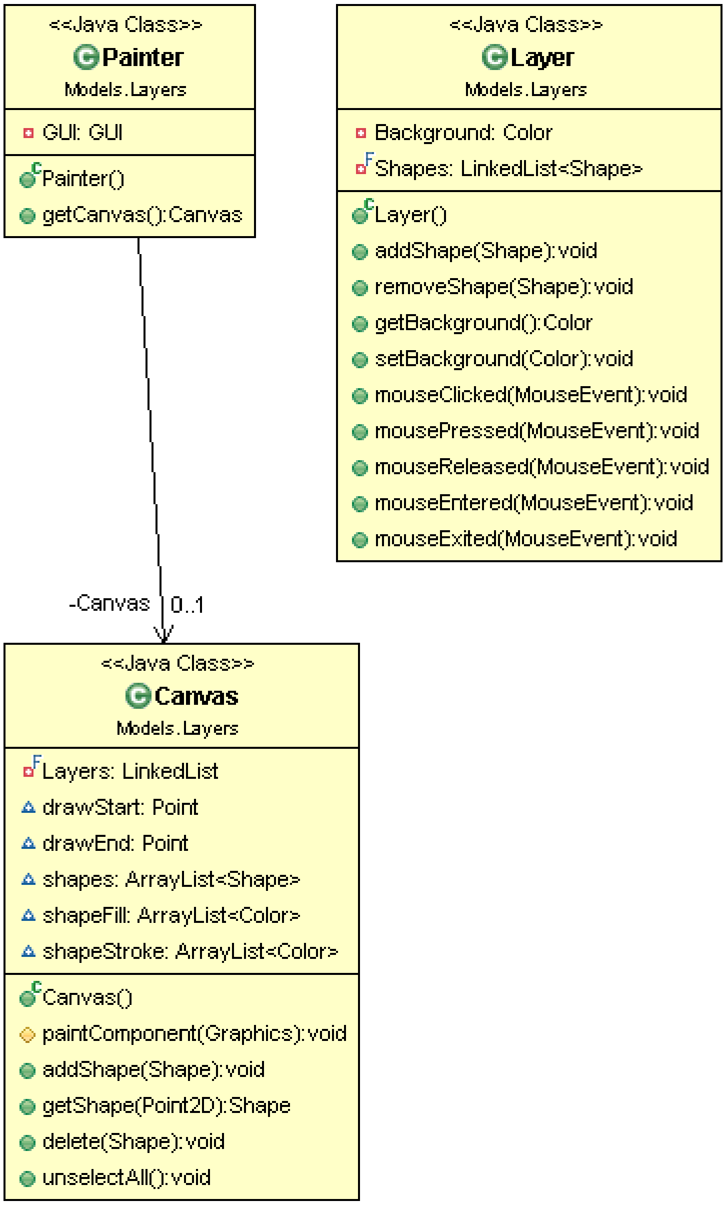


Class Diagrams

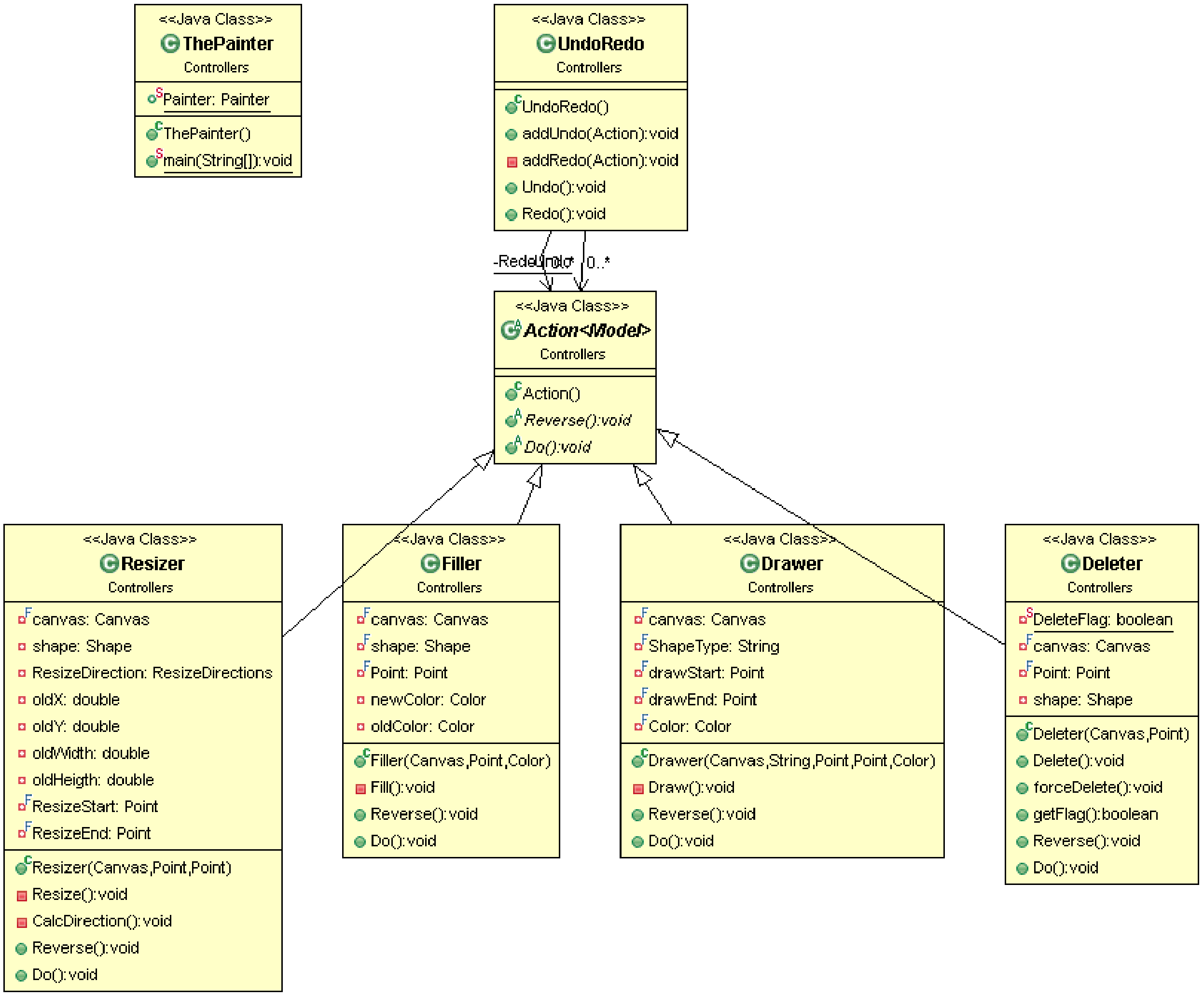
Model.Shapes



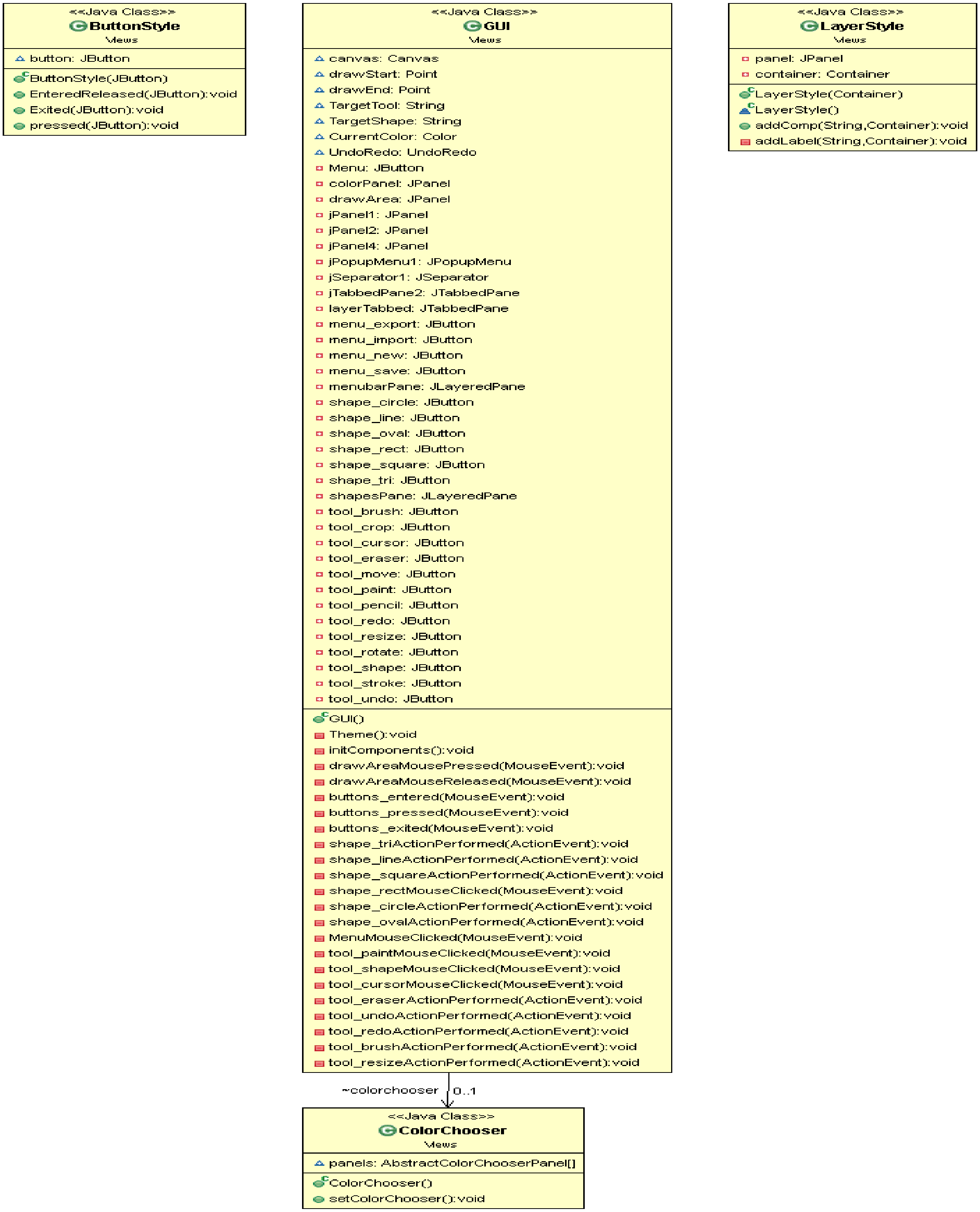
Model.Layers



Controller


<<Java Class>>
ThePainter
Controllers
Painter: Painter
ThePainter()
main(String[]):void
<<Java Class>>
UndoRedo
Controllers
UndoRedo()
addUndo(Action):void
addRedo(Action):void
Undo():void
Redo():void
-RedoUndo*
0..*
<<Java Class>>
Action<Model>
Controllers
Action()
Reverse():void
Do():void
<<Java Class>>
Resizer
Controllers
canvas: Canvas
shape: Shape
ResizeDirection: ResizeDirections
oldX: double
oldY: double
oldWidth: double
oldHeigth: double
ResizeStart: Point
ResizeEnd: Point
Resizer(Canvas,Point,Point)
Resize():void
CalcDirection():void
Reverse():void
Do():void
<<Java Class>>
Filler
Controllers
canvas: Canvas
shape: Shape
Point: Point
newColor: Color
oldColor: Color
Filler(Canvas,Point,Color)
Fill():void
Reverse():void
Do():void
<<Java Class>>
Drawer
Controllers
canvas: Canvas
ShapeType: String
drawStart: Point
drawEnd: Point
Color: Color
Drawer(Canvas,String,Point,Point,Color)
Draw():void
Reverse():void
Do():void
<<Java Class>>
Deleter
Controllers
DeleteFlag: boolean
canvas: Canvas
Point: Point
shape: Shape
Deleter(Canvas,Point)
Delete():void
forceDelete():void
getFlag():boolean
Reverse():void
Do():void

View


<<Java Class>>
ButtonStyle
Views
button: JButton
ButtonStyle(JButton)
EnteredReleased(JButton):void
Exited(JButton):void
pressed(JButton):void
<<Java Class>>
GUI
Views
canvas: Canvas
drawStart: Point
drawEnd: Point
TargetTool: String
TargetShape: String
CurrentColor: Color
UndoRedo: UndoRedo
Menu: JButton
colorPanel: JPanel
drawArea: JPanel
jPanel1: JPanel
jPanel2: JPanel
jPanel4: JPanel
jPopupMenu1: JPopupMenu
jSeparator1: JSeparator
jTabbedPane2: JTabbedPane
layerTabbed: JTabbedPane
menu_export: JButton
menu_import: JButton
menu_new: JButton
menu_save: JButton
menubarPane: JLayeredPane
shape_circle: JButton
shape_line: JButton
shape_oval: JButton
shape_rect: JButton
shape_square: JButton
shape_tri: JButton
shapesPane: JLayeredPane
tool_brush: JButton
tool_crop: JButton
tool_cursor: JButton
tool_eraser: JButton
tool_move: JButton
tool_paint: JButton
tool_pencil: JButton
tool_redo: JButton
tool_resize: JButton
tool_rotate: JButton
tool_shape: JButton
tool_stroke: JButton
tool_undo: JButton
GUI()
Theme():void
initComponents():void
drawAreaMousePressed(MouseEvent):void
drawAreaMouseReleased(MouseEvent):void
buttons_entered(MouseEvent):void
buttons_pressed(MouseEvent):void
buttons_exited(MouseEvent):void
shape_triActionPerformed(ActionEvent):void
shape_lineActionPerformed(ActionEvent):void
shape_squareActionPerformed(ActionEvent):void
shape_rectMouseClicked(MouseEvent):void
shape_circleActionPerformed(ActionEvent):void
shape_ovalActionPerformed(ActionEvent):void
MenuMouseClicked(MouseEvent):void
tool_paintMouseClicked(MouseEvent):void
tool_shapeMouseClicked(MouseEvent):void
tool_cursorMouseClicked(MouseEvent):void
tool_eraserActionPerformed(ActionEvent):void
tool_undoActionPerformed(ActionEvent):void
tool_redoActionPerformed(ActionEvent):void
tool_brushActionPerformed(ActionEvent):void
tool_resizeActionPerformed(ActionEvent):void
<<Java Class>>
LayerStyle
Views
panel: JPanel
container: Container
LayerStyle(Container)
LayerStyle()
addComp(String,Container):void
addLabel(String,Container):void
~colorchooser
0..1
<<Java Class>>
ColorChooser
Views
panels: AbstractColorChooserPanel[]
ColorChooser()
setColorChooser():void

Functions and subroutines

Model Classes:

Public abstract class Shape

The parent class contains all attributes and functions to be inherited to all geometric shapes

- Constructor: (Color color, Color stColor, double stThickness, double x, double y)
- public abstract void Draw(Graphics2D g);
- public abstract void resize(ResizeDirections direction, Point newPoint)
- public Boolean isSelected();
- public abstract boolean contains(Point2D Point);
- public abstract Point getStart();
- public abstract Point getEnd();
- public enum Type
- public enum ResizeDirections
- Getters and Setters for :
  - ShapeType :Type
  - Start :Point
  - End : Point
  - color: Color
  - stColor : Color
  - stThicknessL double
  - x: double
  - y: double
  - select: Boolean
  - fill: booolean

Public class Elipse extends Shape

Drawing an Ellipse with a 2D point, height and width

- Constructor: (Color color, Color stColor, double stThickness, double x, double y, double height, double width)
  - Calling Super Constructor class
- Override methods:
  - Draw(Graphics2D g);
  - resize(ResizeDirections direction, Point newPoint);
  - contains(Point2D point);
  - getStart();
  - getEnd();

Public class Circle extends Elipse

Drawing circle, extending all functions and attributes of Elipse class

- Constructor: (Color color,Color stColor, double stThickness, double x, double y, double radius)
    - Calling Super Constructor

Public class Line extends Shape

Drawing a line with start points and end points x1,y1 x2,y2 respectively

- Constructor: (Color color, Color stColor, double stThickness, double x,double x1, double y, double y1)
    - Calling Super constructor
- Override methods:
    - Draw(Graphics2D g);
    - resize(ResizeDirections direction, Point newPoint);
    - contains(Point2D point);
    - getStart();
    - getEnd();

Public class Polygon extends Shape

Drawing regular shape with n sides, taking array of int[] as points of x and y

- Constructor: Color color, Color stColor,
  double stThickness, double firstPointX, double firstPointY ,int sides)
    - Calling Super constructor
- public void setPoints(int[] xArr, int[] yArr)
    - Set different points to int[] array for drawing polygon edges
- Override methods:
    - Draw(Graphics2D g);
    - resize(ResizeDirections direction, Point newPoint);
    - contains(Point2D point);
    - getStart();
    - getEnd();

Public class Rectangle extends Polygon

Second child class extends Polygon, for drawing rectangle with start point, height and width

- Constructor: (double x1, double y1, double x2, double y2, Color color)
  - Calling Super constructor
- Override methods
  - Draw(Graphics2D g);
  - resize(ResizeDirections direction, Point newPoint);
  - contains(Point2D point);
  - getStart();
  - getEnd();

Public class Square extends Rectangle

Third child class extends Rectangle, drawing a square with length equals to height

- Constructor: (double x1, double y1, double x2 , double y2, Color color)
  - Calling Super constructor

Public class Triangle extends Polygon

Second child class extends Polygon , drawing a three side polygon(triangle)

- Constructor : (Color color, Color stColor,
  double stThickness, int firstPointx, int firstPointy)
  - Calling super Constructor

View Classes:

Public class GUI

Creating new form GUI, initializing components and handling mouse events

- Constructor: initializing Container, LayerStyle, colorChooser
- private void drawAreaMousePressed(MouseEvent evt);
- private void drawAreaMouseReleased(MouseEvent evt);
- private void buttons_entered(.MouseEvent evt);
- private void buttons_pressed(MouseEvent evt) ;
- private void buttons_exited(MouseEvent evt) ;
- private void MenuMouseClicked(MouseEvent evt) ;
- private void tool_paintMouseClicked(.MouseEvent evt) ;

Public final class ColorChooser extends JColorChooser

Set and Get color from the Color Chooser panel

- public ColorChooser();
- public void setColorChooser();

Public class ButtonStyle

Handling buttons size, color and after click events

- Constructor: (JButton button)
- public void EnteredReleased(JButton button)
- public void Exited(JButton button)
- public void pressed(JButton button)

----

public class LayerStyle extends JPanel

Adding labels and components to layer

- Constructor: initializing new Container
- public void addComp (String text, Container container)
- private void addLabel(String text, Container container)

Controller classes

public abstract class Action<Model>

Parent class implementing abstract methods to keep tracking of actions for Redo and Undo methods

- public abstract void Reverse();
- public abstract void Do();

----

public final class Deleter extends Action

Delete selected shape, or force delete

- Constructor: (Canvas canvas, Point Point)
- public void Delete();
- public void forceDelete();
- Override functions:
  - public void Reverse();
  - public void Do();

----

public class Drawer extends Action

Drawing in the Canvas according to shape type, color,start and end points data taken from GUI

- Constructor: (Canvas canvas, String ShapeType, Point drawStart, Point drawEnd,Color Color)
- private void Draw()
- Override functions:
  - public void Reverse();
  - public void Do();

----

public class Filler extends Action

Fill shapes with the color specified from GUI

- Constructor: (Canvas canvas,Point Point,Color newColor)
- private void Fill();
- Override functions:
  - public void Reverse();
  - public void Do();

public class Resizer extends Action

Resizing shapes in canvas and calculating direction of change

- Constructor: (Canvas canvas, Point ResizeStart, Point ResizeEnd)
- private void Resize();
- private void CalcDirection();
- Override functions:
    - public void Reverse();
    - public void Do();

----

public class UndoRedo

Implementing stacks for Undo and Redo actions

- Constructor: initialing new 2 stacks
- public void addUndo(Action action)
- private void addRedo(Action action)
- public void Undo()
- public void Redo()

----

public class Move extends Action

Moving shape from point to point inside the canvas

- Constructors: `(Canvas canvas, Point MoveStart, Point MoveEnd)`
- private void Move()
- private String getType(Shape shape)
- Override functions:
    - public void Reverse();
    - public void Do();