Drawing and Painting Application

FINAL PROJECT

Ahmed Ashraf Abdelkarim	6940
Ahmed Yasser Mohamed	6856
Adel Yasser Yassin	6848
Mostafa Mohamed Elkassas	6992

General overview:

This program is mainly used for drawing different geometrical shapes as well as free hand drawing using brush. By implementing the OOP concepts & principles.

Classes and their attributes:

In this program different classes were used in order to make the code more organized and readable; the following classes are the main ones used in our program.

BetterShape:

This class contains an object shape and its attributes as variables, as such:

1. int stroke
2. Color color
3. Boolean fill
4. Shape s
5. Boolean brush
6. float alpha

It also has public get methods for each of those private variables.

DynamicStack:

This is an abstract class with six useful abstract methods for interacting with a dynamically sized stack.

DynamicStackBetterShape:

This class extends the DynamicStack abstract class and is responsible for creating dynamic stacks containing BetterShape objects.

DynamicStackInt:

This class extends the DynamicStack abstract class and is responsible for creating dynamic stacks containing primitive int objects.

Square:

This class extends Rectangle2D.Float, the constructor initializes a square by using the start coordinates (x1,y1) that are made when the mouse is clicked. The side length is calculated. Then, the end coordinate x2 (after mouse dragging is lifted) is checked if it is larger than the start coordinates x1. If it is larger, we make a new start coordinate x1New by subtracting side length from x1. Else we use the original x1. Similarly, for y2, end coordinate y2 is checked if it is larger than the start coordinates y1. If it is larger, we make a new start coordinate y1New

by subtracting side length from y1. Else we use the original y1. Then the start coordinates and side length are used to construct the square object.

Rectangle:

This class extends Rectangle2D.Float, the constructor initializes a rectangle by using the start coordinates (x1,y1) that are made when the mouse is clicked, and the end coordinates (x2,y2) that are made when the mouse click is lifted after dragging. It compares and chooses the lower values between x1 and x2 and between y1 and y2 as the start coordinates, we also use the side lengths (Absolute value subtraction between x1 and y1 and between x2 and y2) as end coordinates.

Circle:

This class extends Ellipse2D.Float, the constructor initializes a circle by using the start coordinates (x1,y1) that are made when the mouse is clicked, the diameter length is calculated. Then, the end coordinate x2 (after mouse dragging is lifted) is checked if it is larger than the start coordinates x1. If it is larger, we make a new start coordinate x1New by subtracting side length from x1. Else we use the original x1. Similarly, for y2, end coordinate y2 is checked if it is larger than the start coordinates y1. If it is larger, we make a new start coordinate y1New by subtracting side length from y1. Else we use the original y1. Then the start coordinates and side length are used to construct the square object.

Ellipse:

This class extends Ellipse2D.Float, the constructor initializes a rectangle by using the start coordinates (x1,y1) that are made when the mouse is clicked, and the end coordinates (x2,y2) that are made when the mouse click is lifted after dragging. It compares and chooses the lower values between x1 and x2 and between y1 and y2 as the start coordinates, we also use the side lengths (Absolute value subtraction between x1 and y1 and between x2 and y2) as end coordinates.

Triangle:

This class extends Path2D.Float, the constructor initializes a triangle by using the start coordinates (x1,y1) that are made when the mouse is clicked, and the end coordinates (x2,y2) that are made when the mouse click is lifted after dragging. Side length is calculated by an absolute subtraction between x1 and y1 divided by 2. Then, we obtain an x coordinate in the middle (xMid) by checking if x2 is larger than x1, it adds side length to x1. else, it subtracts side length from x1. Then it goes to the start coordinates (x1,y1) and creates two lines, one from xMid to y2 and the other from x2 to y1.

Brush:

This class extends Path2D.Float, the constructor initializes a brush with the given int rule and coordinates (x1,y1) by calling super(rule), then moveTo(x1,y1) and lineTo(x1, y1).
The brushMove method is used to continue the path to the given coordinates (x2, y2), by calling lineTo(x2, y2) and returning itself.

ShapeFactory:

This class contains a method called createShape method that receives the following arguments (int tool, int x1, int y1, int x2, int y2, Shape r)
The method has a switch case for the value tool where:
Case 0: returns Brush object using Brush constructor
Case 1: returns Line using Line constructor
Case 2: returns Rectangle using Rectangle constructor
Case 3: returns Square using Square constructor
Case 4: returns Ellipse using Ellipse constructor
Case 5: returns Circle using Circle constructor
Case 6: returns Triangle using Triangle constructor
For case 8 and case 9, if r is null, it returns null,
Else,
Case 8: returns new Shape translated using AffineTransform
Case 9: returns new Shape scaled and translated using AffineTransform
Default: returns null.

RoundButton:

This class extends JButton and is used to create the round buttons found in the GUI as seen in the color choosing buttons. The class makes an oval with preferred dimensions and fills it with preferred color.

Gui:

This class has the following variables:
Boolean fill
Color color
Color colorBg
int tool // 0 brush, 1 line, 2 rectangle, 3 square, 4 ellipse, 5 circle, 6 triangle, 7 select, 8 move, 9 resize
Font subtitle, titleButton, labelButton
JSlider sliderOpacity, sliderStroke
Frame frame, frameColors
JPanel panelColorPreview

The createColourChooser method creates the color chooser window to enable the user to choose colors at astounding accuracy.
The createGui method is responsible for creating the Gui including every JLabel, JButton, round button and JSlider, along with the ActionListener for the buttons. It also creates a PaintSurface object.

The subclass PaintSurface extends JComponent:
This has DynamicStackBetterShape and DynamicStackInt objects which are used to contain the current stack of objects to be drawn as well as for undo/redo of any action.
It also has the following class variables:
Brush brushPath
int selectedShape
Point startDrag, endDrag
ShapeFactory shapeFactory

Adds a mouse listener in its constructor. The mouse listener listens to the following actions: press, drag and release. Each of these actions calls repaint() at the end, to update the PaintSurface.
The paintBackground method in PaintSurface is used to draw a white rectangle in the drawing area as a background.

The paint method (scheduled by the repaint method) first goes through each BetterShape object in the main DynamicStackBetterShape and using its attributes it draws the shape as intended.
Then if there is a selected object, draw a rectangular dashed gray outline around that object.
Finally, if startDrag and endDrag don't equal null (user has the mouse pressed inside the painting area) depending on the tool and if there is a selected shape it will draw an appropriate preview.

The undo method uses the various stacks to undo any action done and then call repaint().
The redo method uses the various stacks to redo any action done and then call repaint().
The makeShape method first creates Shape r which is given the value null if no shape is selected otherwise it points to the selected shape from the stack. If the brushPath isn't null then it returns brushPath.moveTo(x2, y2). Otherwise, if the tool is 0 (brush) then it sets the value of brushPath to equal shapeFactory.createShape(tool, x1, ...) casted to Brush. Otherwise, return shapeFactory.createShape(tool, x1, …).
The selectShape method goes through the main DynamicStackBetterShape stack from top to bottom and if the mouse is inside the 2D rectangular bounds of the object, then the object is selected, if the mouse is inside none, then no object is selected.
The deleteShape method deletes the selected object and does nothing if no shape is selected. It is overloaded with another that takes the argument index and deletes the object with the given index.

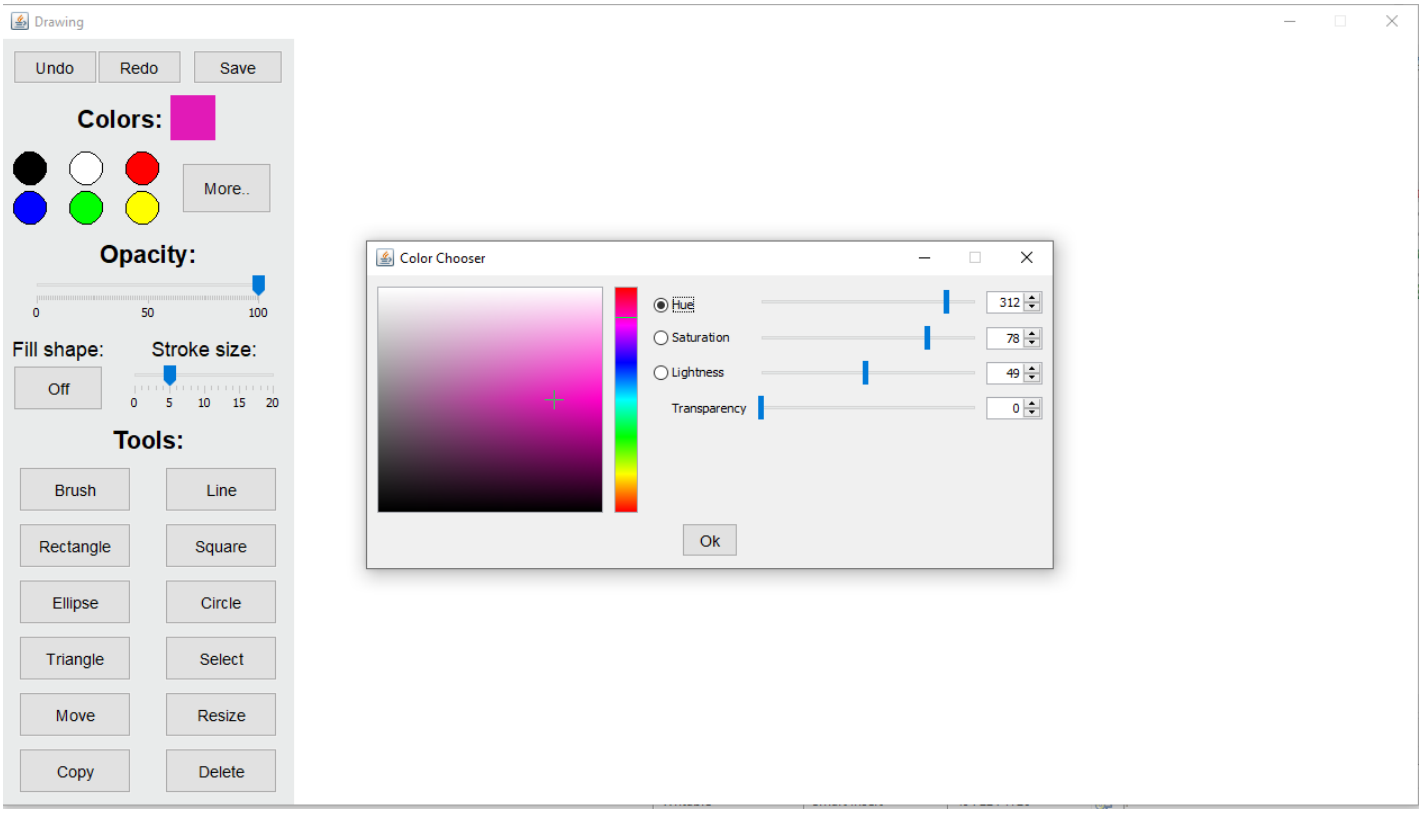
The copyShape method does nothing if no shape is selected, otherwise it will duplicate the selected shape and push it to the top of the main stack.
The clearHistory method clears the redo stacks, this method is called when the mouse is pressed on the paint surface.
The save method creates a BufferedImage and calls createGraphics method on it, then paints it with the paintSurface; finally, it will create a PNG image file named output_image.png, if a file with that name already exists it increments until it finds a file name that doesn't exist (output_image*.png).

Main:

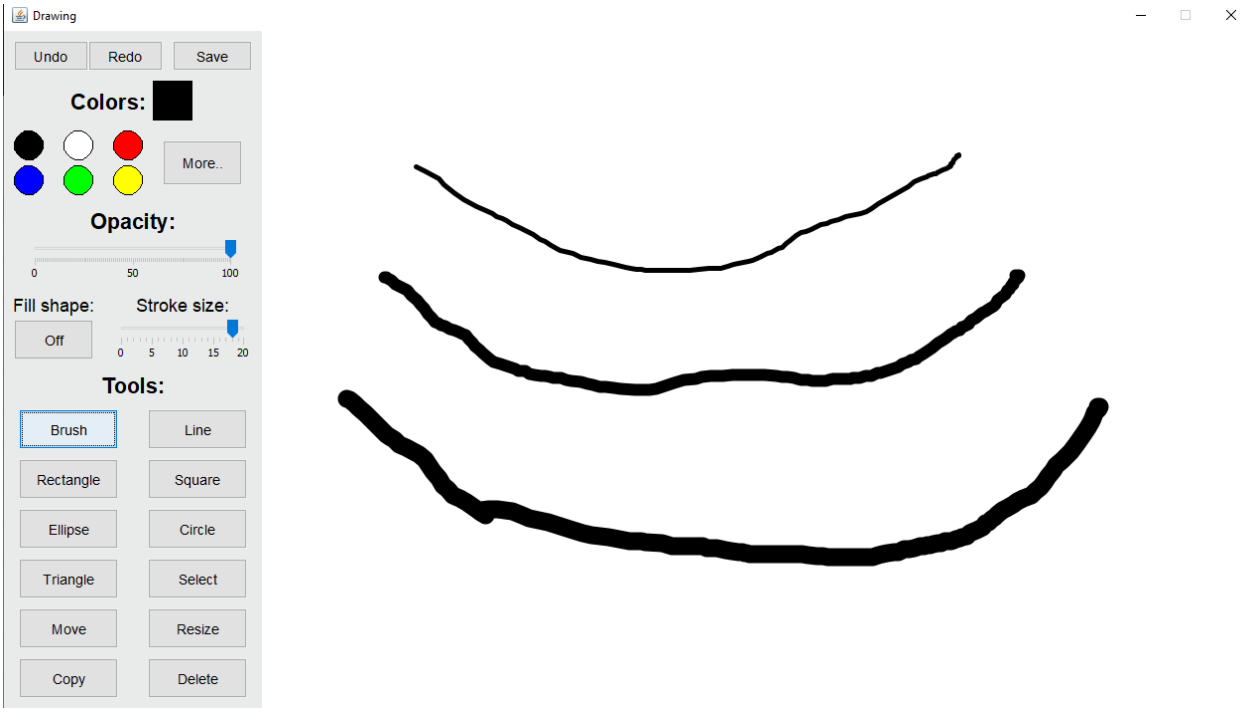
This class has the main method which creates a Gui object in the variable gui, then calls the createGui method from gui.

GUI Snapshots:

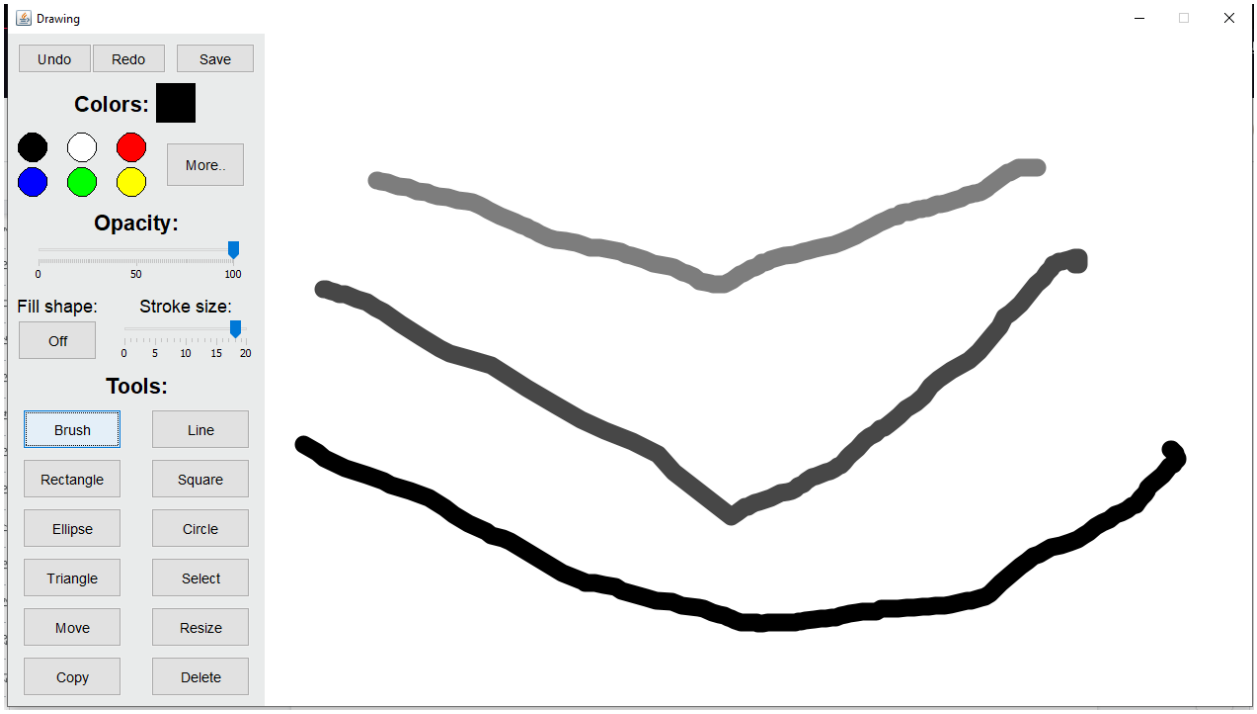
More Colors:



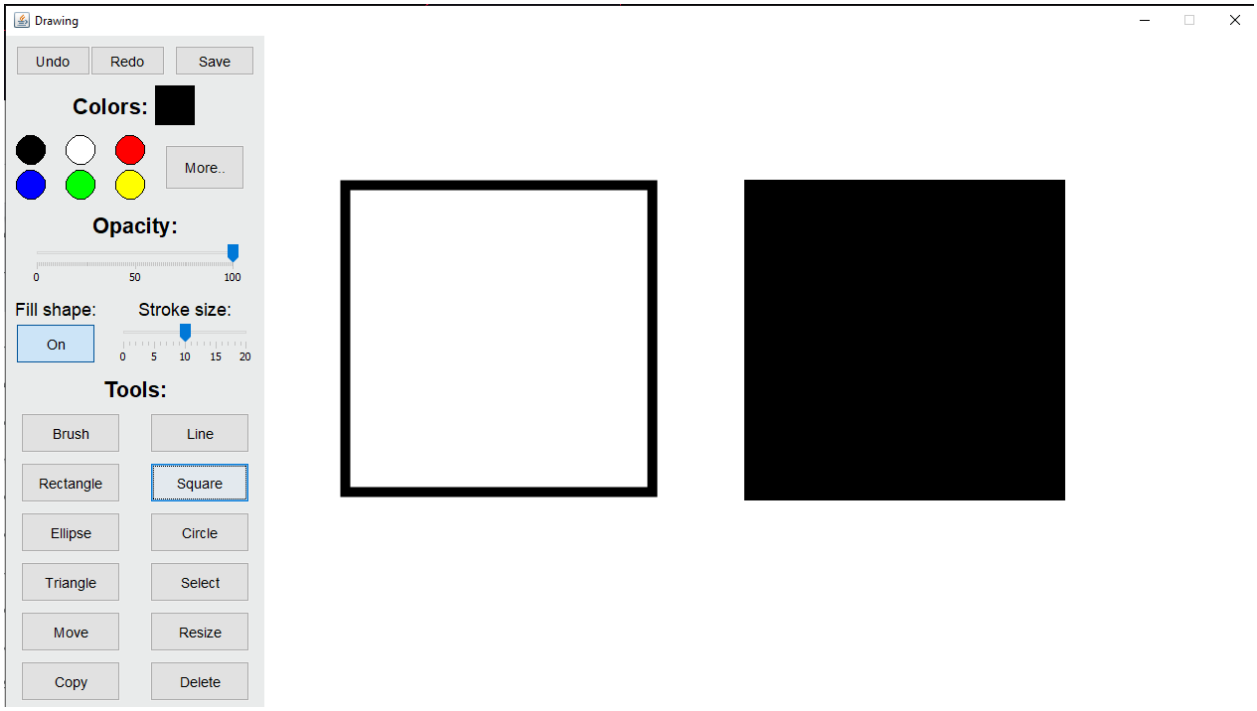
Stroke size:



Opacity:

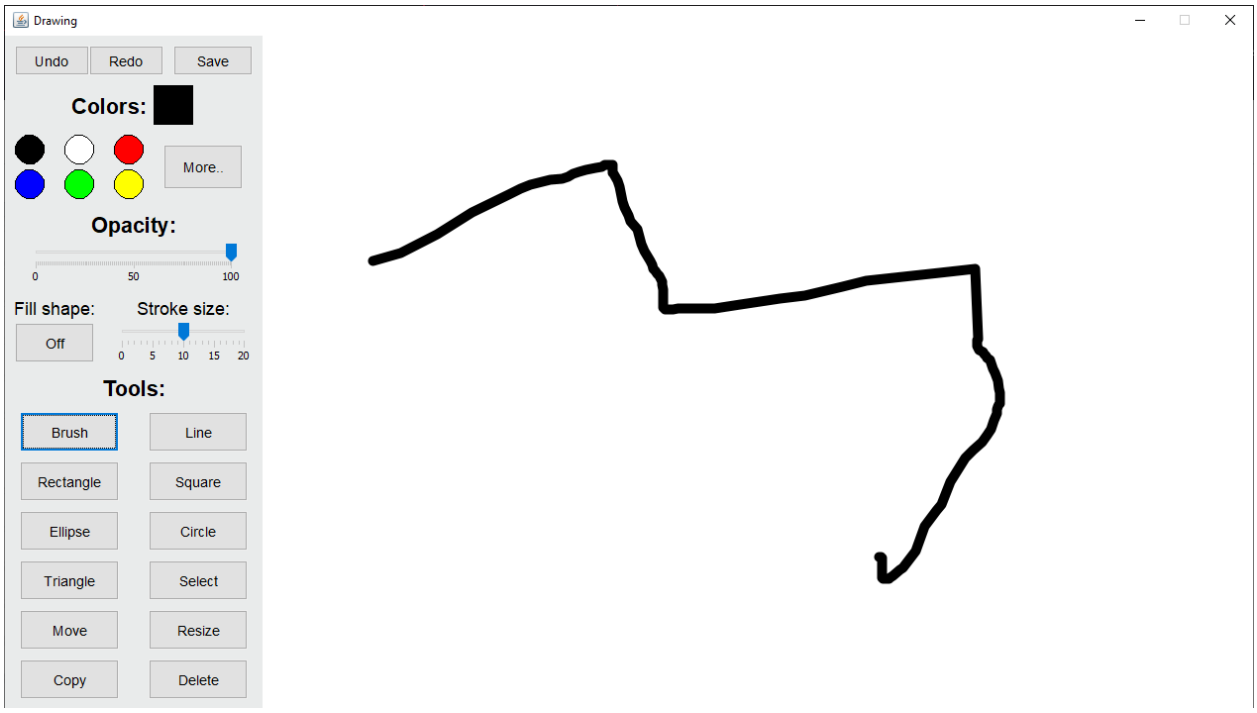


Fill shape:

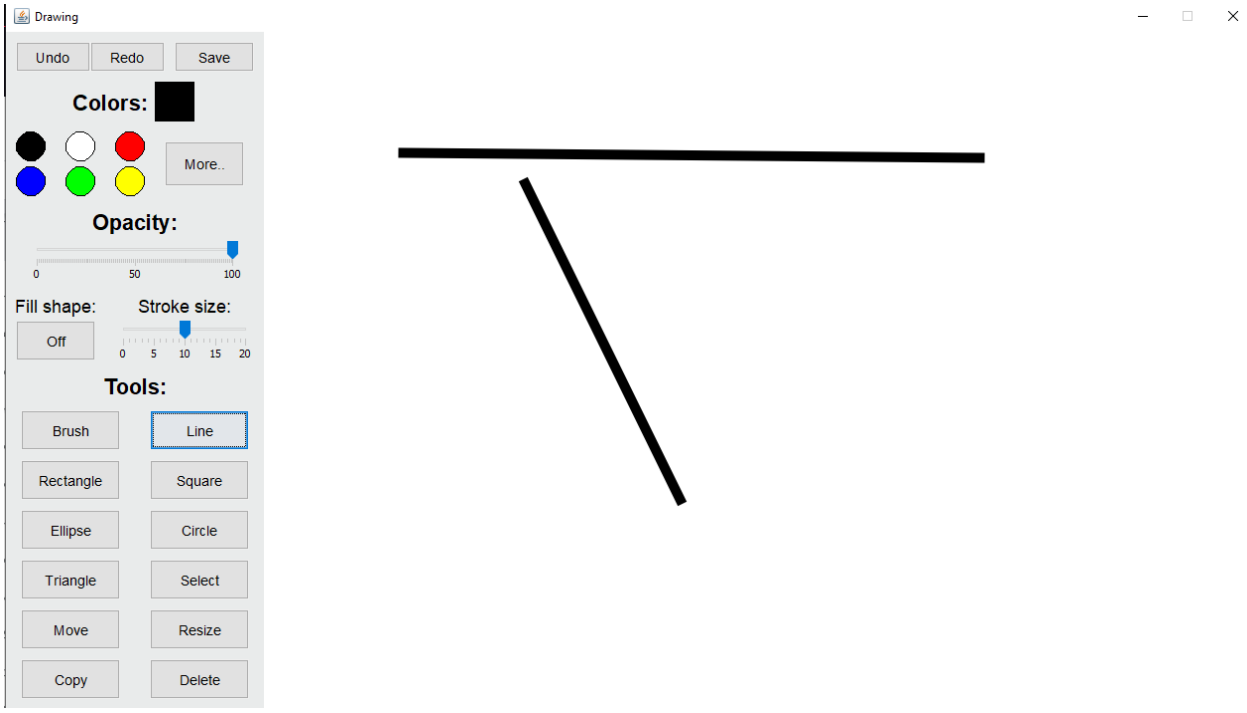


TOOLS

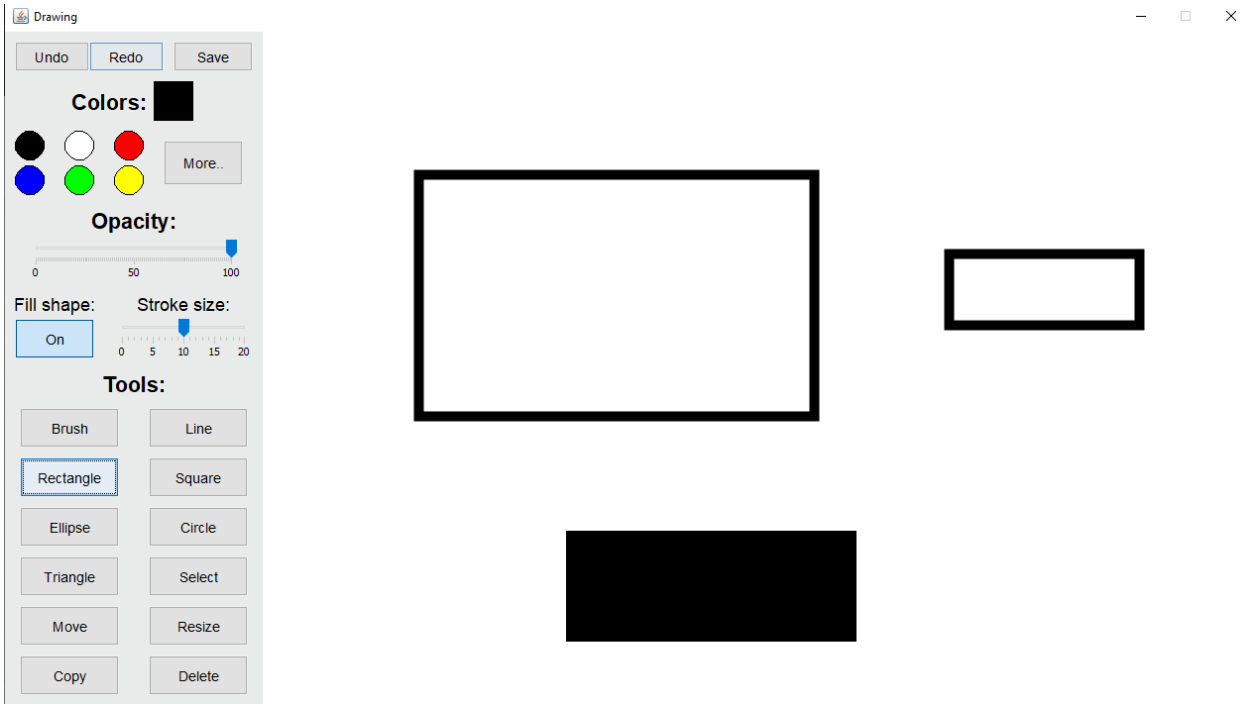
Brush:



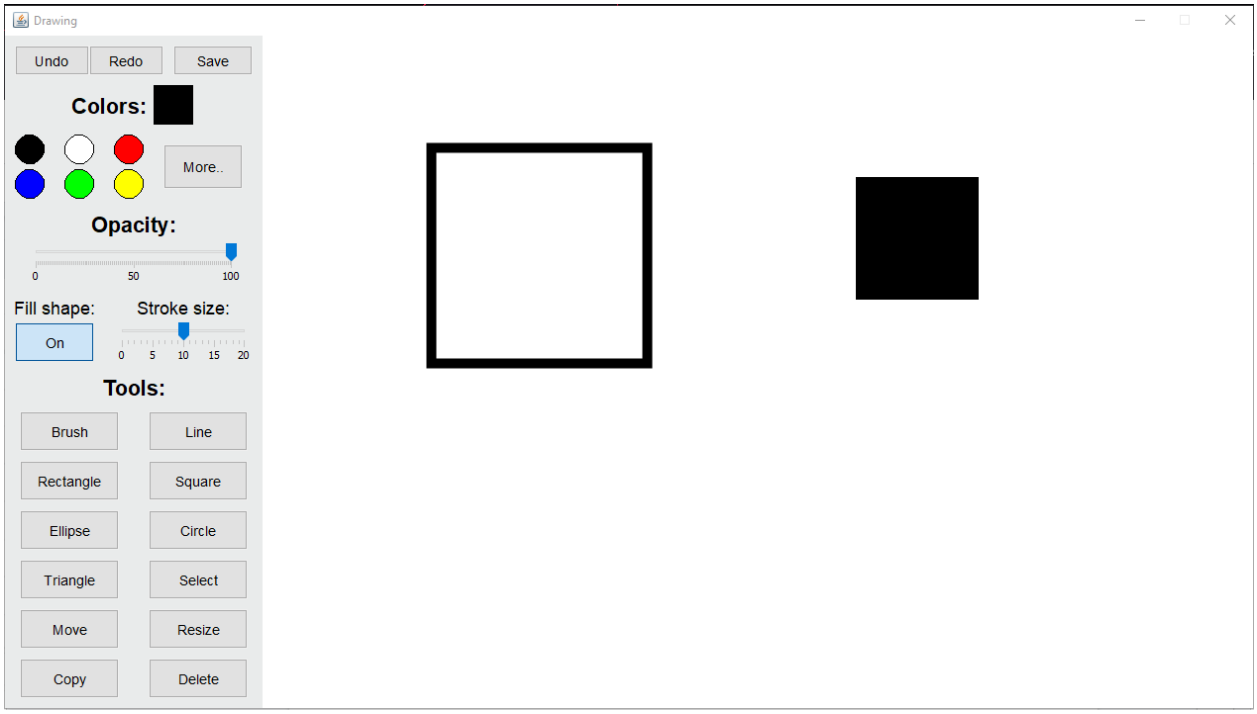
Line:



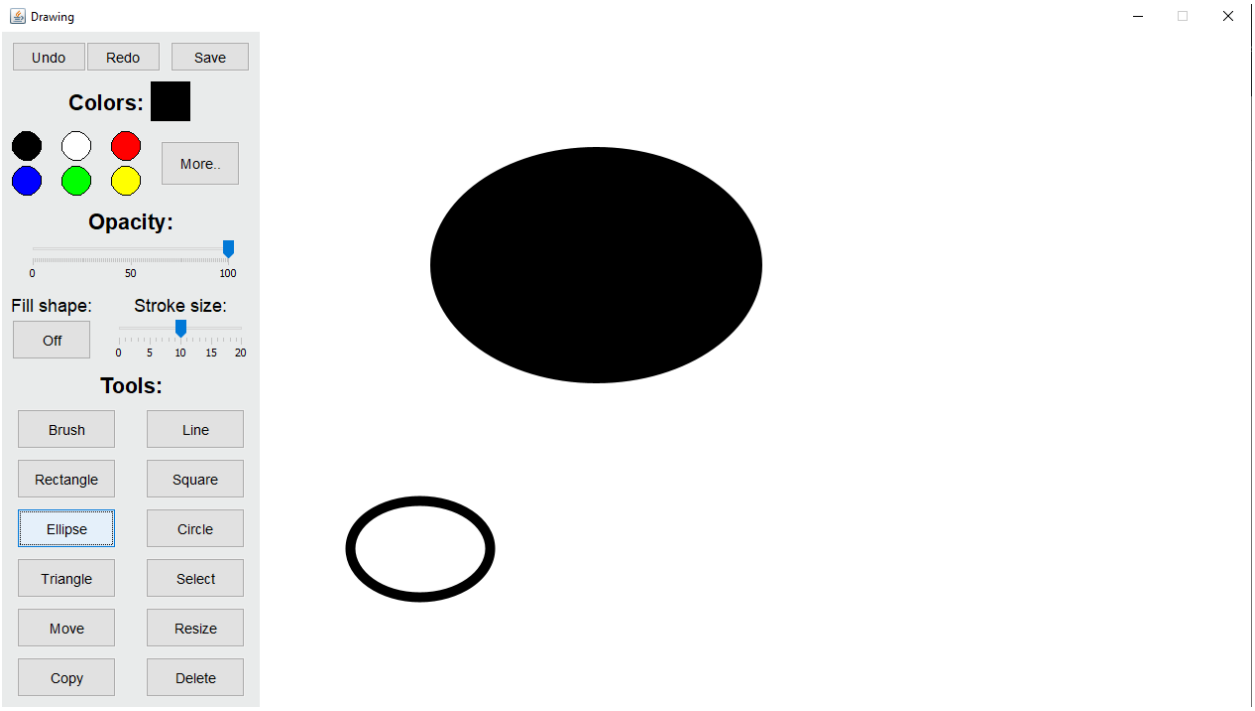
Rectangle:



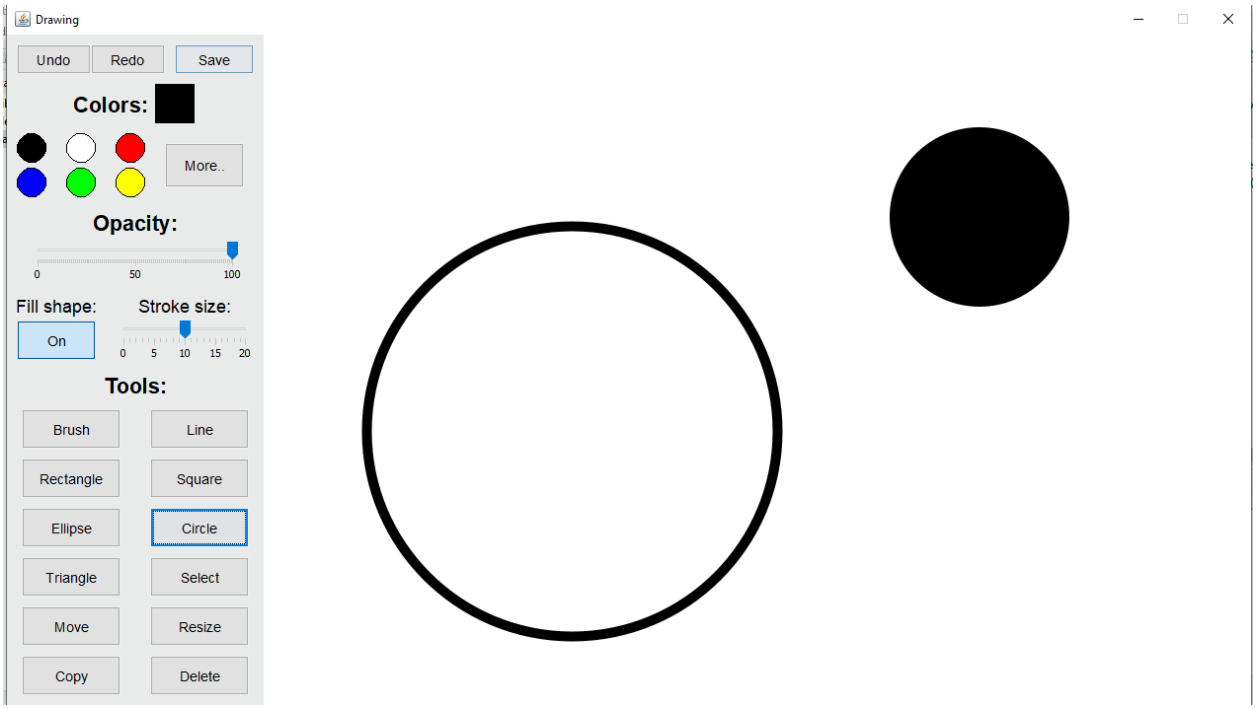
Square:



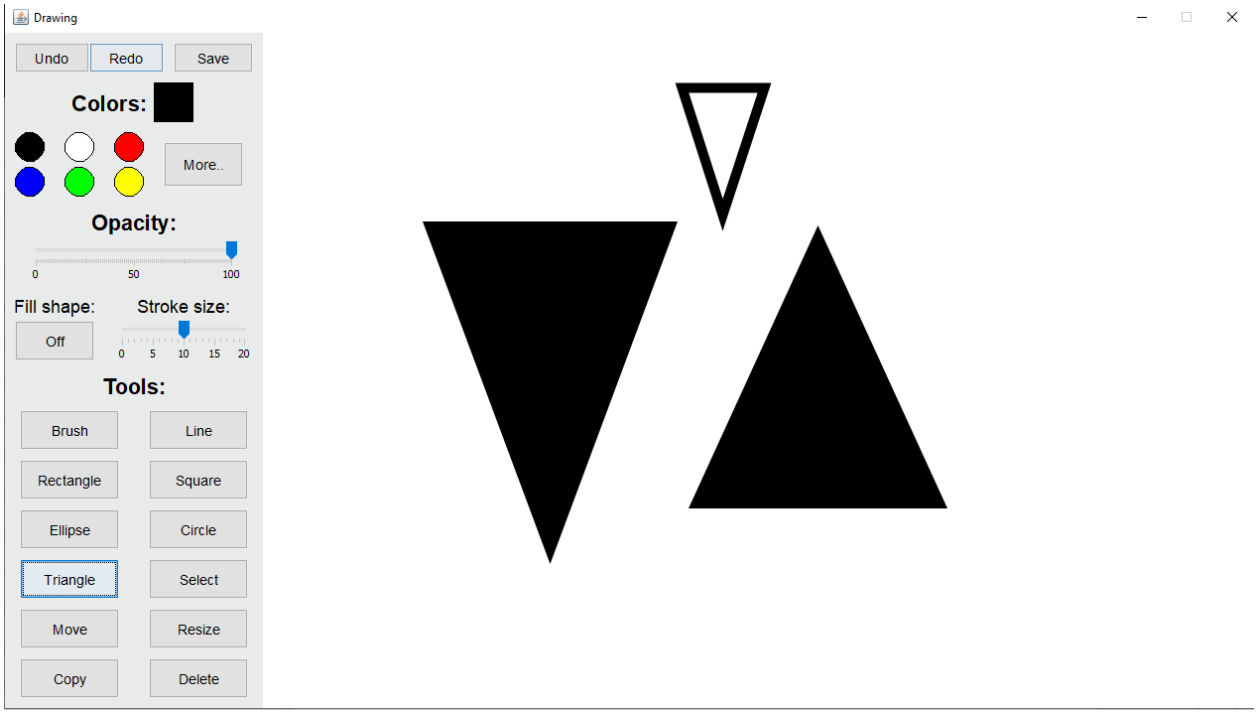
Ellipse:



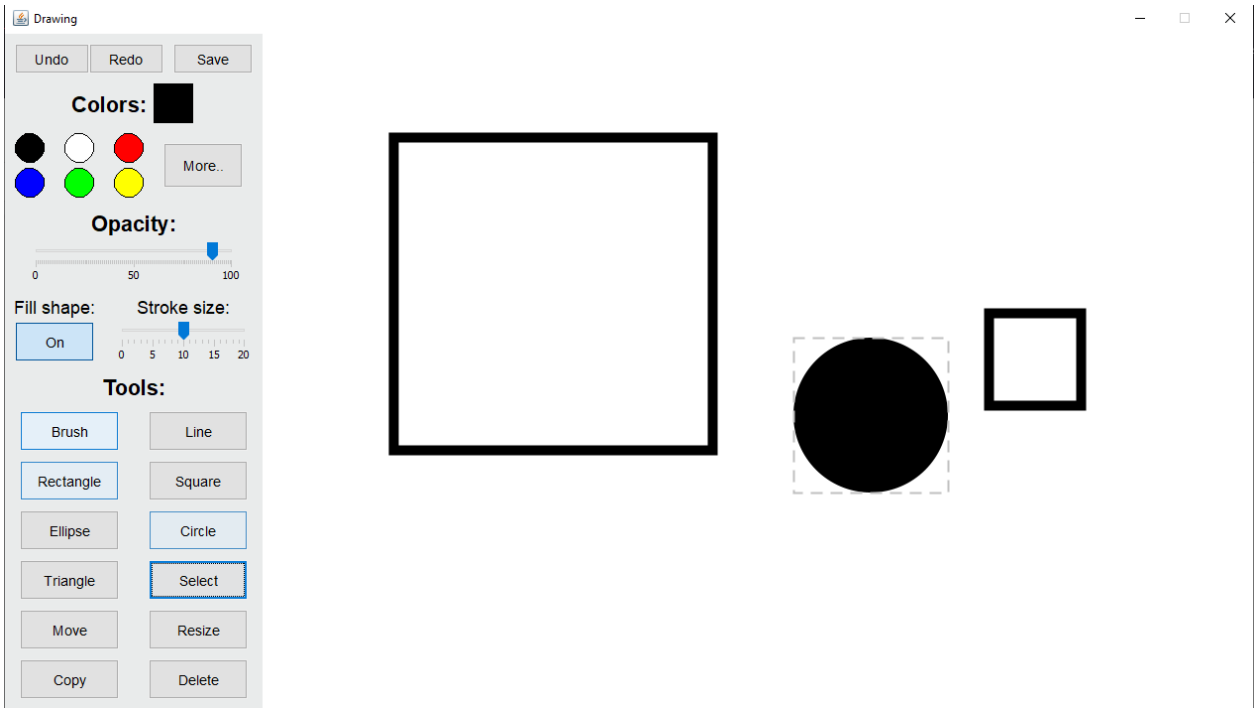
Circle:



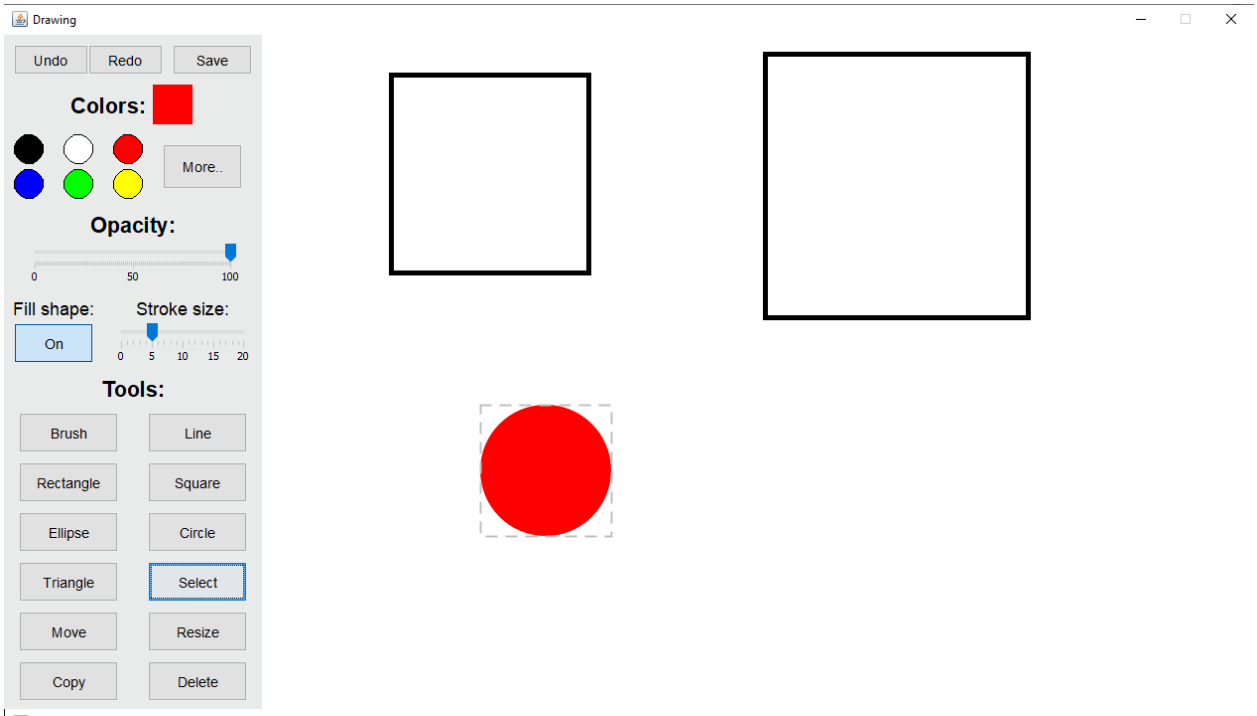
Triangle:

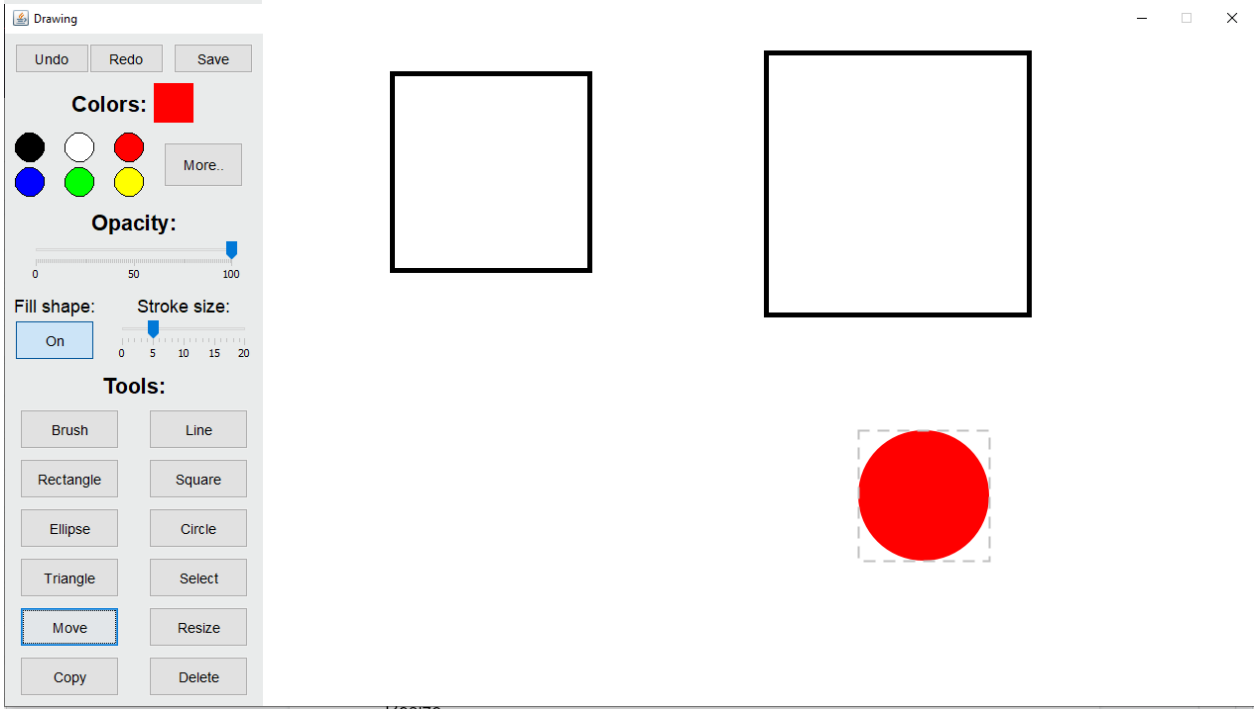


Select:

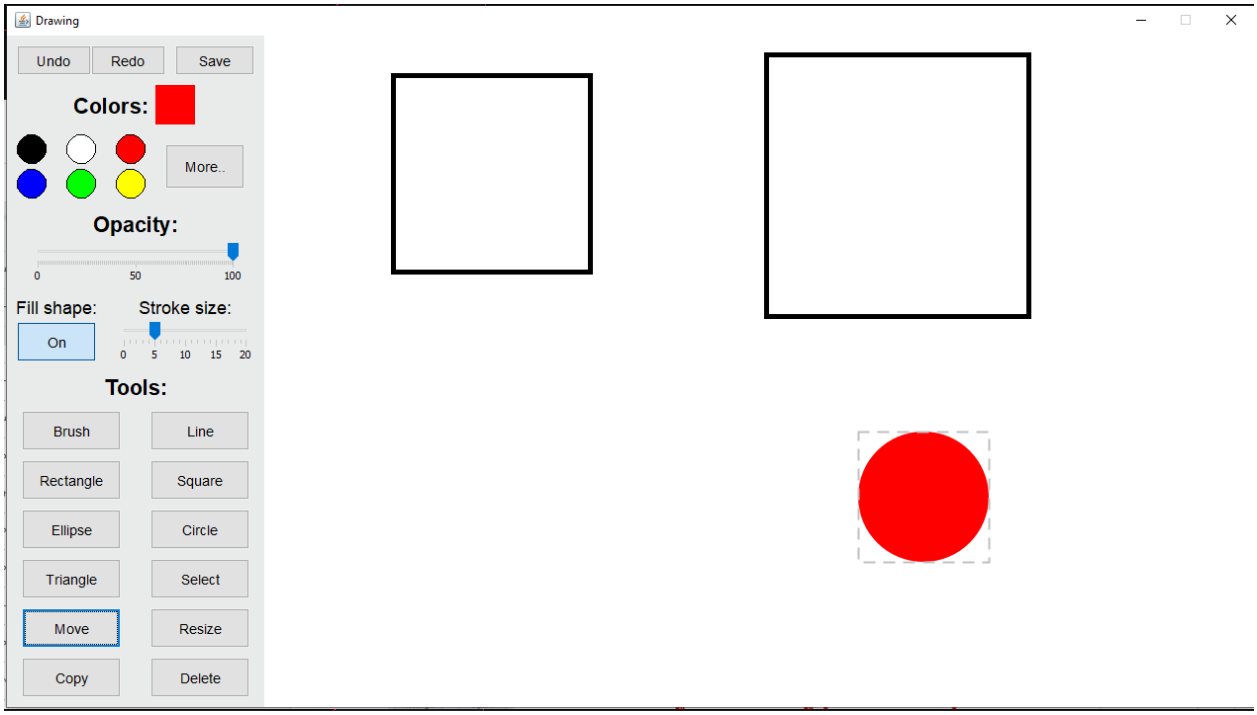


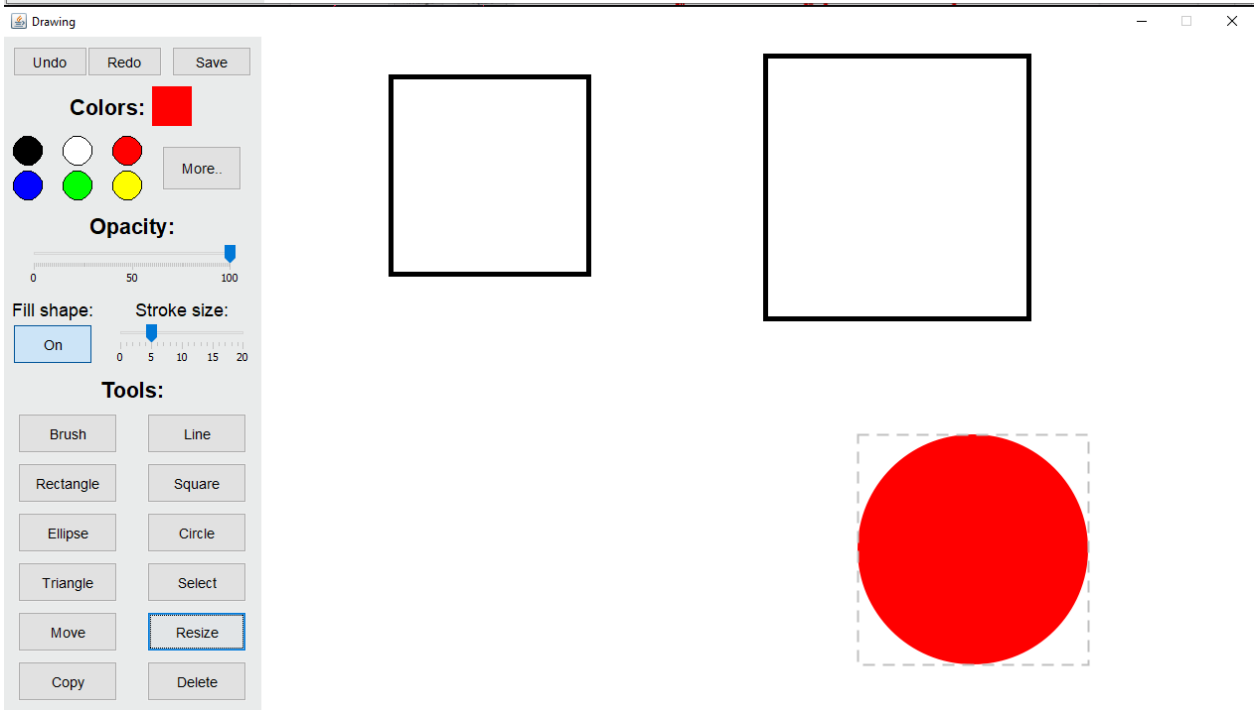
Move:


Drawing
Undo
Redo
Save
Colors:
More..
Opacity:
0
50
100
Fill shape:
Stroke size:
On
0
5
10
15
20
Tools:
Brush
Line
Rectangle
Square
Ellipse
Circle
Triangle
Select
Move
Resize
Copy
Delete

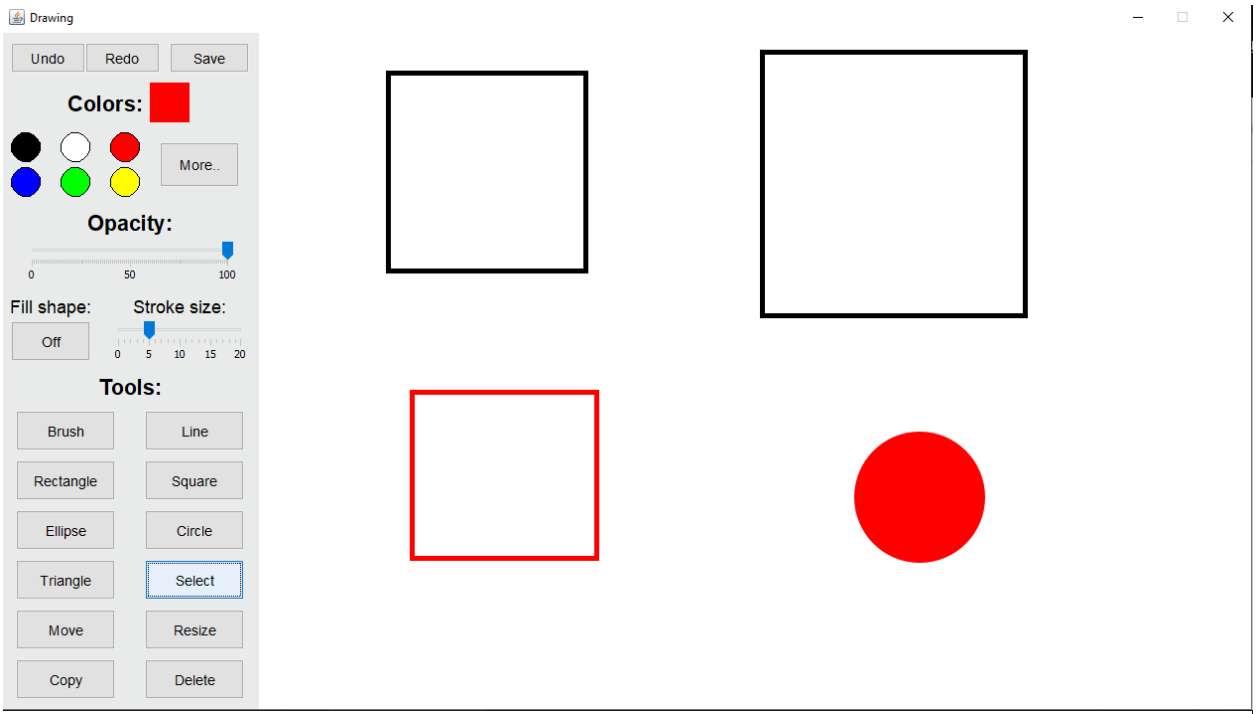

Drawing
Undo
Redo
Save
Colors:
More..
Opacity:
0
50
100
Fill shape:
Stroke size:
On
0
5
10
15
20
Tools:
Brush
Line
Rectangle
Square
Ellipse
Circle
Triangle
Select
Move
Resize
Copy
Delete

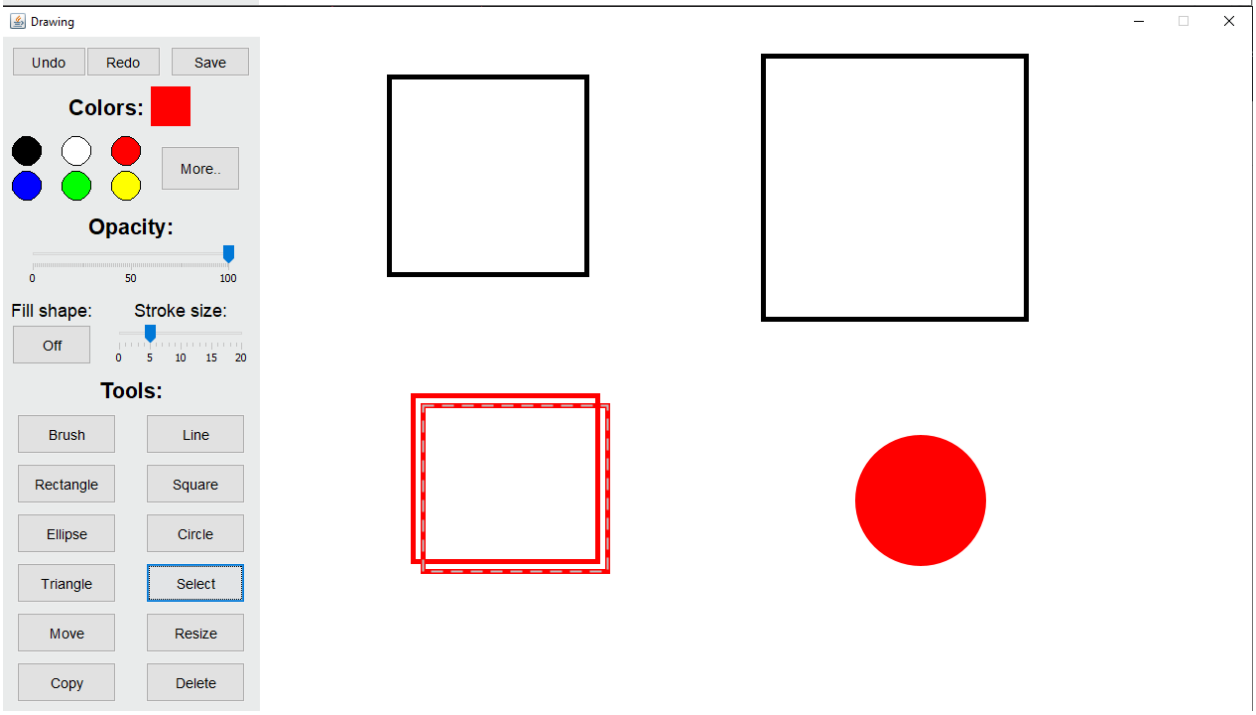
Resize:


Drawing
Undo
Redo
Save
Colors:
More..
Opacity:
0
50
100
Fill shape:
Stroke size:
On
0
5
10
15
20
Tools:
Brush
Line
Rectangle
Square
Ellipse
Circle
Triangle
Select
Move
Resize
Copy
Delete

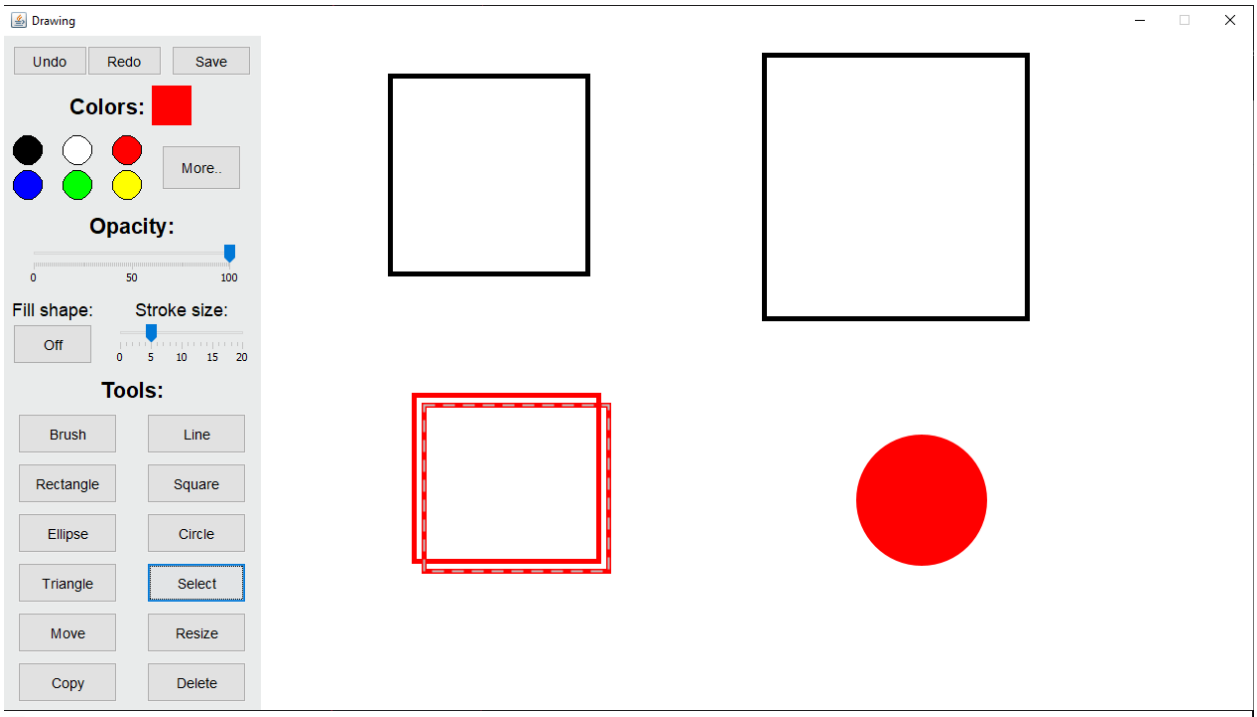

Drawing
Undo
Redo
Save
Colors:
More..
Opacity:
0
50
100
Fill shape:
Stroke size:
On
0
5
10
15
20
Tools:
Brush
Line
Rectangle
Square
Ellipse
Circle
Triangle
Select
Move
Resize
Copy
Delete

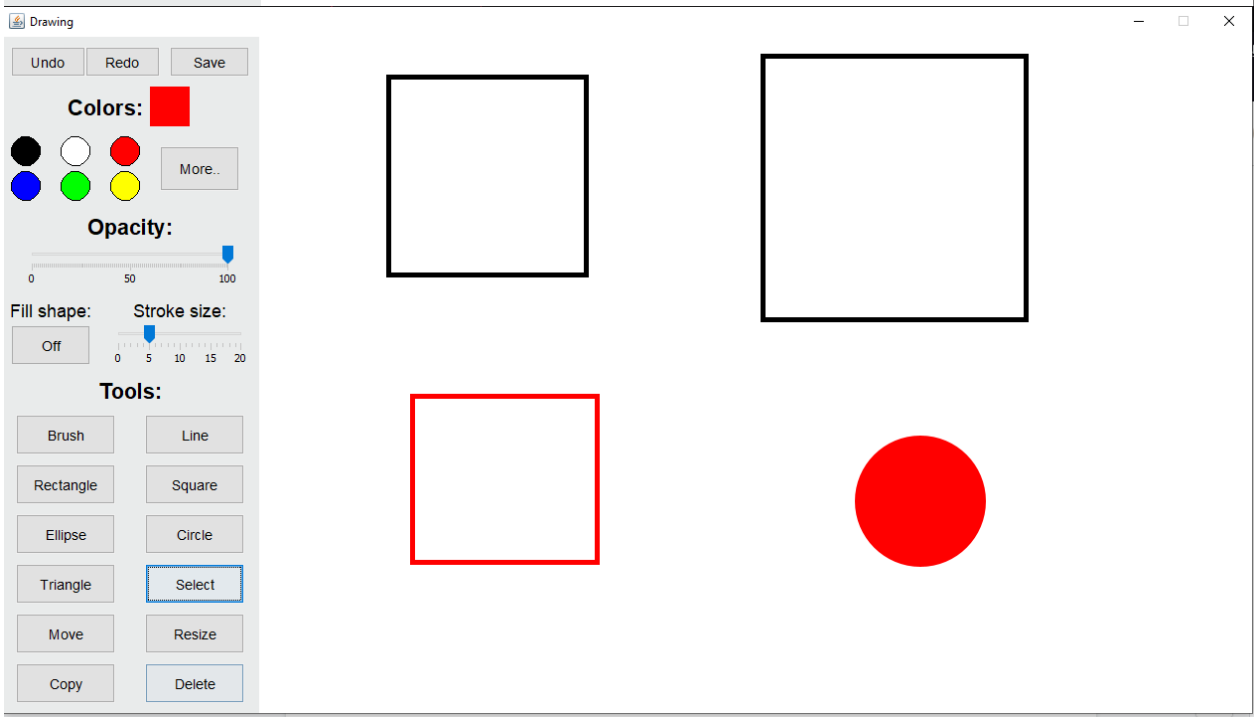
Copy:


Drawing
Undo
Redo
Save
Colors:
More..
Opacity:
0
50
100
Fill shape:
Stroke size:
Off
0
5
10
15
20
Tools:
Brush
Line
Rectangle
Square
Ellipse
Circle
Triangle
Select
Move
Resize
Copy
Delete

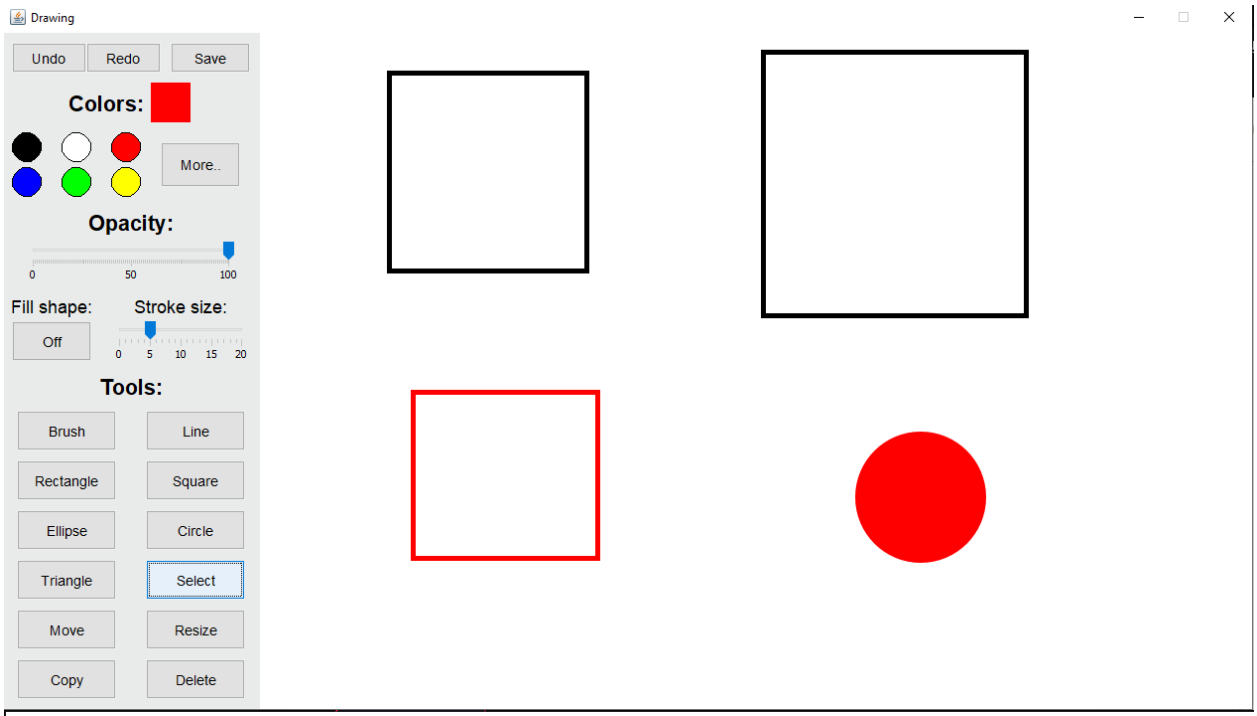

Drawing
Undo
Redo
Save
Colors:
More..
Opacity:
0
50
100
Fill shape:
Stroke size:
Off
0
5
10
15
20
Tools:
Brush
Line
Rectangle
Square
Ellipse
Circle
Triangle
Select
Move
Resize
Copy
Delete

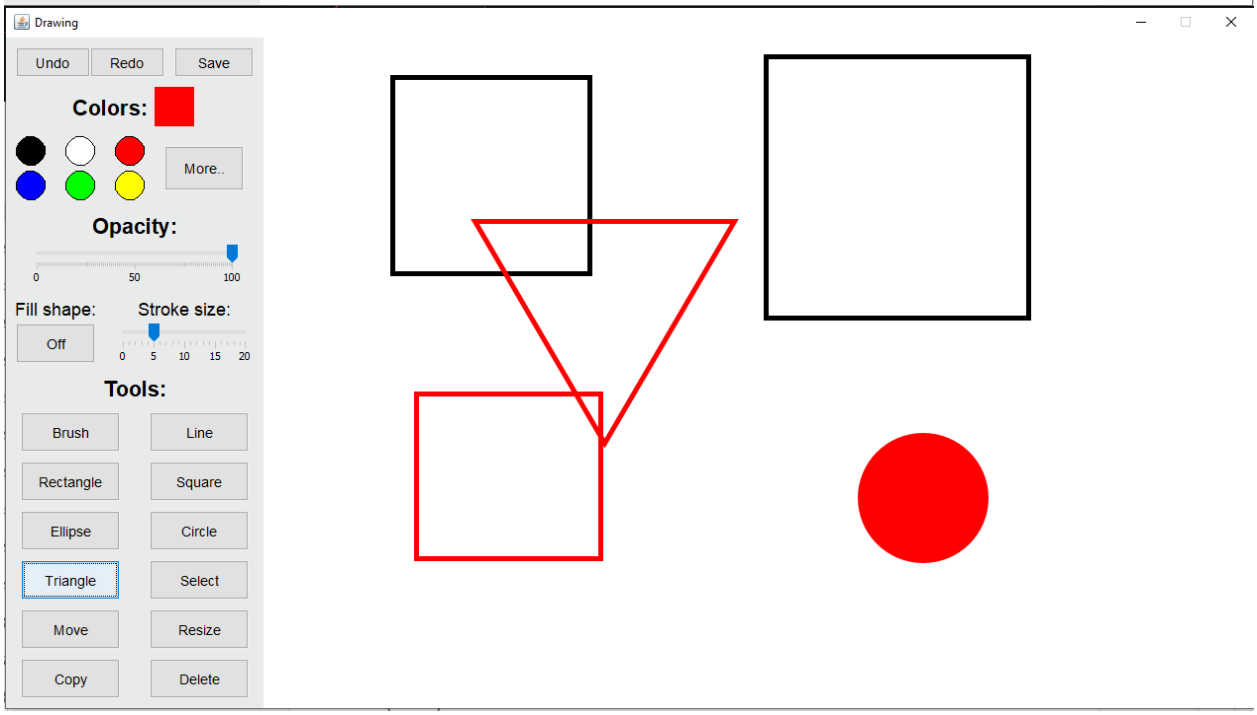
Delete:


Drawing
Undo
Redo
Save
Colors:
More..
Opacity:
0
50
100
Fill shape:
Stroke size:
Off
0
5
10
15
20
Tools:
Brush
Line
Rectangle
Square
Ellipse
Circle
Triangle
Select
Move
Resize
Copy
Delete

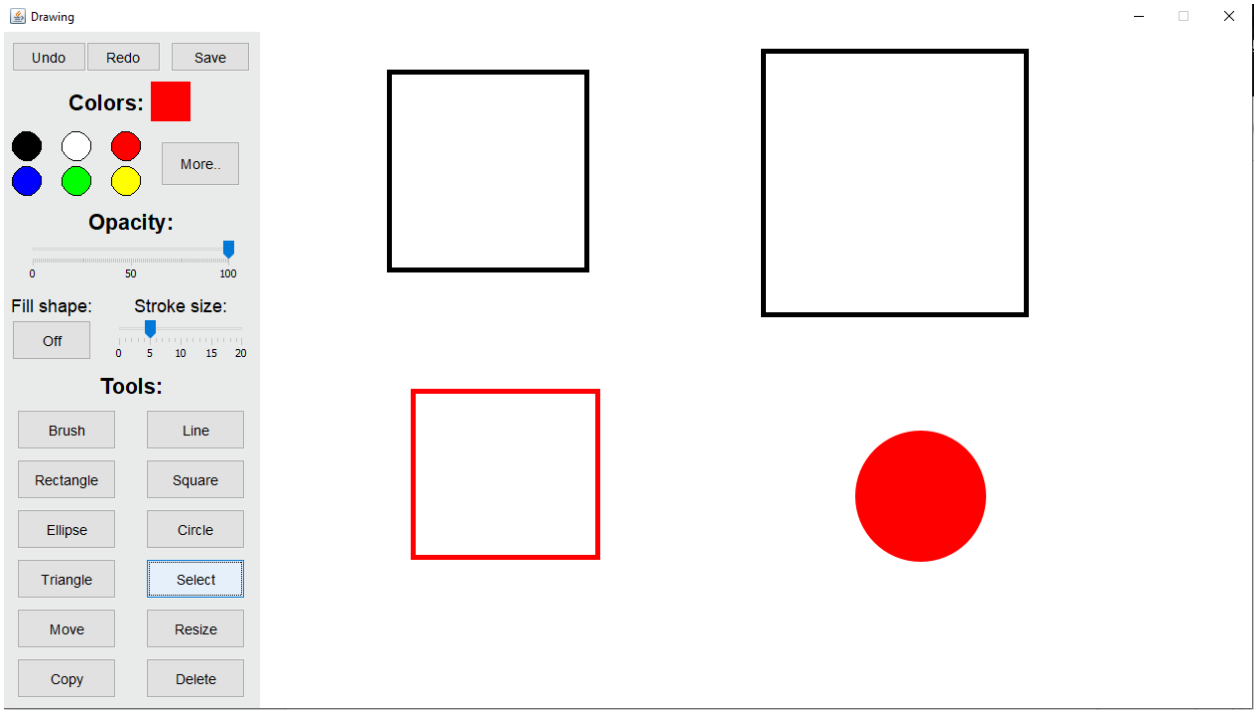

Drawing
Undo
Redo
Save
Colors:
More..
Opacity:
0
50
100
Fill shape:
Stroke size:
Off
0
5
10
15
20
Tools:
Brush
Line
Rectangle
Square
Ellipse
Circle
Triangle
Select
Move
Resize
Copy
Delete

UNDO AND REDO:

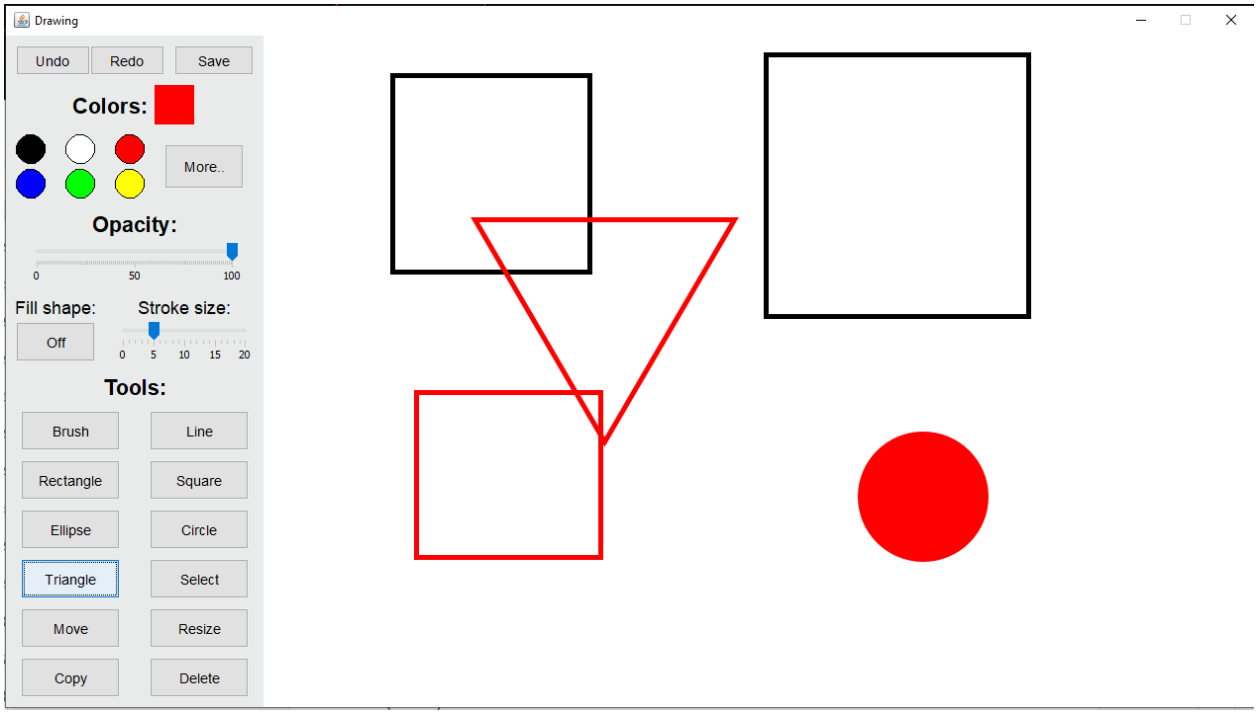




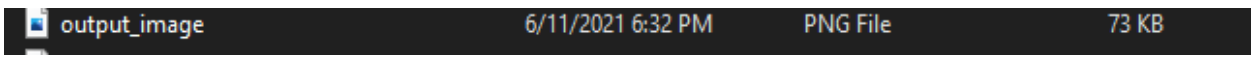
(Undo):



(Redo):

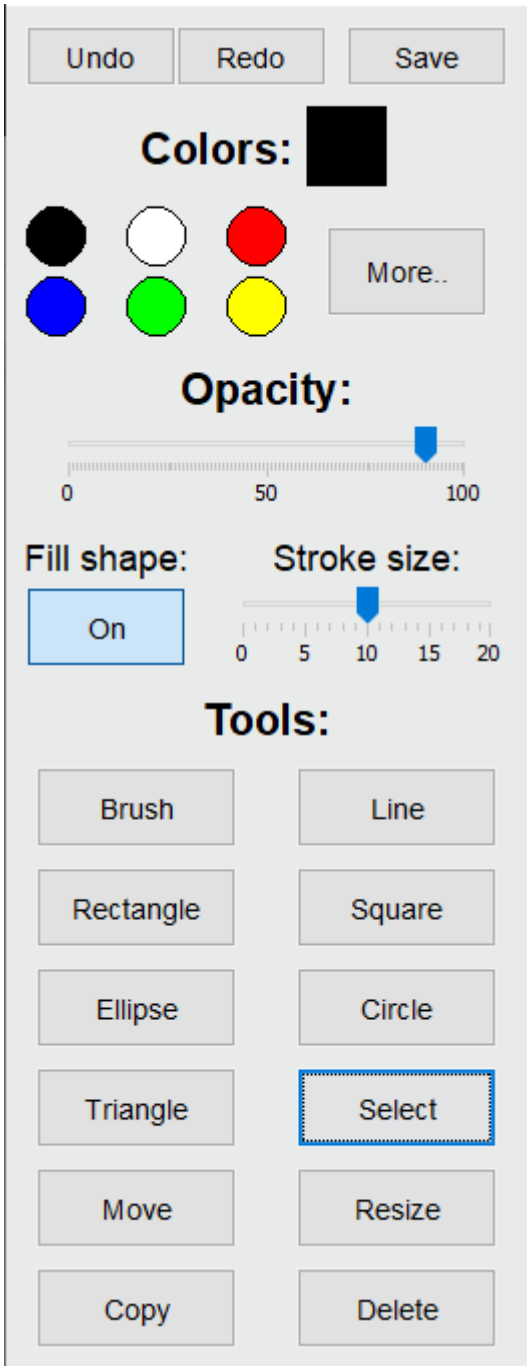


Save:



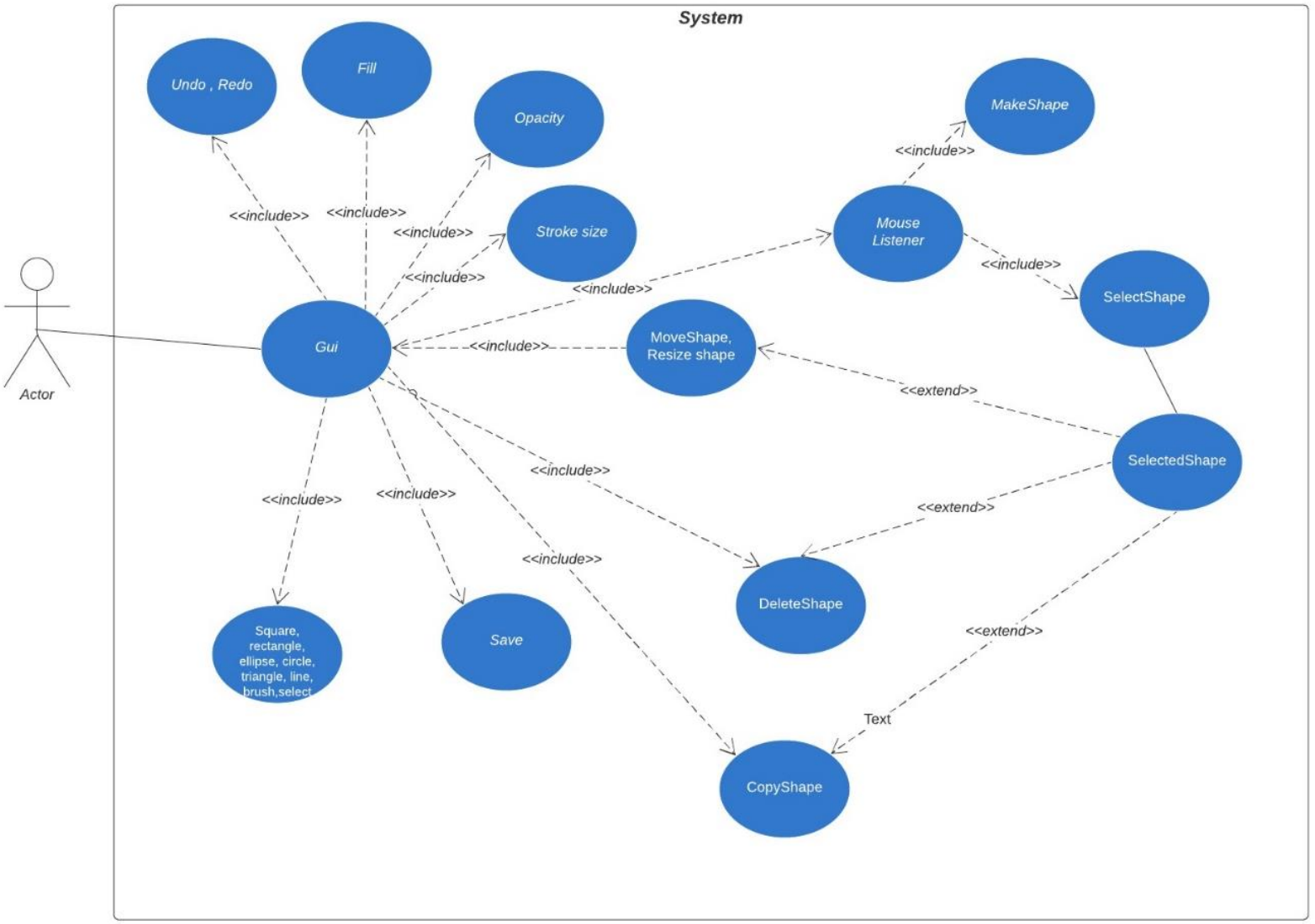
User Guide:

- To draw the desired painting on the app. All of the buttons are on the left side of the drawing window.
- To choose which color to use by clicking on one of the color circles on the left. If the user would like to access more colors click on the “More…” button to select any precise color the user desires.
- To choose the Opacity, drag the Opacity slider to choose from 0 to 100.
- To choose the Stroke size, drag the Stroke size slider to choose from 0 to 100.
- To Fill each shape Drawn, click on the Fill shape button to turn it On.
- To select the desired shape/ brush click on the desired button from the Tools section.
- For freestyle drawing brush click on Brush
- For line segment click on Line
- For rectangle click on Rectangle
- For square click on Square
- For ellipse click on Ellipse
- For triangle click on Triangle
- To select a shape, click on Select.
- To move a shape, click on Select then click on Move to move the shape.
- To resize a shape, click on Select then click on Resize to Resize the shape.
- To Copy a shape, click on Select then click on Copy to Copy the shape.
- To Delete a shape, click on Select then click on Delete to Delete the shape.
- To Undo the last shape added, click on Undo
- To Redo the last shape Undone, click on Redo



UML:

USE-CASE DIAGRAM



CLASS DIAGRAM