

**Programming 2 – OOP and Design Patterns**

**Assignment 2**

**Paint Application**

**Individual Project**

**Muhammad Salah Mahmoud Osman – 41**

**Started 1 Nov. 2018 – finished 11 Nov. 2018**

### **Problem Description:**

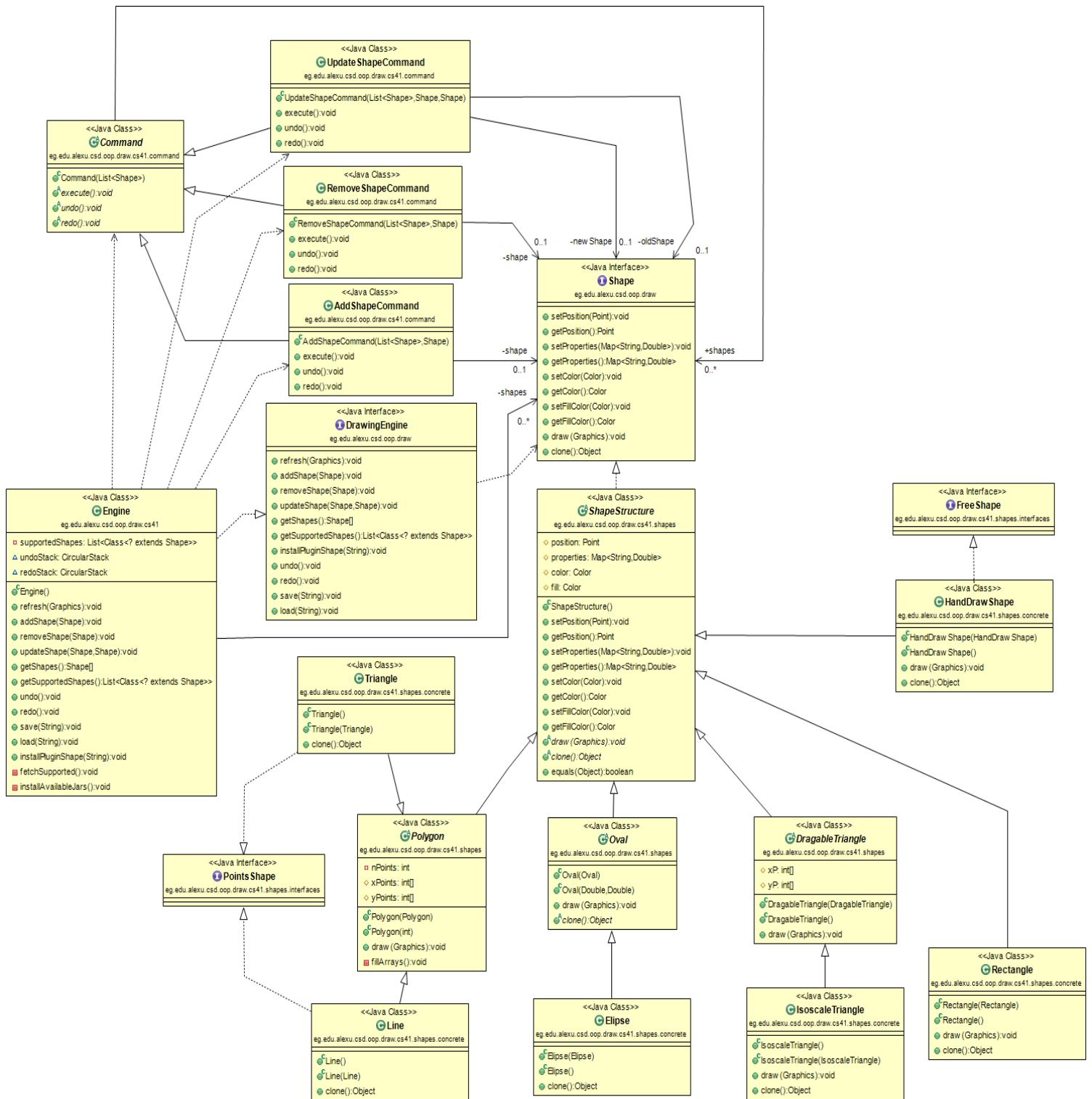
Design an object oriented model for geometric shapes with a simple interactive graphical user interface.

### **Hypothesis and Design Decisions:**

- To provide a general extendable design I assumed that all shapes should be drawn using mouse listener. So I grouped shapes to three categories:
  1. Points shape: which may include lines and triangles.
  2. Free Shape: in which the user defines the shape himself.
  3. Default Shape: which is none of the above groups it has just width and length attributes.
  
- To provide easy undo-redo operations I assumed the following:
  1. The use of the command design pattern is essential to build an easy and extendable undo-redo system.
  2. Using normal stacks wouldn't be enough to have a clean code and fast executions. Using a custom Data Structure for the undo-redo operations will be perfect.
  
- To build a strong save-load system one must parse the documents perfectly. And any missing bracket or tag in the JSON files or XML files would make an error.
- Shapes selection and editing should also be done using mouse listening.

### **UML Diagram and Design Description:**

1. Regarding the geometric system used in the application I generated a UML class diagram to describe my design:








2. In a try to separate the GUI classes from the control classes the project is organized in packages:

- gui: which contains all the user interface files including the listeners, components and tools.
- utilities: contains the helper classes.
- command: contains the commands for the undo redo.
- Shapes: contains the geometric system

All this packages are in a package named cs41 which include the engine and the main class.

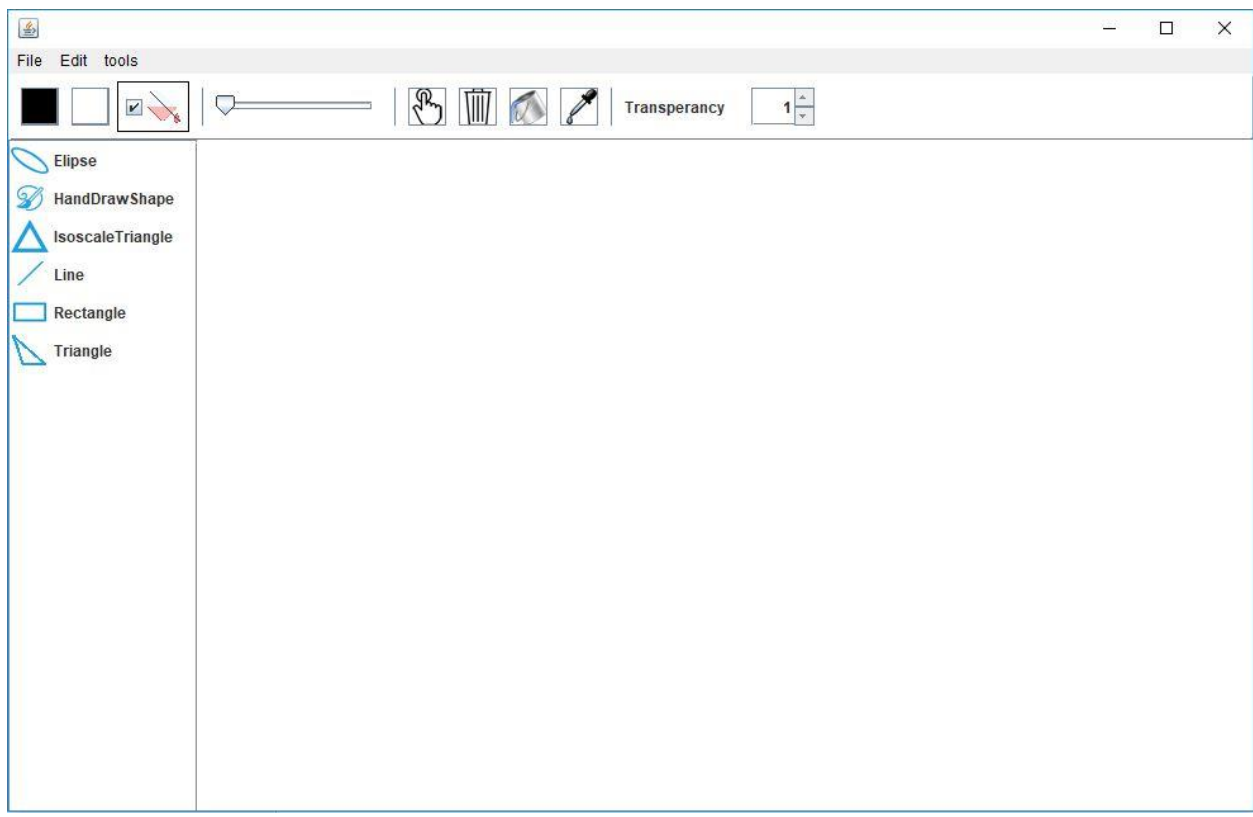
## **User Guide and Tools Description:**

1.  No fill checkbox used when creating shapes with borders only.
2.  Fill tool used to fill a closed shape or the background of the drawing area. (doesn't work with the free shapes).
3.  The Delete button used to delete the selected shapes.
4.  Color Picker is used to set the fill color with any color on the drawing area.
5.  the select tool is used to select, move and resize shapes on the drawing area.
6. Color selection Buttons: on the top left corner of the application clicking the color selecting button will open a color selection dialog to choose a color to close it click the color button again.
7. To undo and redo open the edit menu
8. To save and load open the file menu
9. To add an external shape while the application is running open tools > add external shape and select the jar you want to add.
10. To add an external shape permanently add the jar to the external shapes folder in the application directory.
11. To save your painting as a JPG image open file menu and use export as image.

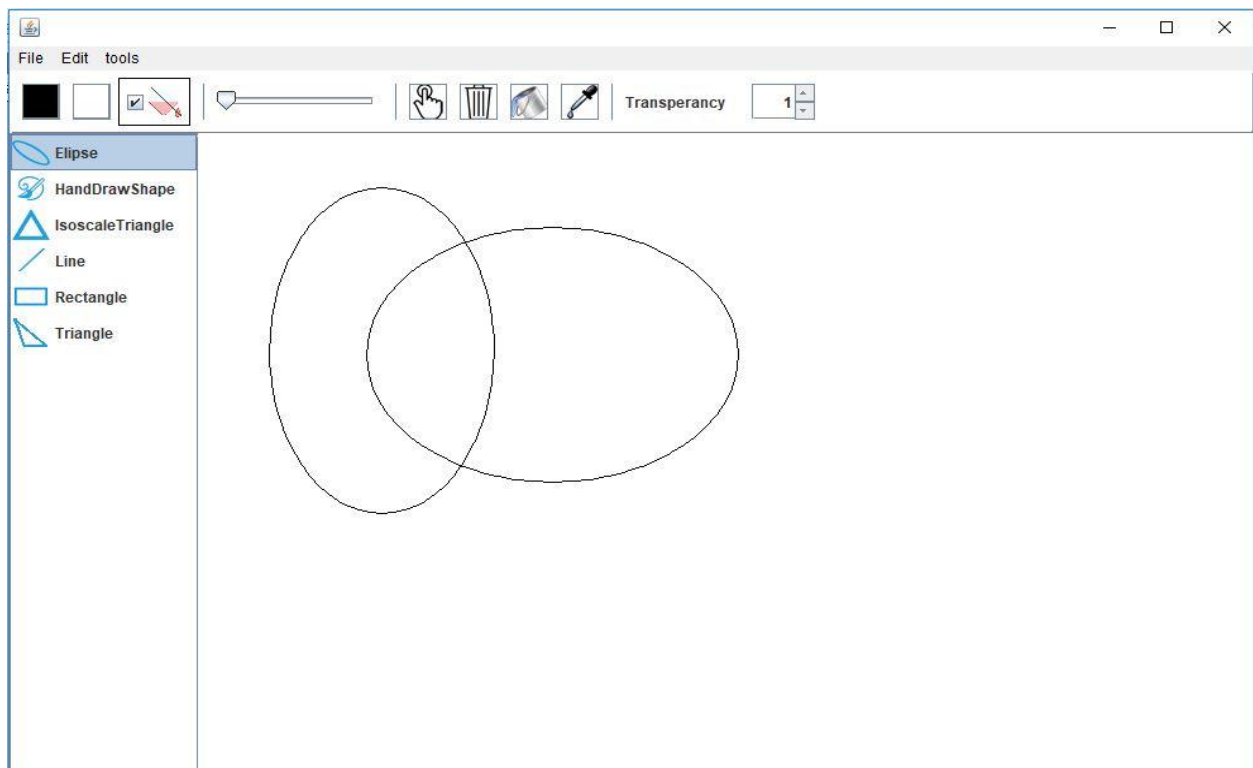
## **Application Features:**

1. Simple and clean user interface. Easy to use.
2. Drawing difficulty is almost zero as you control everything by mouse.
3. Adding external shapes easily.
4. Adjusting the stroke of the drawings. Five different stroke levels.
5. Adjusting the transparency of the layers from zero to 100.
6. Easy selecting using mouse.
7. Easy resizing using the resizing boxes around the shape.
8. Easily move any shape using mouse dragging.
9. Smart and fast color panels.
10. Hand drawing shape which can draw many beautiful shapes.
11. Fill the color of the background.
12. The fill tool; you don't have to re draw the shape to change its color.
13. Custom triangle shape.
14. Save and load in both XML and JSON formats.
15. Export your art as an images with a button click.

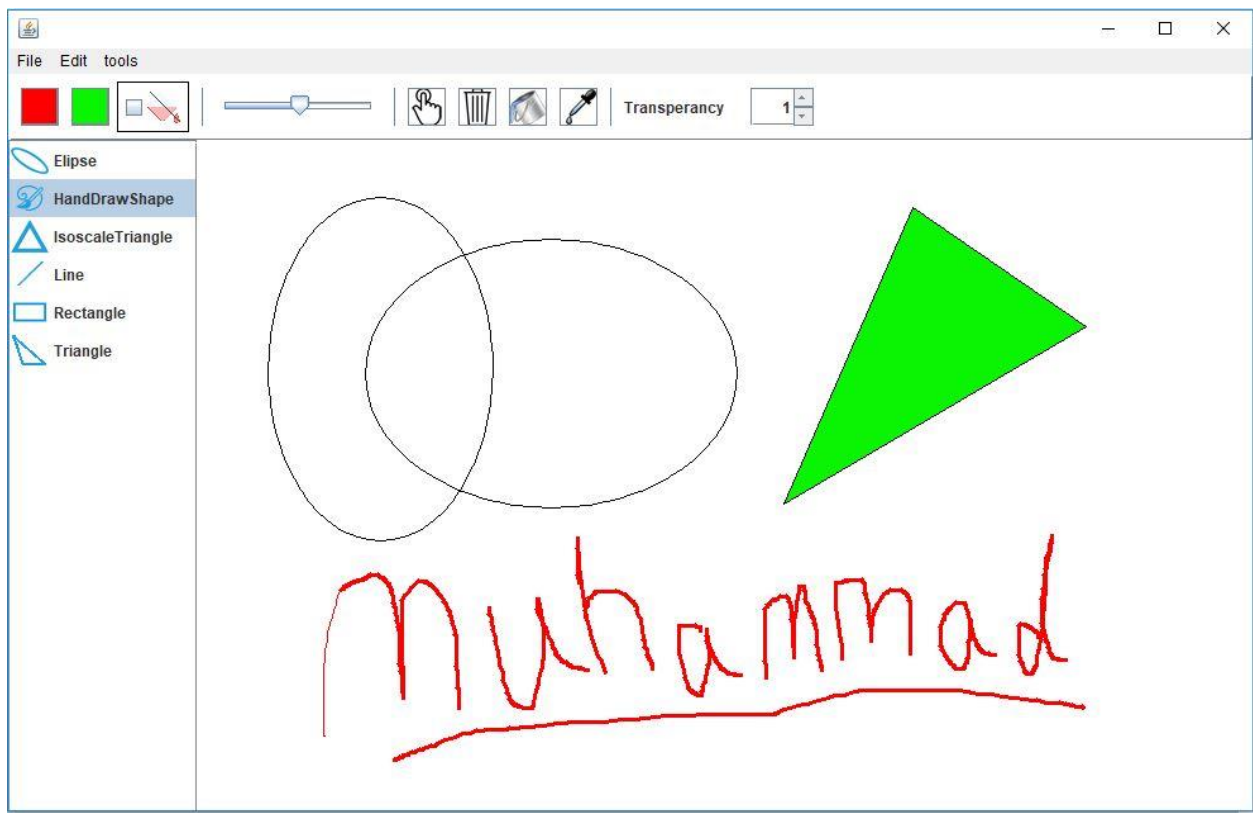
## Application Running:



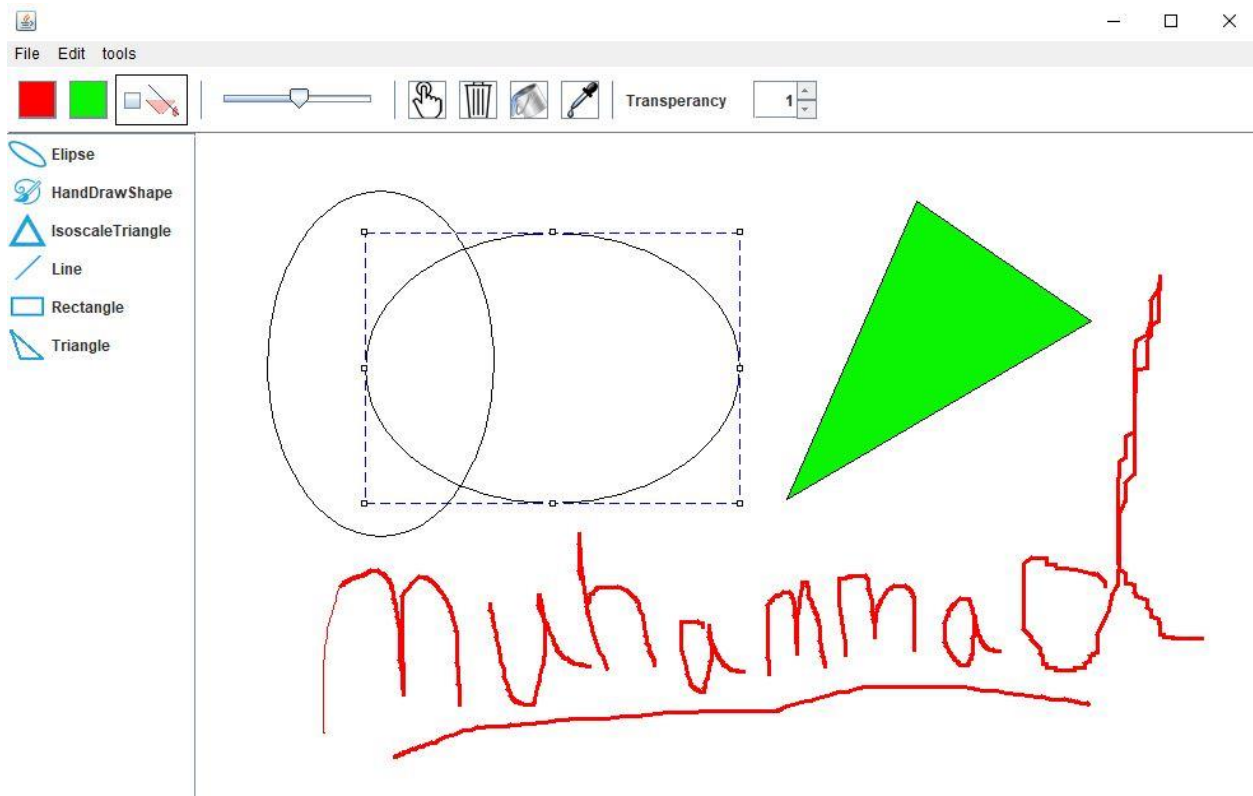
## The app on the startup.



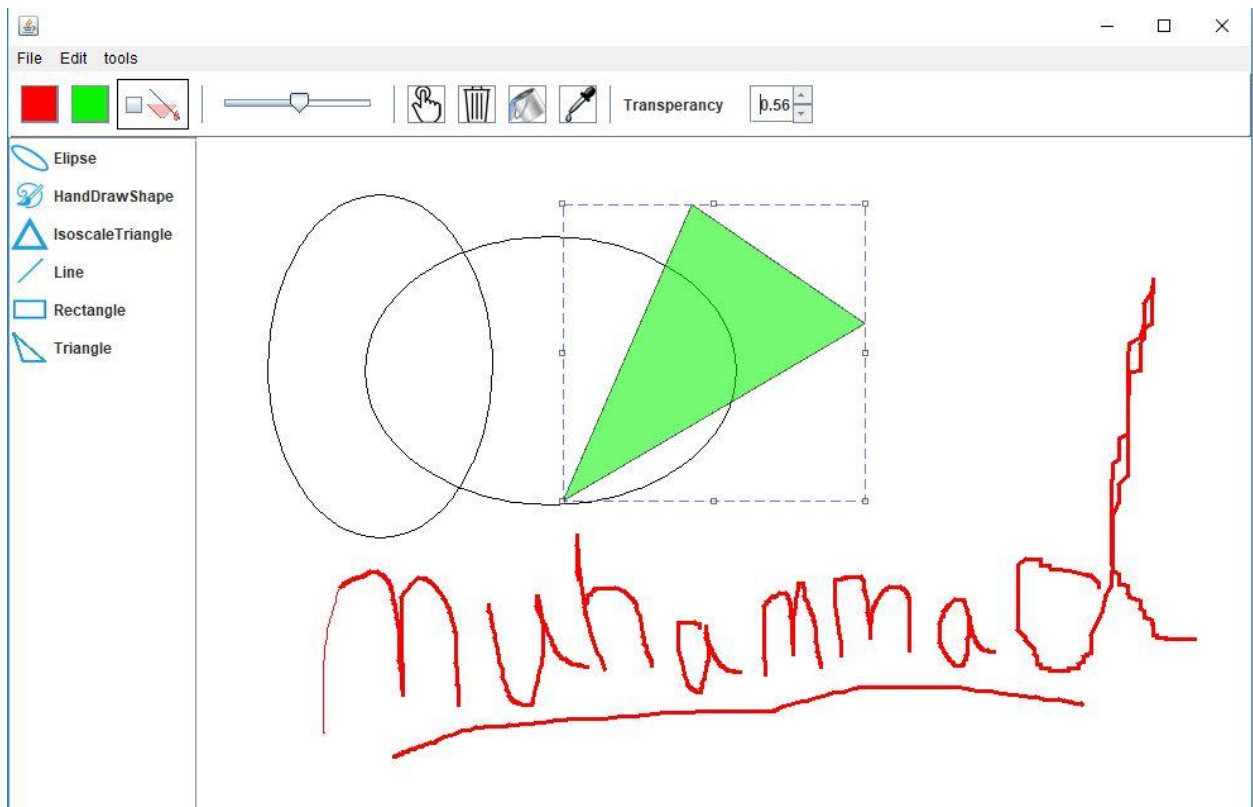
## Drawing two ellipses with noFill.



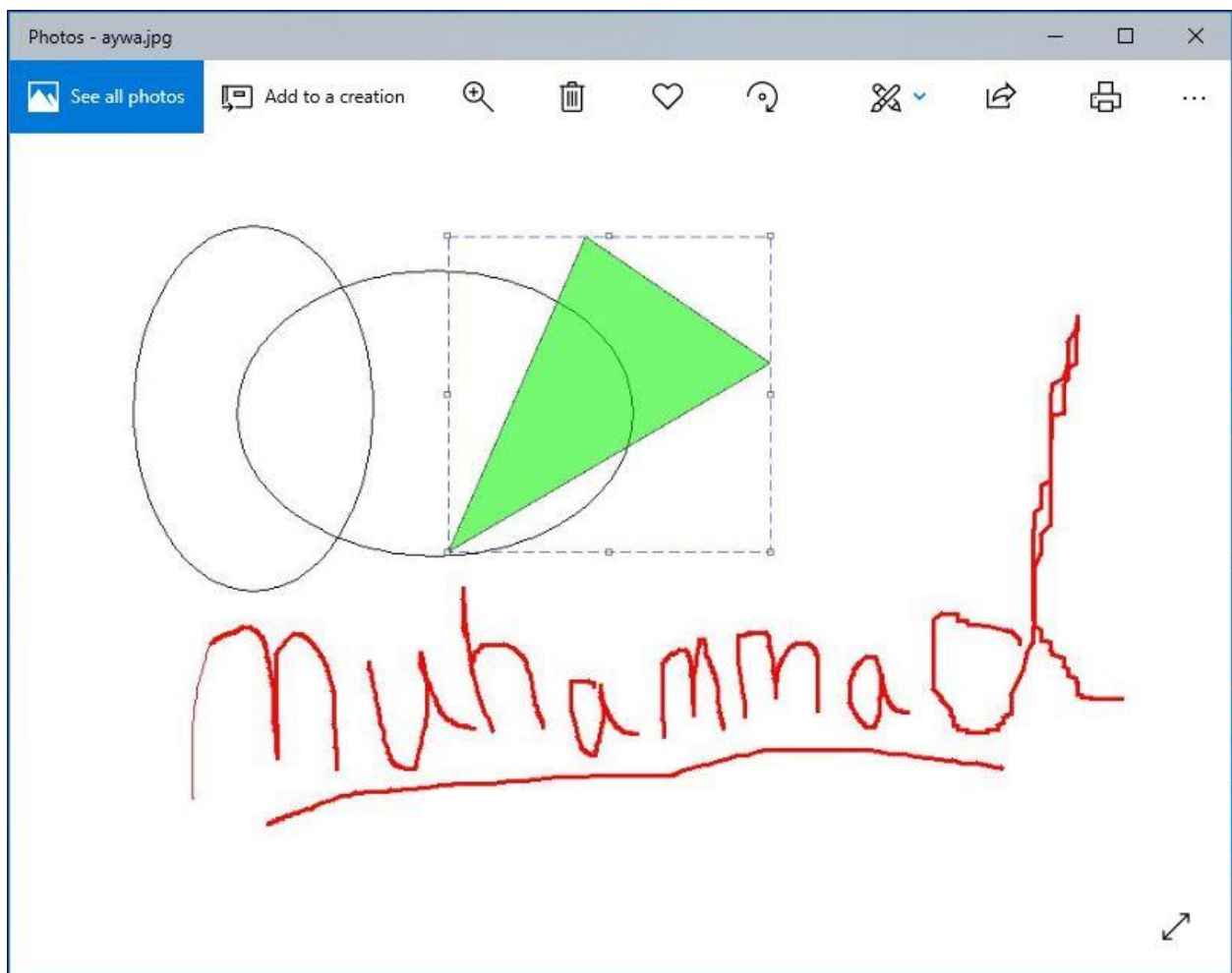
### Drawing several shapes.



### Selecting.



**Changing transparency.**



**Exporting as an image.**