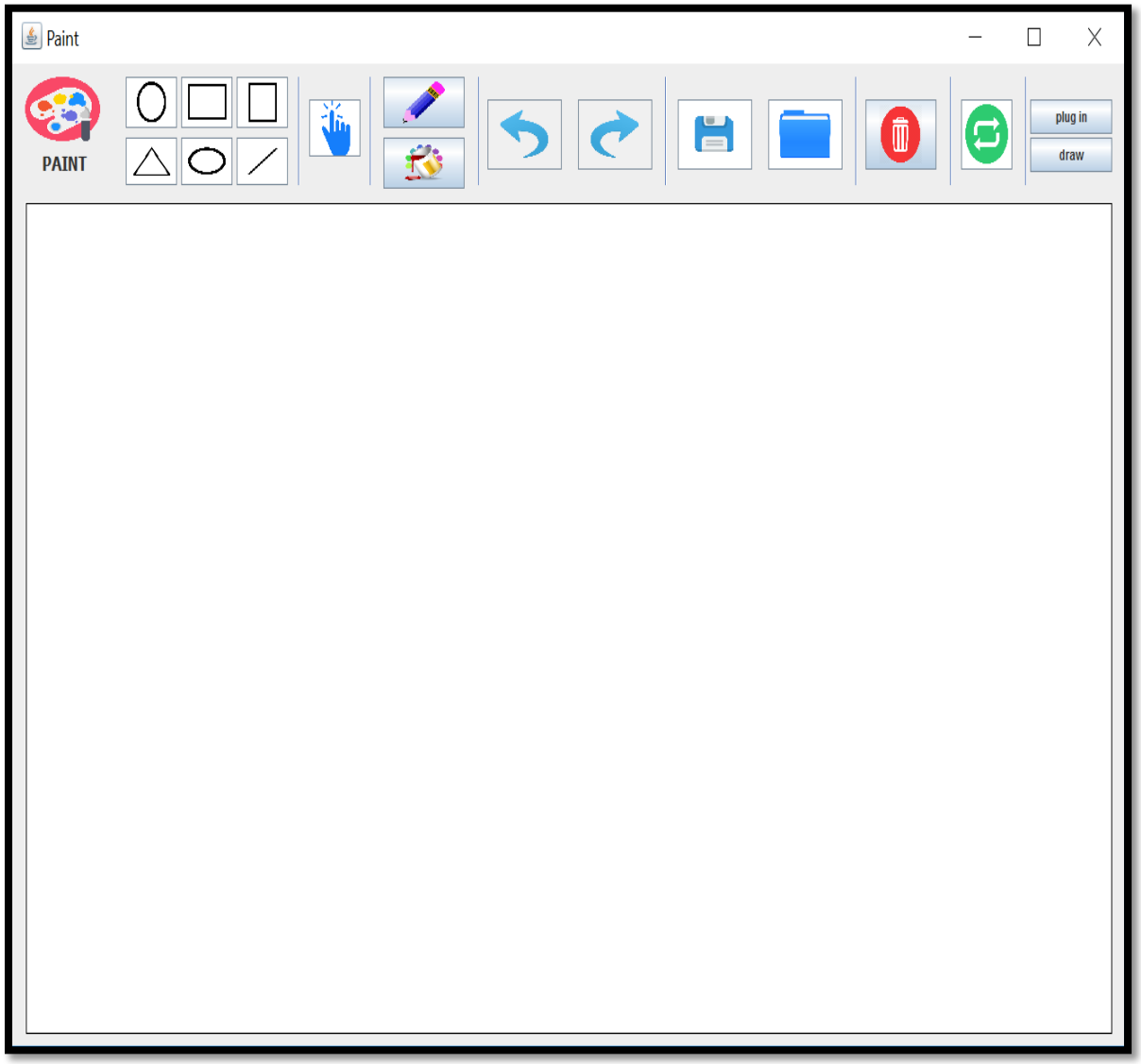
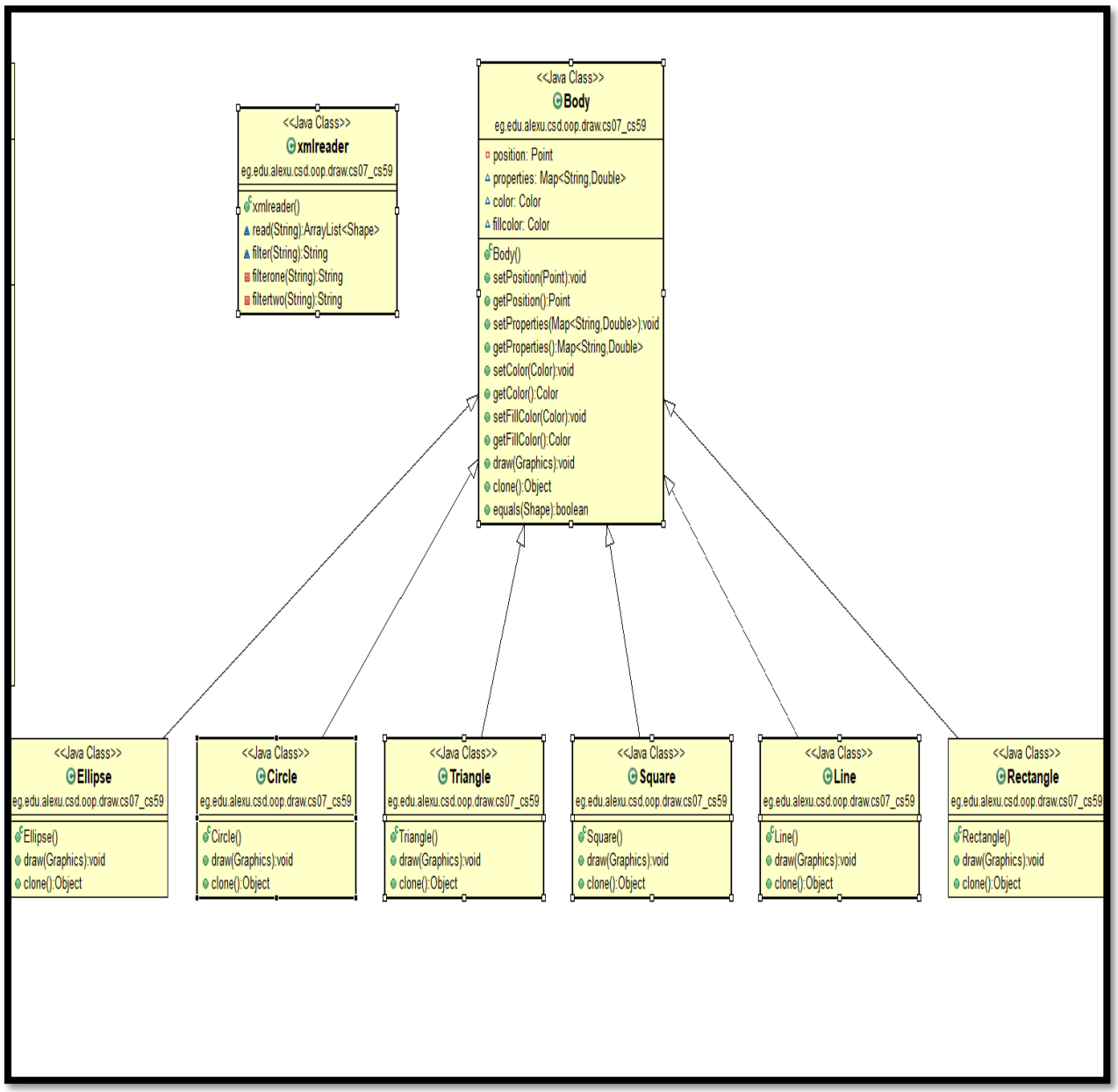
Paint Project

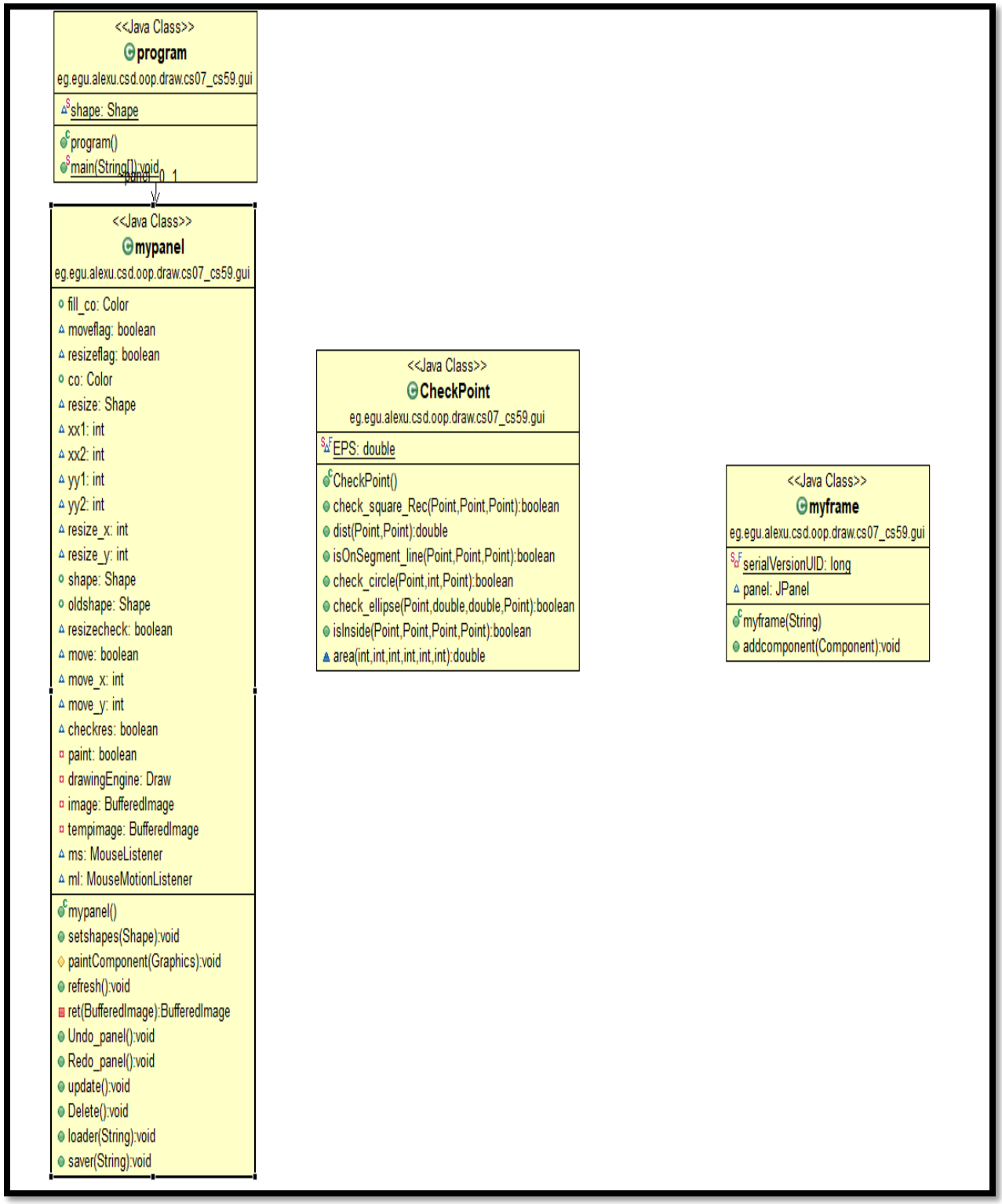
Names :

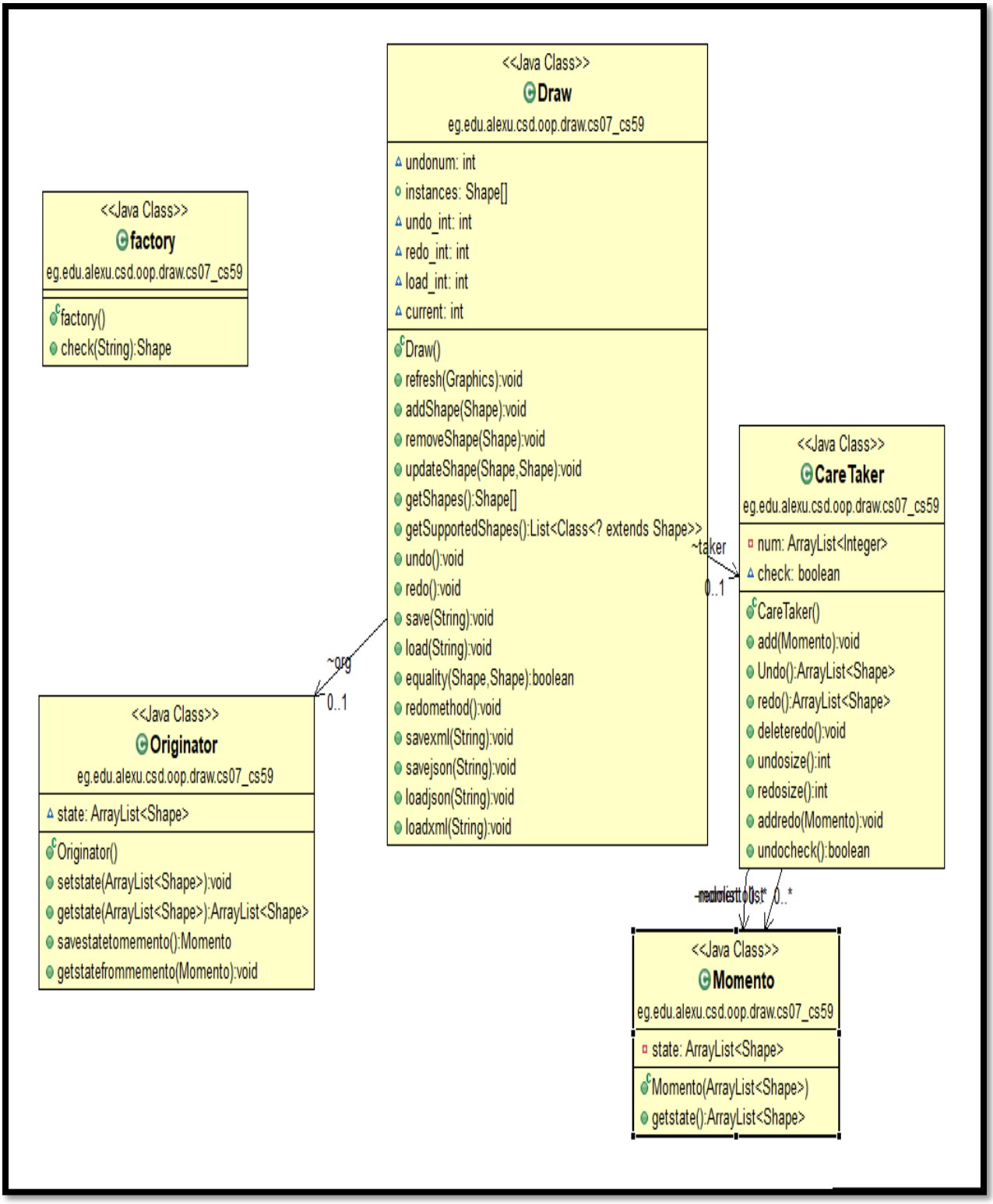
1. Yehia Elsayed Mohamed Mohamed.
2. Ahmed Nabil Mohamed Anwar.



❖ UML Diagrams :




<<Java Class>>
program
eg.egu.alexu.csd.oop.draw.cs07_cs59.gui
shape: Shape
program()
main(String[]):void
panel
0..1
<<Java Class>>
mypanel
eg.egu.alexu.csd.oop.draw.cs07_cs59.gui
fill_co: Color
moveflag: boolean
resizeflag: boolean
co: Color
resize: Shape
xx1: int
xx2: int
yy1: int
yy2: int
resize_x: int
resize_y: int
shape: Shape
oldshape: Shape
resizecheck: boolean
move: boolean
move_x: int
move_y: int
checkres: boolean
paint: boolean
drawingEngine: Draw
image: BufferedImage
tempimage: BufferedImage
ms: MouseListener
ml: MouseMotionListener
mypanel()
setshapes(Shape):void
paintComponent(Graphics):void
refresh():void
ret(BufferedImage):BufferedImage
Undo_panel():void
Redo_panel():void
update():void
Delete():void
loader(String):void
saver(String):void
<<Java Class>>
CheckPoint
eg.egu.alexu.csd.oop.draw.cs07_cs59.gui
EPS: double
CheckPoint()
check_square_Rec(Point,Point,Point):boolean
dist(Point,Point):double
isOnSegment_line(Point,Point,Point):boolean
check_circle(Point,int,Point):boolean
check_ellipse(Point,double,double,Point):boolean
isInside(Point,Point,Point,Point):boolean
area(int,int,int,int,int,int):double
<<Java Class>>
myframe
eg.egu.alexu.csd.oop.draw.cs07_cs59.gui
serialVersionUID: long
panel: JPanel
myframe(String)
addcomponent(Component):void


<<Java Class>>
factory
eg.edu.alexu.csd.oop.draw.cs07_cs59
factory()
check(String):Shape
<<Java Class>>
Draw
eg.edu.alexu.csd.oop.draw.cs07_cs59
undonum: int
instances: Shape[]
undo_int: int
redo_int: int
load_int: int
current: int
Draw()
refresh(Graphics):void
addShape(Shape):void
removeShape(Shape):void
updateShape(Shape,Shape):void
getShapes():Shape[]
getSupportedShapes():List<Class<? extends Shape>>
undo():void
redo():void
save(String):void
load(String):void
equality(Shape,Shape):boolean
redomethod():void
savexml(String):void
savejson(String):void
loadjson(String):void
loadxml(String):void
~taker
0..1
<<Java Class>>
CareTaker
eg.edu.alexu.csd.oop.draw.cs07_cs59
num: ArrayList<Integer>
check: boolean
CareTaker()
add(Momento):void
Undo():ArrayList<Shape>
redo():ArrayList<Shape>
deleteredo():void
undosize():int
redosize():int
addredo(Momento):void
undocheck():boolean
~org
0..1
<<Java Class>>
Originator
eg.edu.alexu.csd.oop.draw.cs07_cs59
state: ArrayList<Shape>
Originator()
setstate(ArrayList<Shape>):void
getstate(ArrayList<Shape>):ArrayList<Shape>
savestatetomemento():Momento
getstatefrommemento(Momento):void
0..*
<<Java Class>>
Momento
eg.edu.alexu.csd.oop.draw.cs07_cs59
state: ArrayList<Shape>
Momento(ArrayList<Shape>)
getstate():ArrayList<Shape>

❖ Program Overview :

Drawing and painting applications are very popular and have a huge user base , they generally offer a big number of features that includes but is not limited to:

Drawing, Coloring, and Resizing. They also include several built in, and possibly extensible set of geometric shapes, and classically, they allow the user to undo or redo any instructions to make the application more usable.

❖ Design description :

1. Body Class

- **Each shape of the six required shapes {Line , Ellipse, Circle, Square, Rectangle, Triangle} has its own class that extends Body class which implements the interface Shape .**
- **Every shape has its own properties that stored in the map of properties , this map differs from shape to another shape.**
- **There are some common methods between shapes that no need to override them in each class like : setColor , setFillColor, get the colors , Set and get position and set and get properties.**

- Each shape has its own draw method so we override it each class and we use Graphics in it to be able to draw the shapes.
- Each shape has its own Clone method to a deep copy of it to be used in other operations like : undo , redo , save , load.

2. Add shape :

- By calling method (addShape) that takes the new shape which will be added as parameter, this method add the new shape into my array of shapes and keep it in the history array which used in undo operation (steps in our design) and clear the redo array.

3. Refresh :

- By calling method (Refresh) that makes loop to our array of shapes and draw all shapes on the panel.

4. Remove shape :

- By calling method (RemoveShape) that takes the specified shape as a parameter, it removes it from my array of shapes and update the history array and clear the redo array .

5. Update shape :

- By calling method (updateShape) that takes the old shape and the new shape as parameters, search for for the old shape in the array of shapes then when finding it update its properties.
- Update the history array by putting the old version of the shape in it.
- Clear the redo array after the operation.

6. Save :

- By calling method (save) tha takes the path which the user wants to save the current workspace in it.
- We check if path contains “.xml” then apply the method of saving in XML file.
- We check if path contains “.json” then apply the method of saving in JSON file.
- We loop on the array of shapes and for each shape we save all of its properties and colors in the file.
- There two ways to save using JSON or XML , We implement both of them manually without using any external libraries and

without using the parser for JSON and without using Dom in case of XML.

- We made an interface for each case XML and JSON , Each interface of them contains the needed method to write in file .
- In case of XML we made methods to control open and close tags and the way we write in them .
- In case of JSON we made methods to control open and close JSON array and separate between the JSON objects in the file.

7. Load :

- by calling method (save) that takes the path which the user wants to load from it.
- We check if path contains ".xml" then apply the method of loading from XML file.
- We check if path contains ".json" then apply the method of loading JSON file.
- We loop on the data of the file and for each shape we use THE FACTORY DESIGN PATTERN to detect its type and the make instance of it then we read its properties from the file .
- We set the properties of each shape and then add it to my array of shapes
- Clear the history and the redo arrays .

- There two ways to load using JSON or XML , We implement both of them manually without using any external libraries and without using the parser for JSON and without using Dom in case of XML.
- We made an interface for each case XML and JSON , Each interface of them contains the needed method to read from file .

8. Get shapes :

- By calling method (getShapes) Return my array of shapes.

9. UNDO :

- we use memento design pattern so when we make a new operation we save the previous one in arraylist and move to the current state .
- you can make undo to the last twenty undo only so we save the last twenty operation only , you can not make more than 20 undo operations.
- when you make undo we return to the previus state in the arraylist and add the current state to the redo.

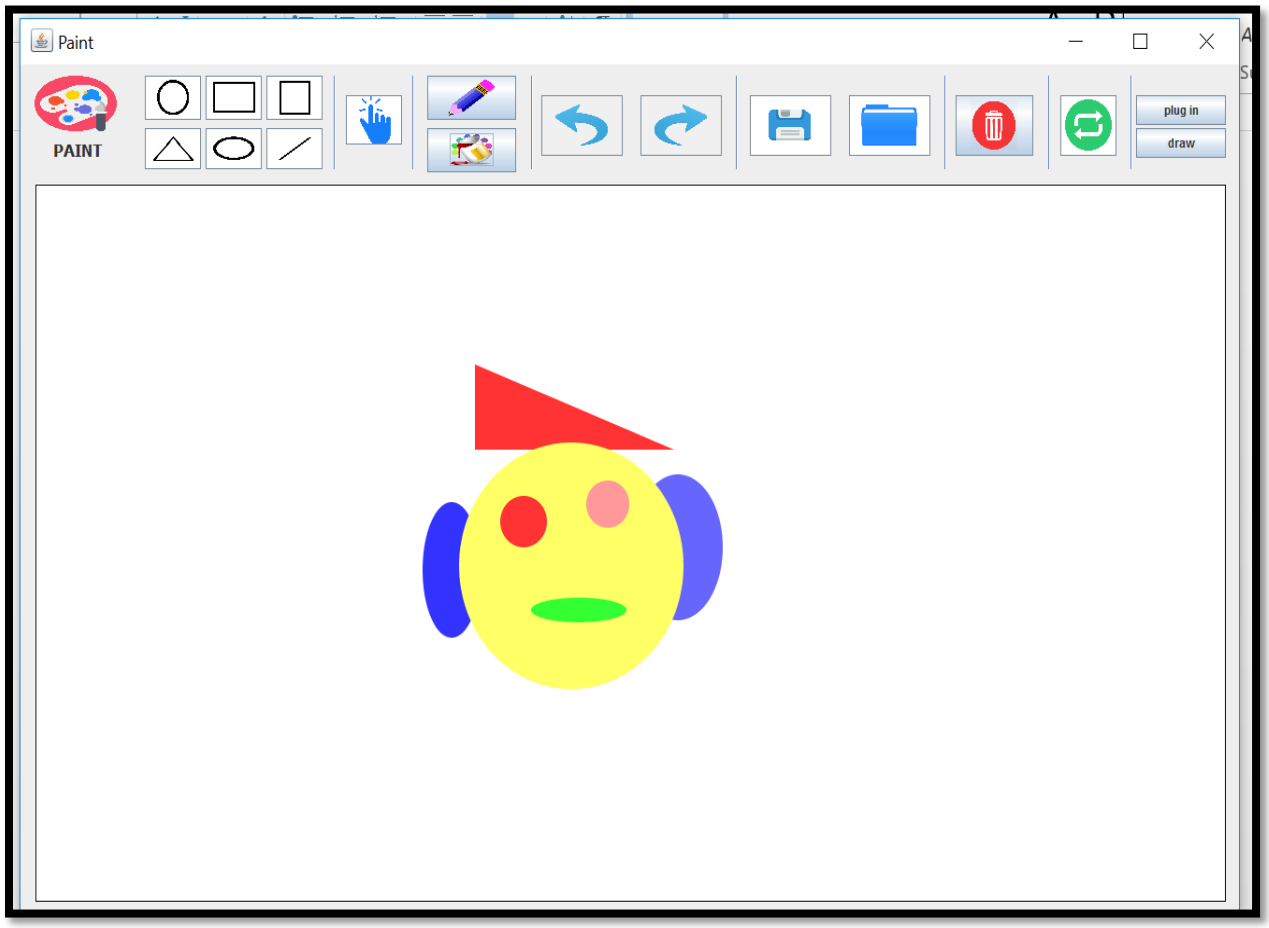
10. REDO:

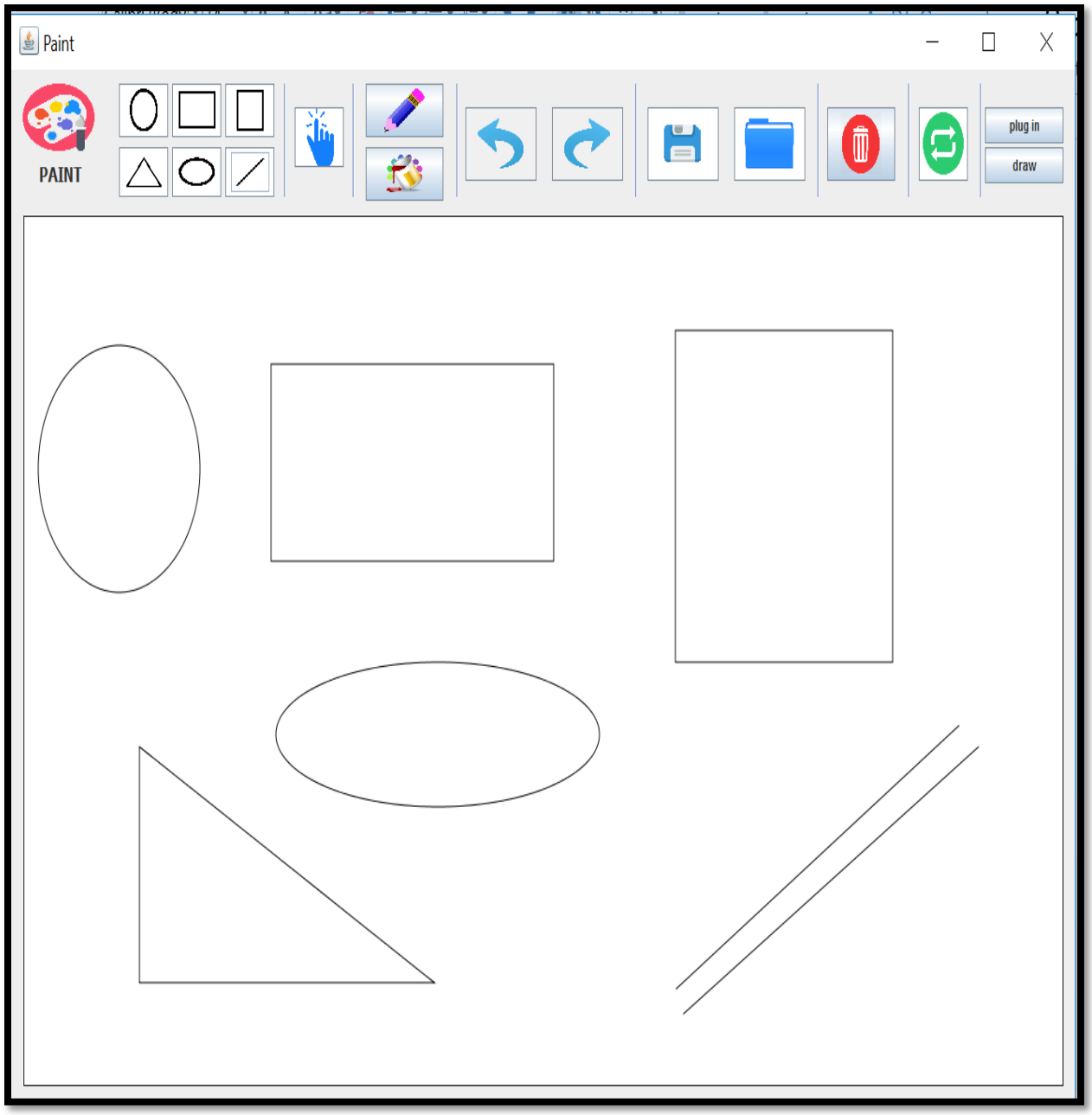
- When you make a we add the current state to the undo list redo and return the last step in the redo arraylist.
- When you make a new operation the redo list will be empty.
- When you make load the undo and redo list will be empty.

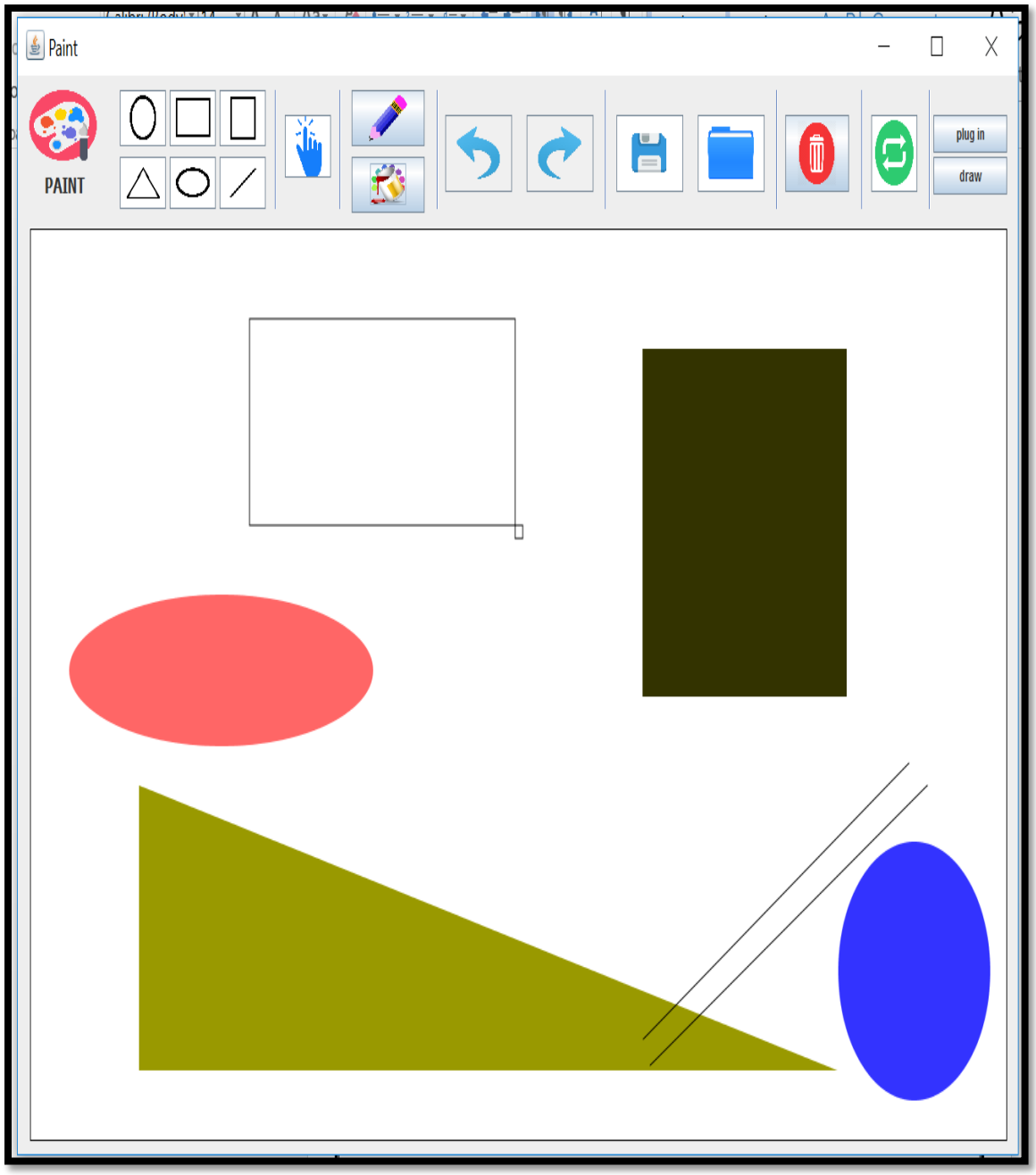
Used Design Patterns :

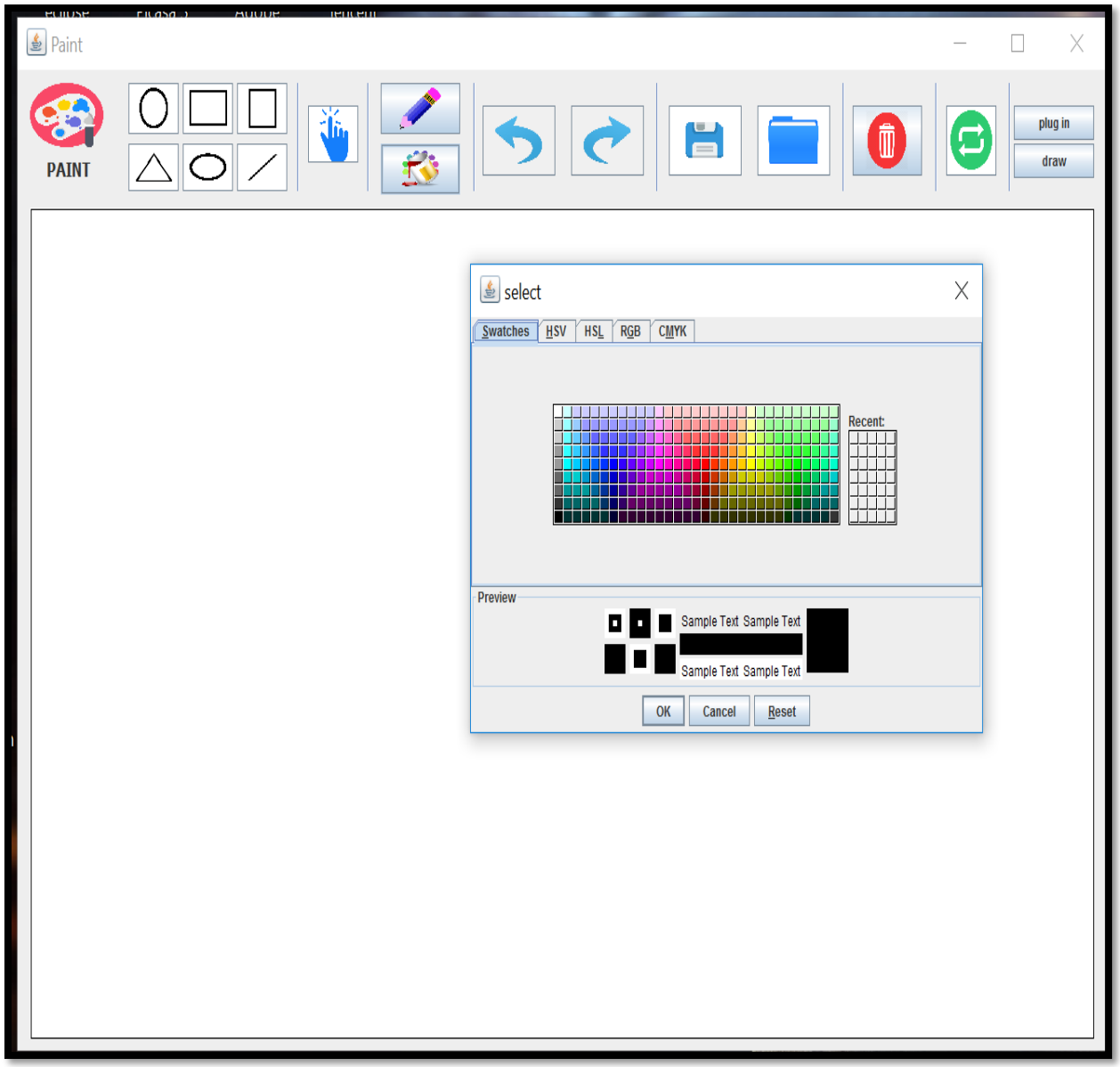
- factory design pattern to make a new shape when we draw and when load a file
- memento design pattern in undo and redo
- delegation design pattern in making some objects in gui and in save and load
- interface design pattern in in json and xml reader and writer

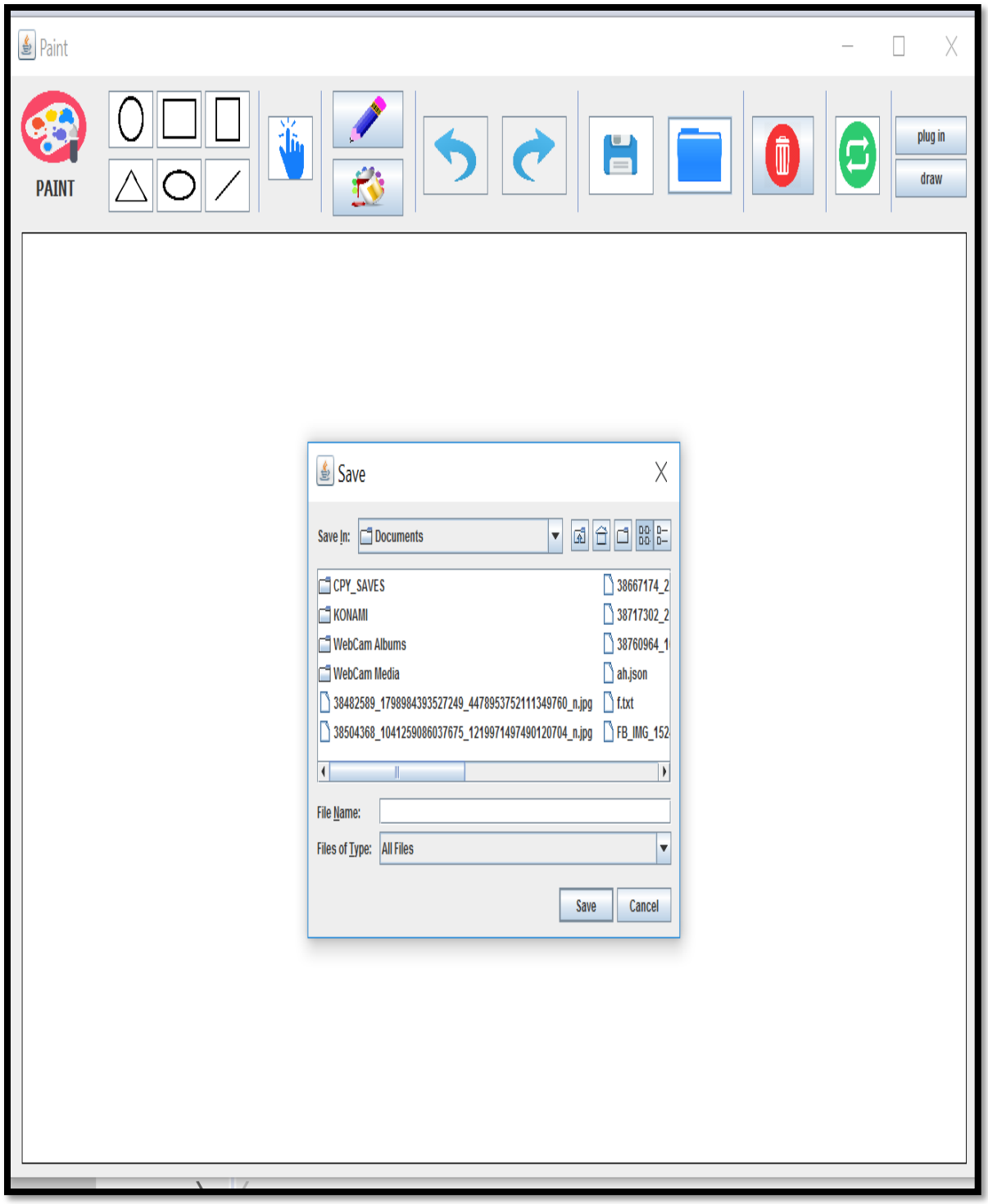
❖ Screenshots of GUI :

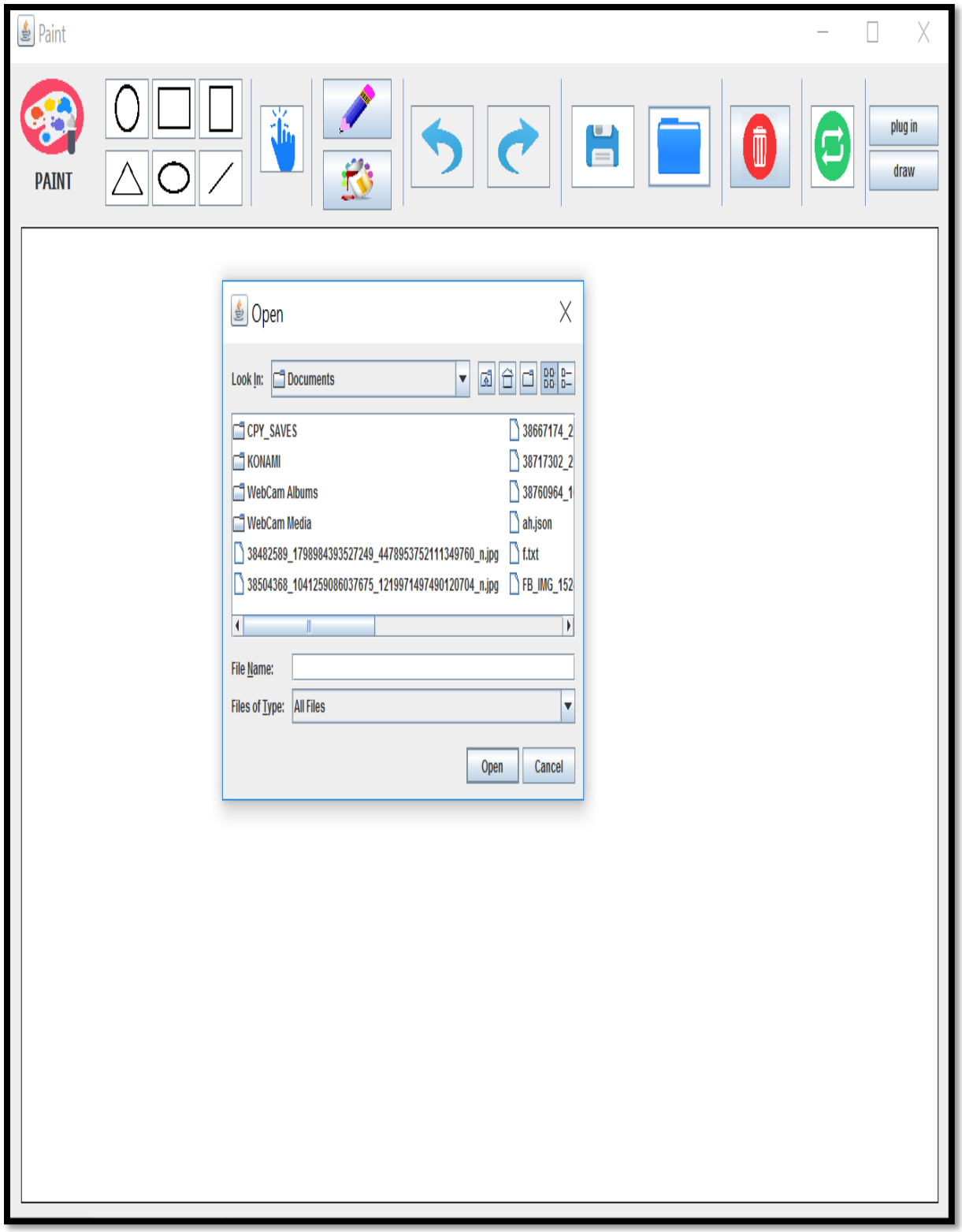



Paint
PAINT
plug in
draw


Paint
PAINT
plug in
draw

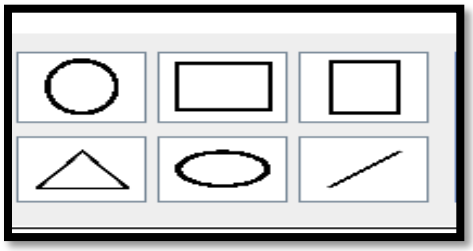

Paint
PAINT
plug in
draw
select
Swatches
HSV
HSL
RGB
CMYK
Recent:
Preview
Sample Text Sample Text
Sample Text Sample Text
OK
Cancel
Reset


Paint
PAINT
plug in
draw
Save
Save In:
Documents
CPY_SAVES
KONAMI
WebCam Albums
WebCam Media
38482589_1798984393527249_4478953752111349760_n.jpg
38504368_1041259086037675_121997149749012070 4_n.jpg
38667174_2
38717302_2
38760964_1
ah.json
f.txt
FB_IMG_152
File Name:
Files of Type:
All Files
Save
Cancel

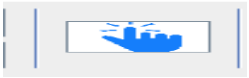

Paint
PAINT
plug in
draw
Open
Look In:
Documents
CPY_SAVES
KONAMI
WebCam Albums
WebCam Media
38482589_1798984393527249_4478953752111349760_n.jpg
38504368_1041259086037675_1219971497490120704_n.jpg
38667174_2
38717302_2
38760964_1
ah.json
f.txt
FB_IMG_152
File Name:
Files of Type:
All Files
Open
Cancel

❖ User manual :

1. **To draw any shape of the required shapes , just click the button which have its photo on it , and you be able to draw the shape using drag and drop , you will be able to draw as many as you want of this shape until you click another action button .**



2. **To select any drawn to edit it click the select hand and then select the shape , after each edit operation if you want to do another one click the select button then the shape and so on .**



3. **To change the color of the drawing border just click on**

The button then chooser the color you want.

4. **To change the fill color of one of the drawn shapes**

Just choose it like mentioned before, the click its

Button and choose the color.

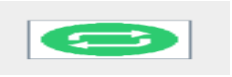
5. **Undo and Redo any operation click undo or redo button**

6. **Save click the save button ,then choose its location and write the name of the file ends with ".json" or ".xml"**

7. **Load click the load button ,then choose the file from its location**

8. **To Delete select shape and press Delete button.**

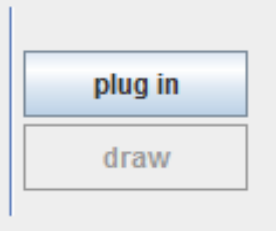
9. **Refresh button.**



10. **Move Shape : select the shape you want to as mentioned before , and keep dragginn on it to the new location where you want to move the shape.**

11. **Resize : Select the shape you want to resize , then a small square will appear on of its border , press this square and drag it to the new size that you want for the selected shape.**

12. **Get plug in for the supported shapes , press its then choose the jar file you want to load .**
13. **To draw the loaded shape press button draw And enter its propeties values .**



❖ Design Decisions :

1. **You can save the drawn SUPPOERTTED SHAPES to file ".json" or ".xml"**

But when you load a file and this file contains SUPPOERTTED SHAPES you must click the PLUG IN button to be able to load this file.

2. **After each operation , if you want to do another one to the same shape or to a new shape you must click the selection button for each operation .**
3. **When save : the user must enter the extension of the file in the field of file name ".json" or ".xml" else you will not be able to save to the file .**