Alexandria University
Faculty of Engineering
Computer and Systems Engineering Dept.
Second Year
Fall 2016

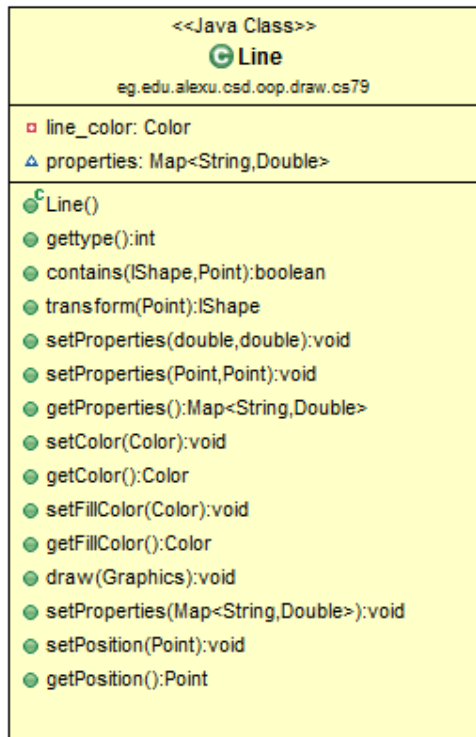# Vector Based Drawing Report

This application is a practical implementation of the program painter who used it frequently in childhood.

This application can be divided part design and writing code, but it is better to split it in several stages consisting of:
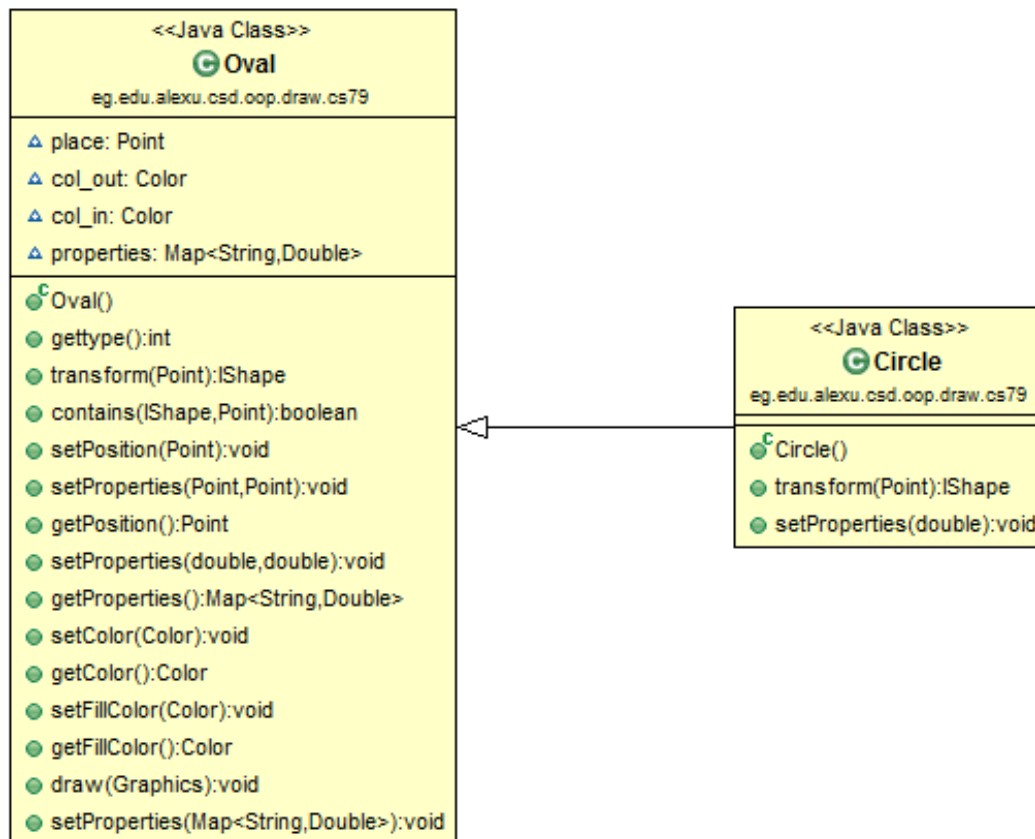
First: Geometric Shapes Data Model:

- ❖ Geometric shapes belong to different groups (ex: Elliptical Shapes, Polygons)
  For details: Line Segment, Circle, Ellipse, Triangle, Rectangle and Square.
- ❖ Members of these different groups are related to each other in the sense that they share common properties like (Circle derived from the Oval) & (Square derived from Rectangle) & (Right Triangle derived from Triangle).
- ❖ In order to be able to implement an efficient and object oriented drawing application, it is essential to design a model that takes these relations into consideration.
- ❖ We can express the code of every shape in the program using UML language.
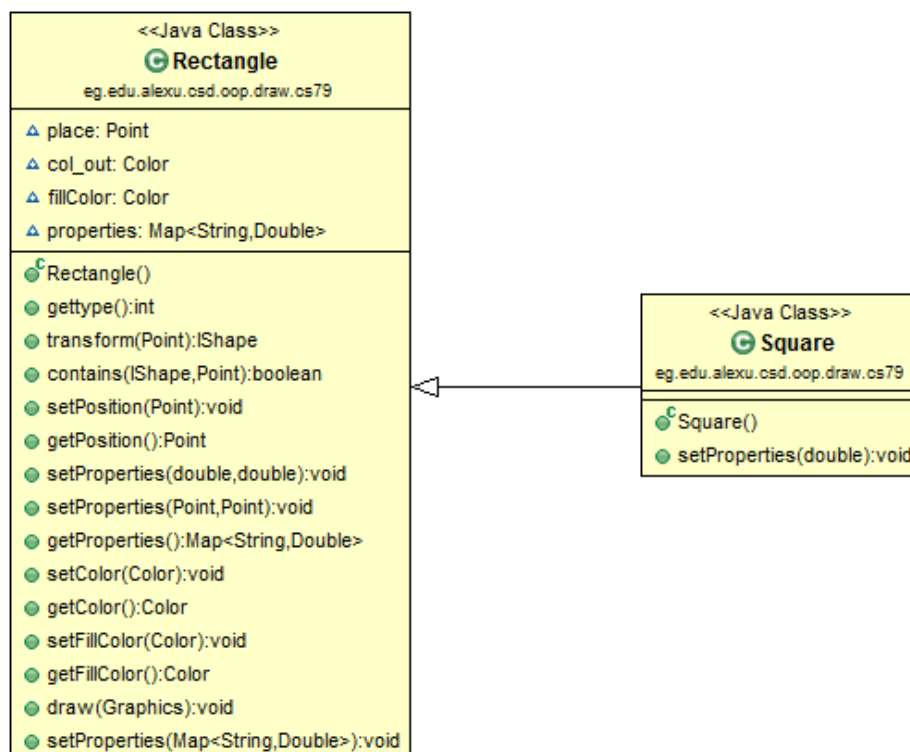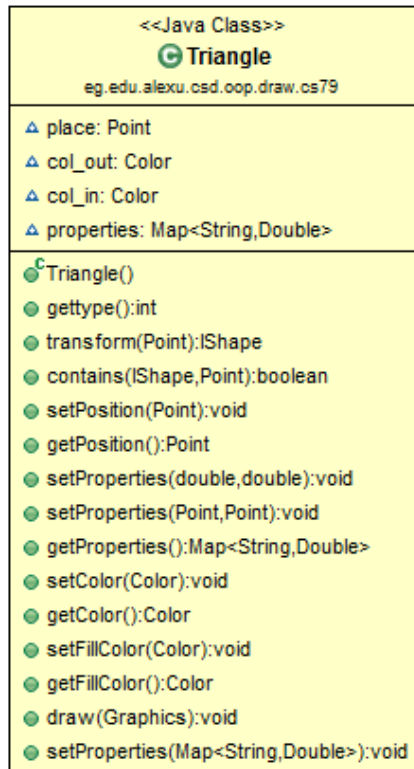
For the line Segment:

```
          <<Java Class>>
             Line
      eg.edu.alexu.csd.oop.draw.cs79

  □ line_color: Color
  △ properties: Map<String,Double>

  Line()
  ● gettype():int
  ● contains(IShape,Point):boolean
  ● transform(Point):IShape
  ● setProperties(double,double):void
  ● setProperties(Point,Point):void
  ● getProperties():Map<String,Double>
  ● setColor(Color):void
  ● getColor():Color
  ● setFillColor(Color):void
  ● getFillColor():Color
  ● draw(Graphics):void
  ● setProperties(Map<String,Double>):void
  ● setPosition(Point):void
  ● getPosition():Point
```

For the Circle and Oval Shape:

```
<<Java Class>>
  Ⓖ Oval
eg.edu.alexu.csd.oop.draw.cs79

△ place: Point
△ col_out: Color
△ col_in: Color
△ properties: Map<String,Double>

●ᶜ Oval()
● gettype():int
● transform(Point):IShape
● contains(IShape,Point):boolean
● setPosition(Point):void
● setProperties(Point,Point):void
● getPosition():Point
● setProperties(double,double):void
● getProperties():Map<String,Double>
● setColor(Color):void
● getColor():Color
● setFillColor(Color):void
● getFillColor():Color
● draw(Graphics):void
● setProperties(Map<String,Double>):void
```

```
<<Java Class>>
  Ⓖ Circle
eg.edu.alexu.csd.oop.draw.cs79

●ᶜ Circle()
● transform(Point):IShape
● setProperties(double):void
```

For the Square and Rectangle Shape:

```
<<Java Class>>
  Ⓖ Rectangle
eg.edu.alexu.csd.oop.draw.cs79

△ place: Point
△ col_out: Color
△ fillColor: Color
△ properties: Map<String,Double>

●ᶜ Rectangle()
● gettype():int
● transform(Point):IShape
● contains(IShape,Point):boolean
● setPosition(Point):void
● getPosition():Point
● setProperties(double,double):void
● setProperties(Point,Point):void
● getProperties():Map<String,Double>
● setColor(Color):void
● getColor():Color
● setFillColor(Color):void
● getFillColor():Color
● draw(Graphics):void
● setProperties(Map<String,Double>):void
```

```
<<Java Class>>
  Ⓖ Square
eg.edu.alexu.csd.oop.draw.cs79

●ᶜ Square()
● setProperties(double):void
```

For the Triangle shape:

<<Java Class>>
ⓒ Triangle
eg.edu.alexu.csd.oop.draw.cs79

△ place: Point
△ col_out: Color
△ col_in: Color
△ properties: Map<String,Double>

⊙ Triangle()
⊙ gettype():int
⊙ transform(Point):IShape
⊙ contains(IShape,Point):boolean
⊙ setPosition(Point):void
⊙ getPosition():Point
⊙ setProperties(double,double):void
⊙ setProperties(Point,Point):void
⊙ getProperties():Map<String,Double>
⊙ setColor(Color):void
⊙ getColor():Color
⊙ setFillColor(Color):void
⊙ getFillColor():Color
⊙ draw(Graphics):void
⊙ setProperties(Map<String,Double>):void

And for all the shapes we implemented an interface called IShape and the following diagram show this implementation:

After design all the shapes that we want we will move to the next step

Second: Drawing and Painting Application:

❖ 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 a number of built in, and possibly extensible set of geometric shapes, and classically, they allow the user to undo or redo any instructions so as to make the application more usable.

Undo and Redo Operations:

These features were made in GUI application so redo and undo machine is that in Class Gui we declared an ArrayList from the type IShape and a Stack.

We store every shape drawn in the frame in the array list so the user when press Undo button we will get the last node in the array and we will put it in the stack and remove it from the frame then if the user press redo button we only will get the last node added to the stack and draw it again.

Colour chooser Operation:

This feature was made in GUI application by using Color Chooser Button, when we pressed on the button it shows this window then we chooses any colour then any shape we will draw will be drawn with this colour.
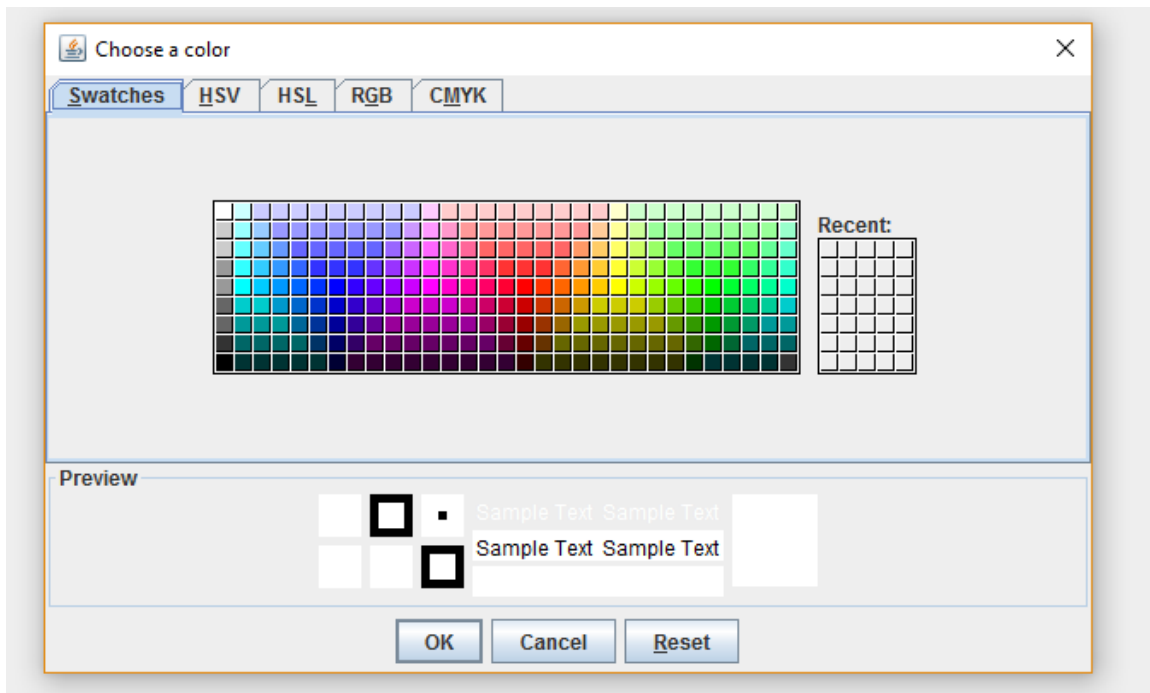
Third: Dynamic class loading:

To dynamically loading a class we loaded the class knowing the path of the jar then took an instance of the class then we dialed with the inner classes using (get method)and invoke then we used two Booleans (enable)for choosing the triangle and (enables)for the square.
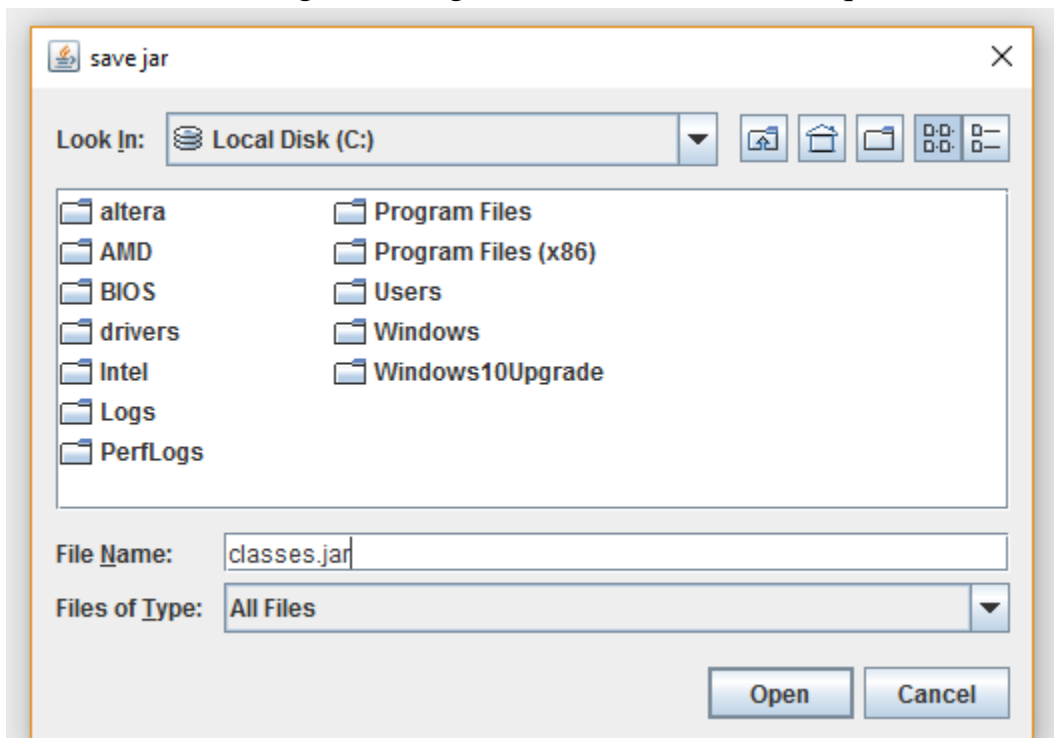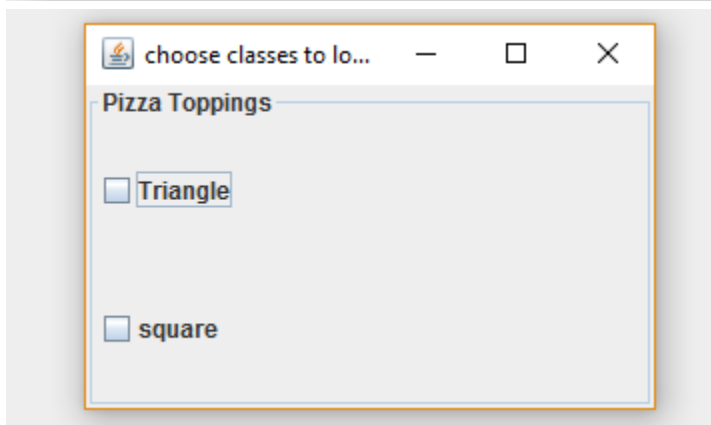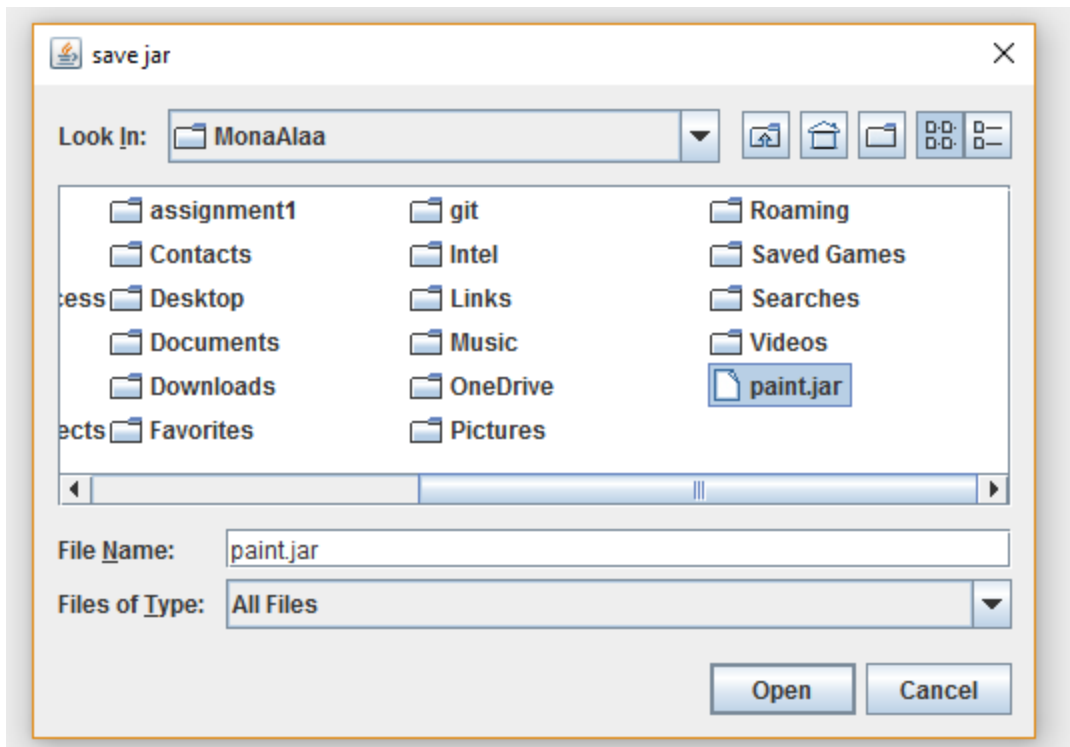
The user will choose one of them to reload then you can use it despite the lack of this class.
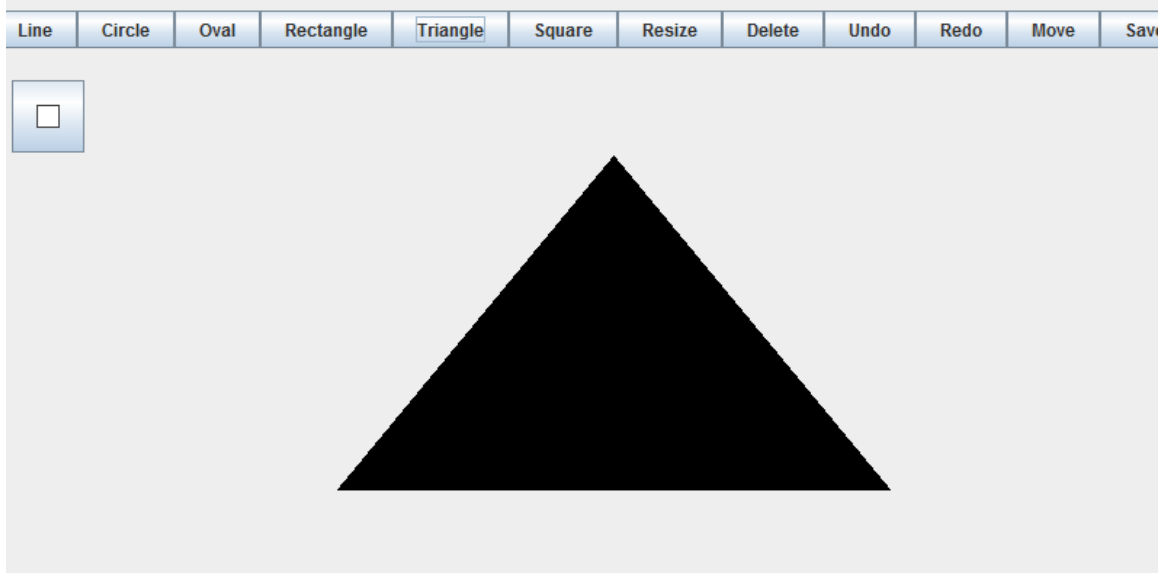
Fourth: Delete and resizing and Movement:

- Delete Operation:

In this feature when we press the button the program will want until the user choose a shape then he will remove it from arraylist and add it to the stack in a case the user presses undo or redo button, and we know which shape the user.

- Resize Operation:

1- In the line we make two small squares in the center of the line and the next line is shown by mouse dragged method. We save the x, y coordinates of the point and by fixing the other point we can resize from that side.
2- In the square and rectangle we consider the top left point is the fixed point and move the other end on the other side of both of them keeping the shapes unchangable
3- In Circle and Oval shape: we have a fixed point which is on the top left corner of both shapes and by dragging the mouse we can control on the radius and axises of the shapes.
4- In Triangle: we consider one fixed point and move it from both points of the base.

- Movement:

   Move operation in this program is represented by transform method, in this method we store the properties of every shape and return the shape with its properties, so when we press on move button we store the point that I pressed and with dragging mouse we determine the point we want to repaint the shape with the same properties and that after remove the shape from the old place.

Fifth: Save and load the window in a file:

In this option we have two options about the place where we will store the window, the first option saving the window in xml file and loading from xml file.

Save in XMLfile:

This XML document does not do anything. XML is just information wrapped in tags. Someone must write a piece of software to send, receive, store, or display it.

XML stores data in plain text format. This provides a software- and hardware-independent way of storing, transporting, and sharing data.

XML also makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

So in this program we the properties of every shape in a tag as we stored it in the map properties and the property of every shape in an element in the document so we can load it again.

Load in XML file:

to load from xml file we load the properties of the shape by the name of the tag of the elements which represent the shape we want to draw.

Save in Json file:
To save json convert the java object which from the type of the interface to a string using mapper.

Load in Json file:
To load json file we should convert the string file to an object then to string as we used interface and there can't be an instance of the interface so we manually divided the string file to substring so that we could deal with each object solo then we take this substring put them in array convert them to json objects deal with them using the type method then convert them to object and add them to the arraylist which would be draw.

In this program we made checkbox dialogue when we press on the button save, a dialogue is shown has two check box one of them is for saving in json file and the other in xml file and the same thing for load button.

This project is the effort of two university students

Name: Mona Alaa Darwesh Mustafa   ID: 76

Name: Aya Fouad Metwally Abu-Nasr ID: 3