Programming II

Paint Application

Esraa Hessiun – 4375
Mohammed Saber – 4245
Shady El-Shafie – 4270
Ahmed Hammad – 4268

Description:

A simple paint Java application that allows the user to draw and color geometric shapes using simple tools.

Features:

- User-friendly Javafx Gui interface.
- Ability to draw, resize, move, recolor and copy using mouse actions.
- Ability to delete, undo, redo, save and load by simple button clicks.
- Provides several warning messages to avoid any crashes during run time.

Design Overview:

We are using MVC architectural pattern to separate and organize the classes.

The model contains some classes related to geometric shapes and its classifications, as long with an interface to allow multiple inheritance.

The view section consists of FXML file with the support of a single CSS stylesheet.

And The control section, comes the functionality provided to the user -organized among several classes-, from which it can use Model’s APIs, and provide a visual results to the view section.

Assumptions:

- Our Application does not support saving or loading using JSON formats. However, it works perfectly for XML files.

- When saving a shape with a transparent fill color, it will be loaded back with black fill color, because our GUI framework does not support a code format for the transparent color.

- We did not find a useful use for either “Command DP” or “Strategy DP” in our design, may be the approach we took did not match the expected design. However, All other essential design patterns, in addition to Observer DP was correctly implemented to provide efficient, clean code.

- We do not provide dynamic extension for our application.

Team Work:

Esraa Hessiun and Shady El-Shafie were responsible for coding the Model section, implementing the classes, interfaces functions and providing helper functions to be used in other sections.

Mohamed Saber was responsible creating the design, providing the team with all the UML diagrams, making continuous changes to meet the coding options.

Ahmed Hammad was responsible for designing the view section, linking it with its controller classes, in addition to file handling.

Data Structures:

We used a couple of stacks to allow Undo and Redo functionalities.

We used some ArrayLists to store shapes in different parts of the program.

We used a HashMap to store the properties of the shapes, to allow saving and loading them into XML files.

We used ObservableLists which uses “Observer DP” to keep the user notified about every change in the List of shapes.

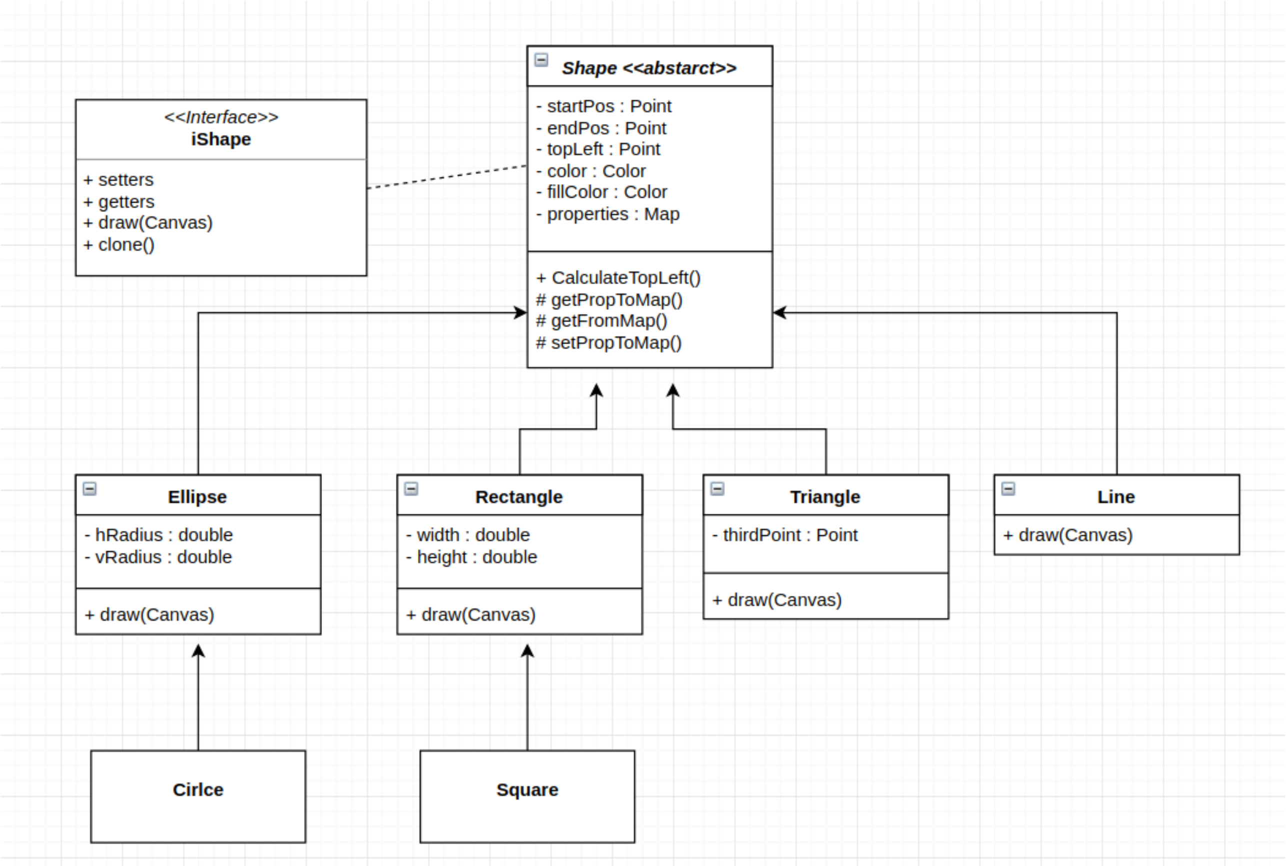
Important Functions:

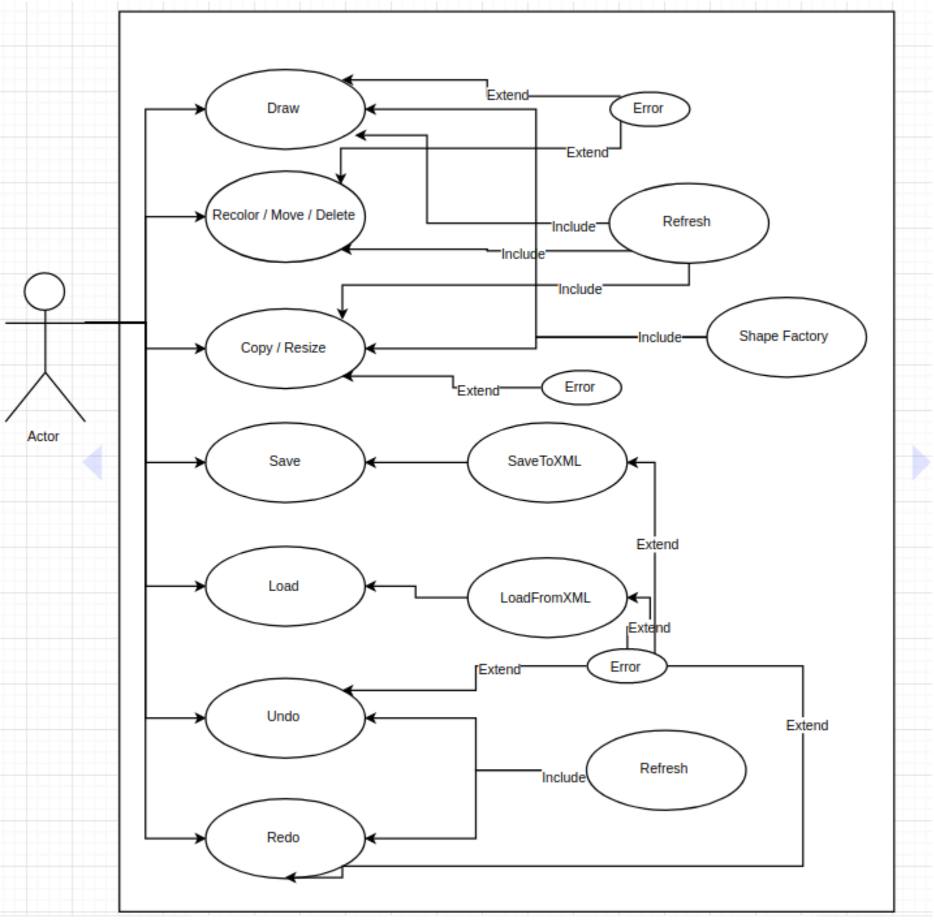
SaveToXML, LoadFromXML are two classes that provides a set of helpful functions that take care of the saving and loading process.

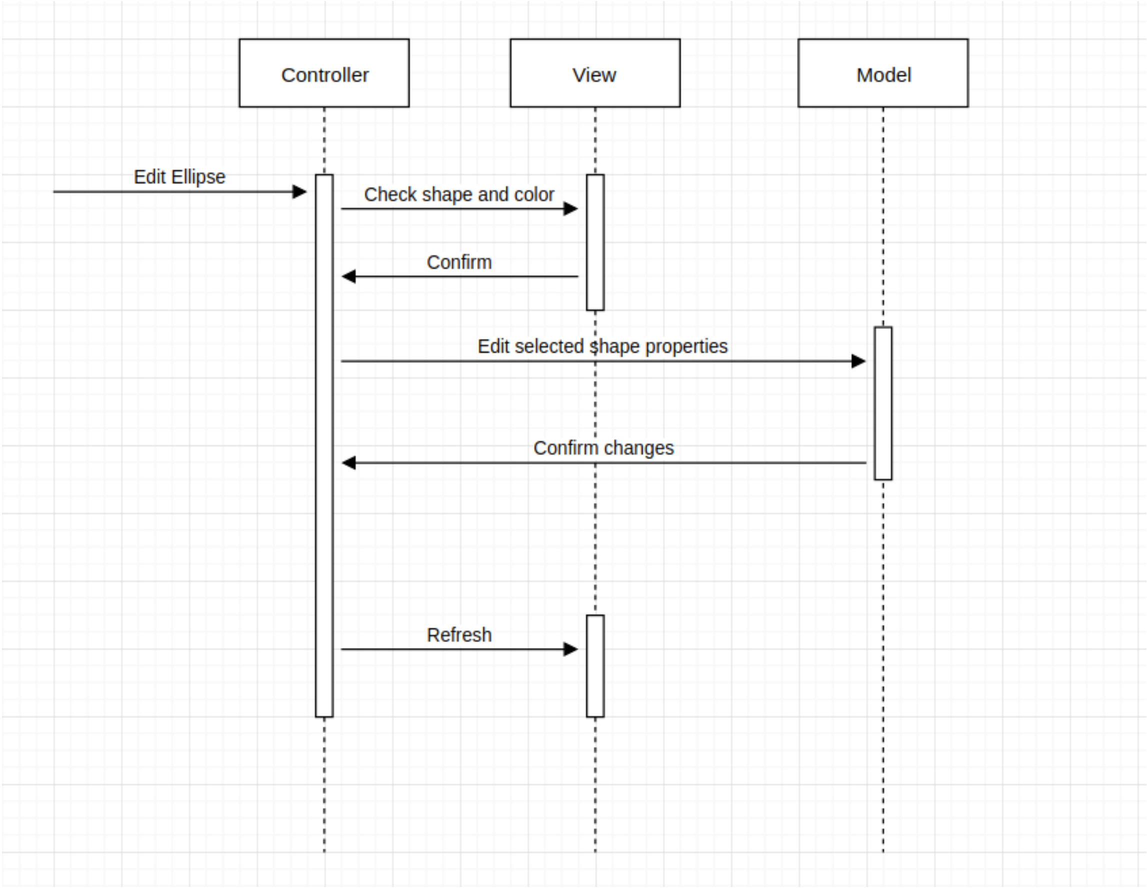
ShapeFactory is a class that has some Overloaded functions, to create a shape and return it to the caller function, given any set of useful parameters.

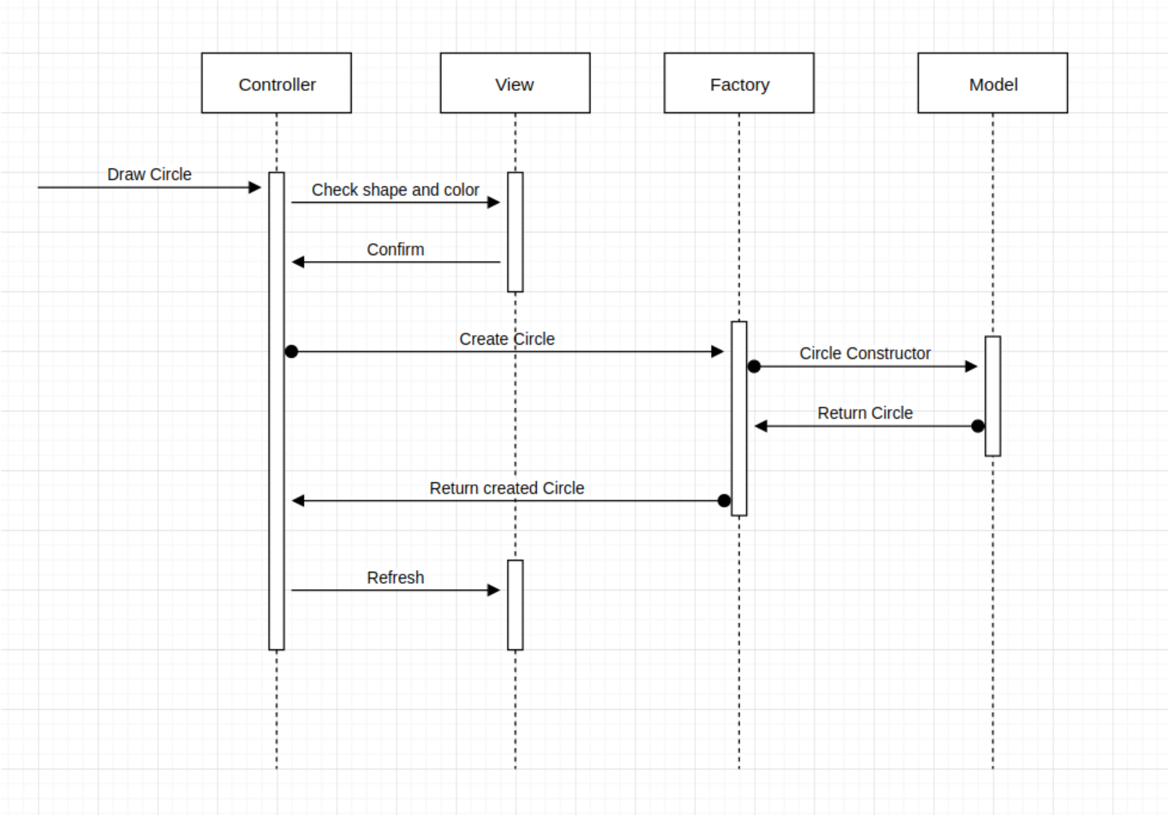
The controller class has too many functions to list, that are all simply, converting user actions into calls for other useful functions, and returning the results to do some view functions.

UML Diagrams:






Controller
View
Model
Edit Ellipse
Check shape and color
Confirm
Edit selected shape properties
Confirm changes
Refresh


Controller
View
Factory
Model
Draw Circle
Check shape and color
Confirm
Create Circle
Circle Constructor
Return Circle
Return created Circle
Refresh

User Manual:

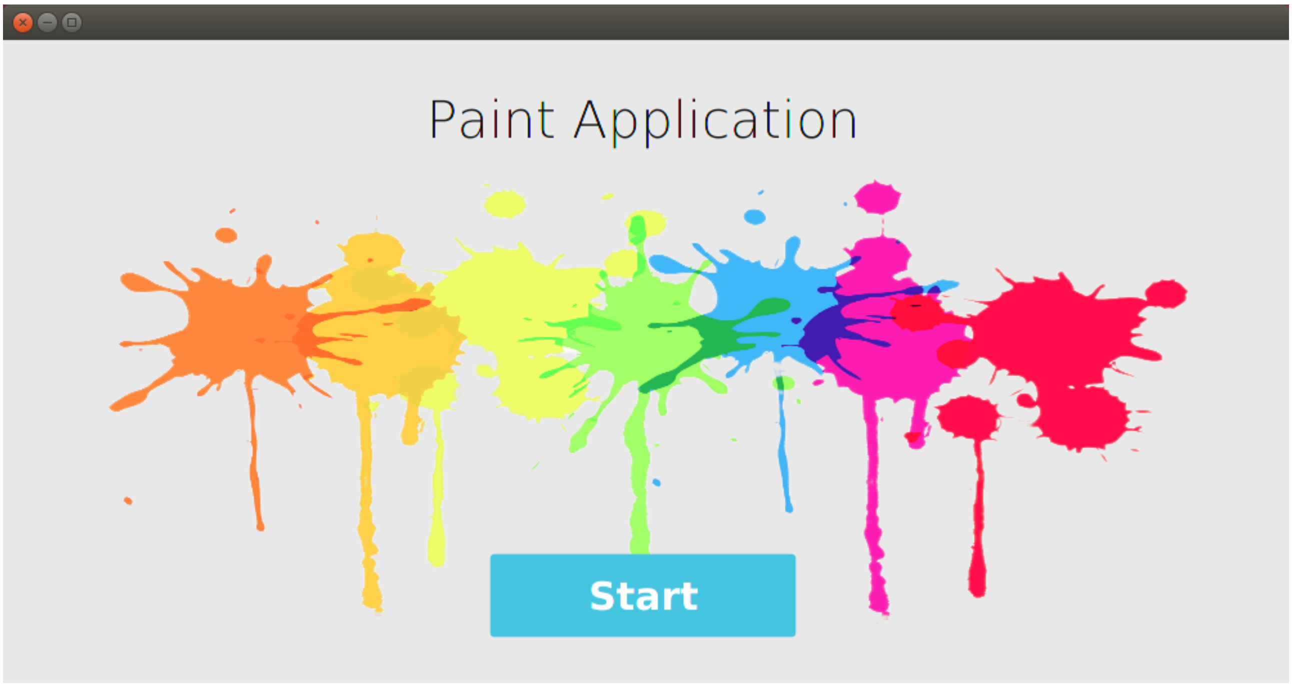
On the start of the application, You need to click the start button to get access to the drawing tools.

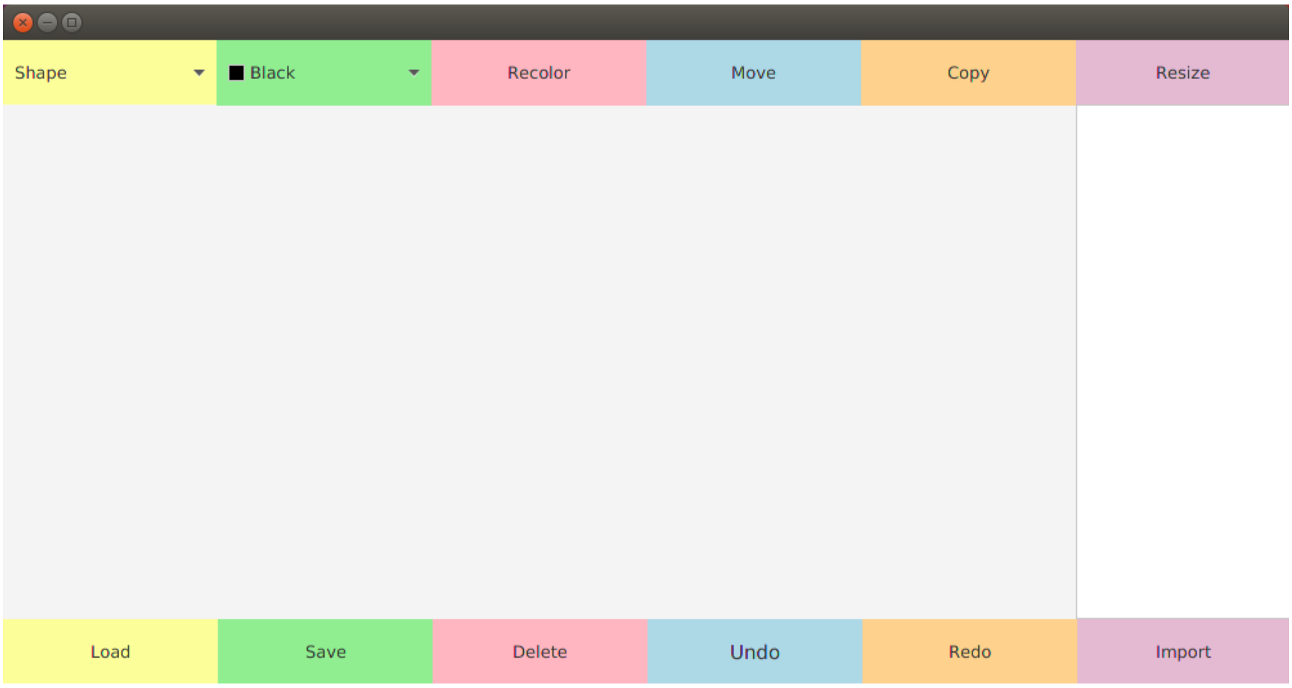
The interface is divided into mainly 4 parts, a set of buttons, rowed on the top and bottom, a list of the drew shapes, located at the right, an initially hidden message bar, just below the top buttons row, and finally the drawing canvas in the remaining area.

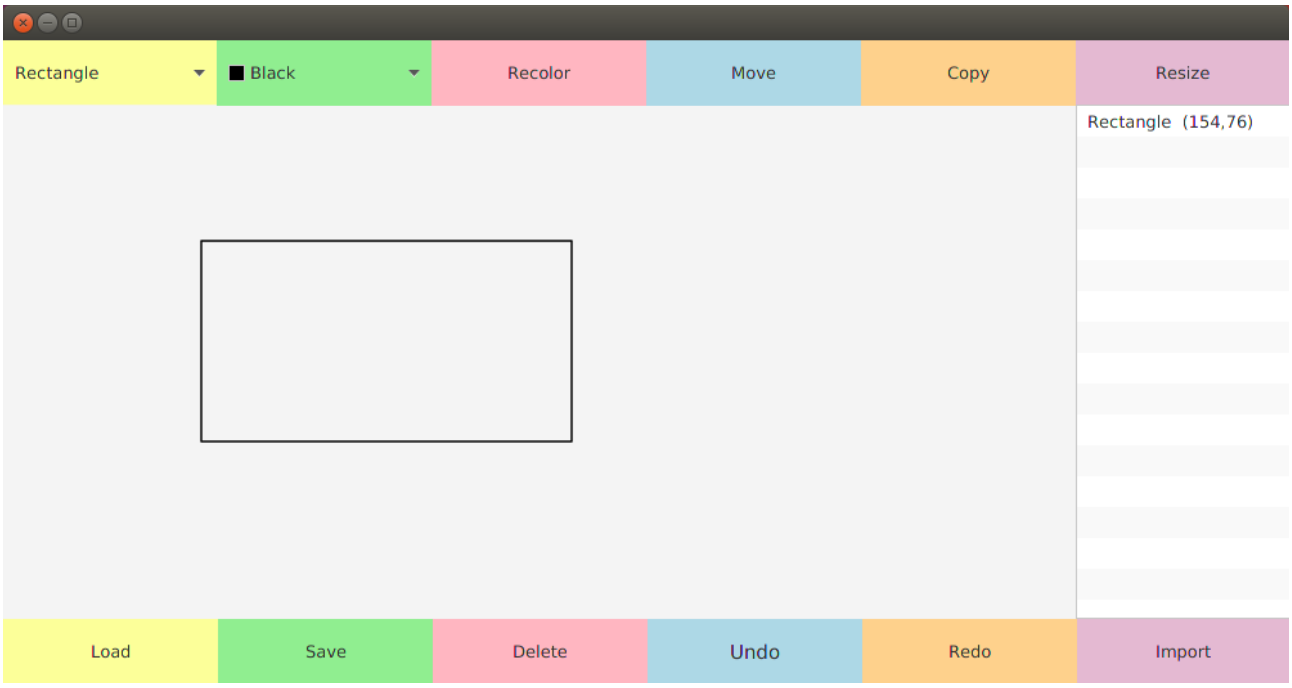
In order to draw, you need to pick a shape from the top left button, pick the strock color, and drag using the mouse in the canvas area.

In order to modify any shape by any mean, you will need to choose it first from the list, click on the button you desire, a message will tell you what to do exactly then.

Sample Runs:


Paint Application
Start

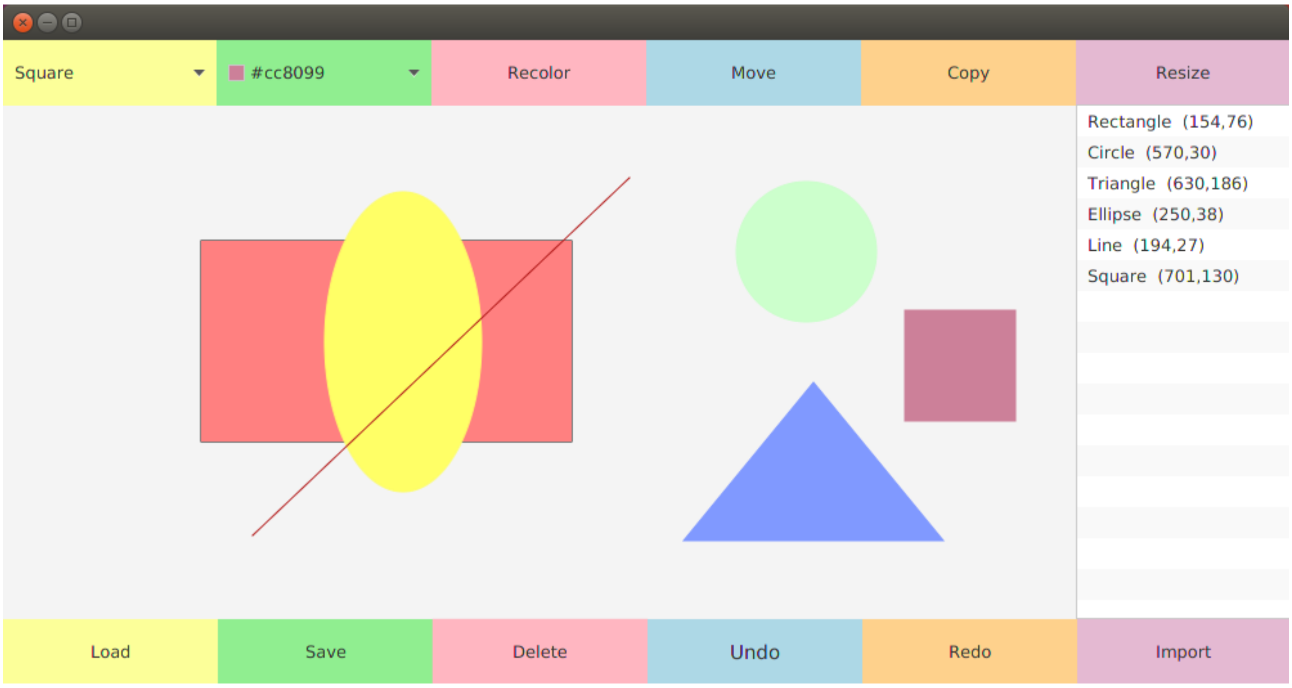

Shape
■ Black
Recolor
Move
Copy
Resize
Load
Save
Delete
Undo
Redo
Import



Rectangle

■ Black

Rectangle (154,76)



Square

■ #cc8099

Rectangle (154,76)
Circle (570,30)
Triangle (630,186)
Ellipse (250,38)
Line (194,27)
Square (701,130)

References:

http://bit.do/xmlPaintProject