**DRAW REPORT**

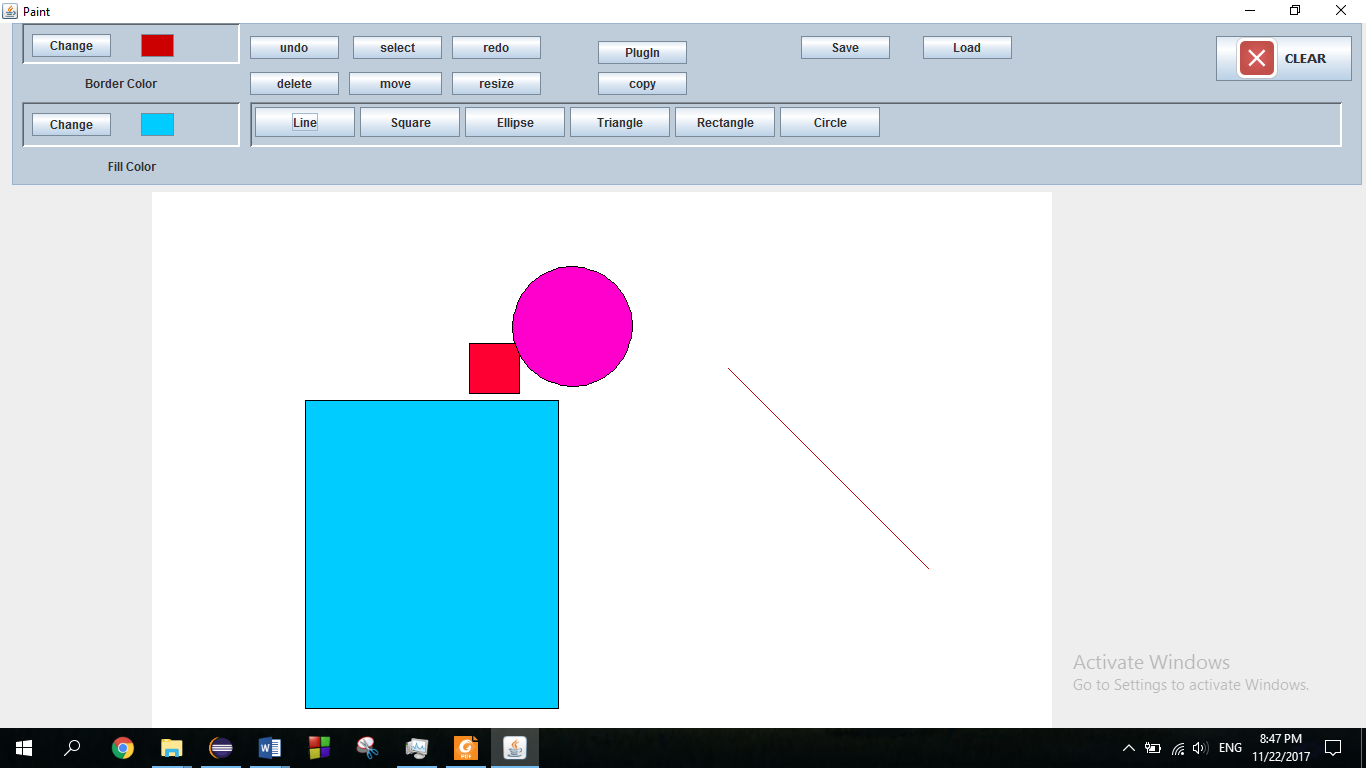
Ziyad el banna

Ahmed khaled Abdel Sayed Abdel Aal [05]

**Introduction**:

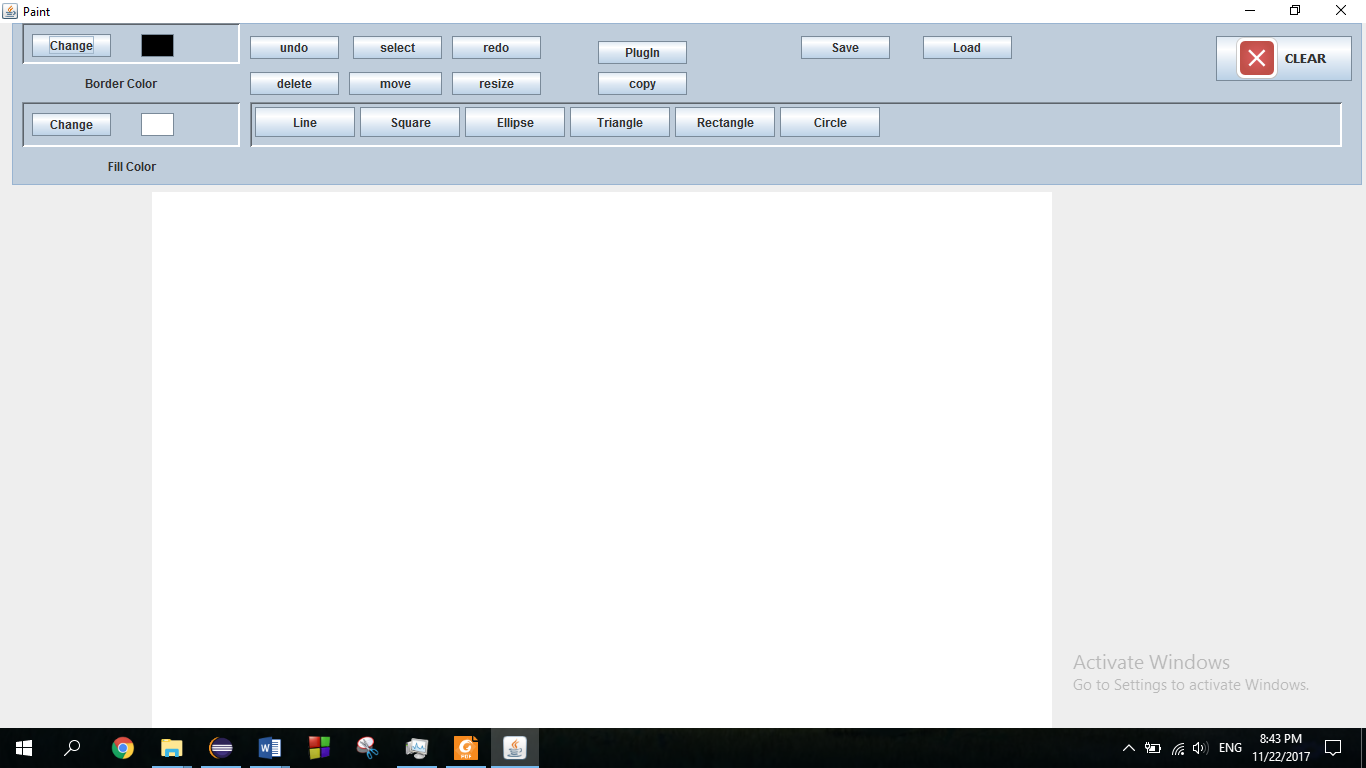
Drawing and painting applications are very popular and have a huge user base. They generally offer a big number of features that includes but is not limited to Drawing, Colouring, and Resizing. They also include several built in, and possibly extensible set of geometric shapes, and classically, they allow the user to undo or redo any instructions to make the application more usable. In this program, we designed a simple draw program with many exciting features. For example, many geometric shapes can be drawn such as line, rectangle, square, and many other shapes. After drawing the shape, it can be deleted, copied, moved, or resized. Also user can save shapes in order to load them in the future. In addition, user can add new shapes to draw to the program.

**GUI design:**

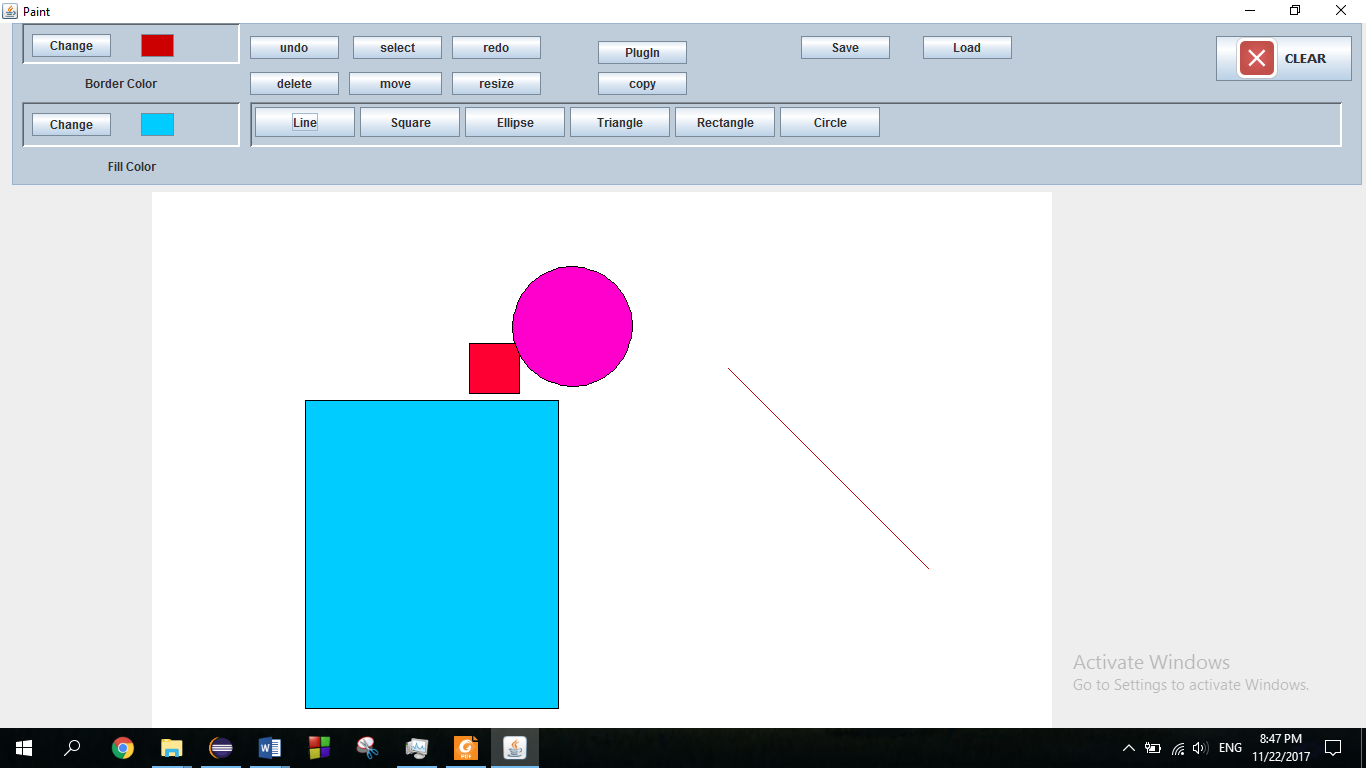


The GUI consists of 2 main areas: the buttons area and the drawing area.

The buttons area contains all the needed buttons for the program as shown below.



The draw area contains all the drawn shapes as shown.



**Buttons:**

1. **Shapes {line, square, rectangle…}:**

This group of buttons contains all the available shapes that user can draw. If a new shape is added, its button will dynamically appear with this group of buttons. The whole drawing process will be discussed in the design assumptions.

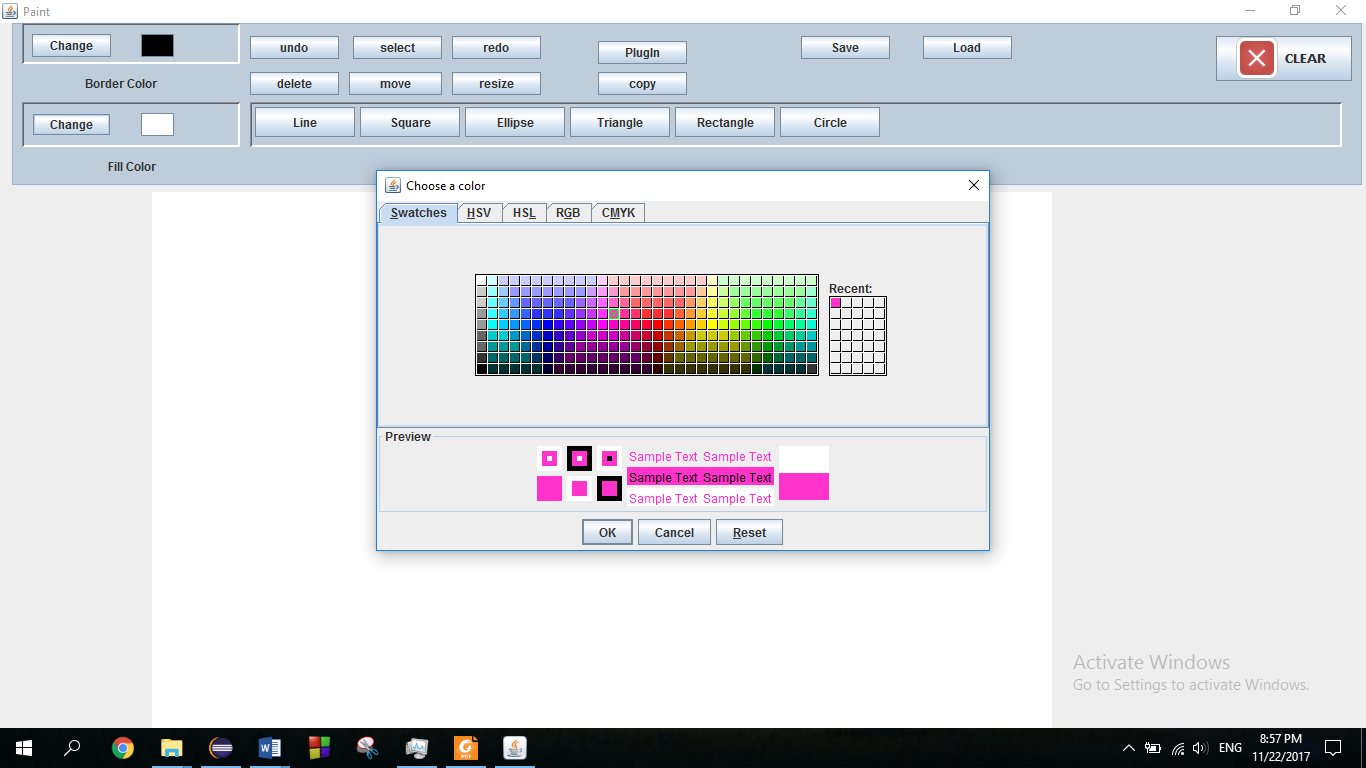
1. **Border Color:**

This button allows user to change the outer line color for the next shapes.

1. **Fill Color:**

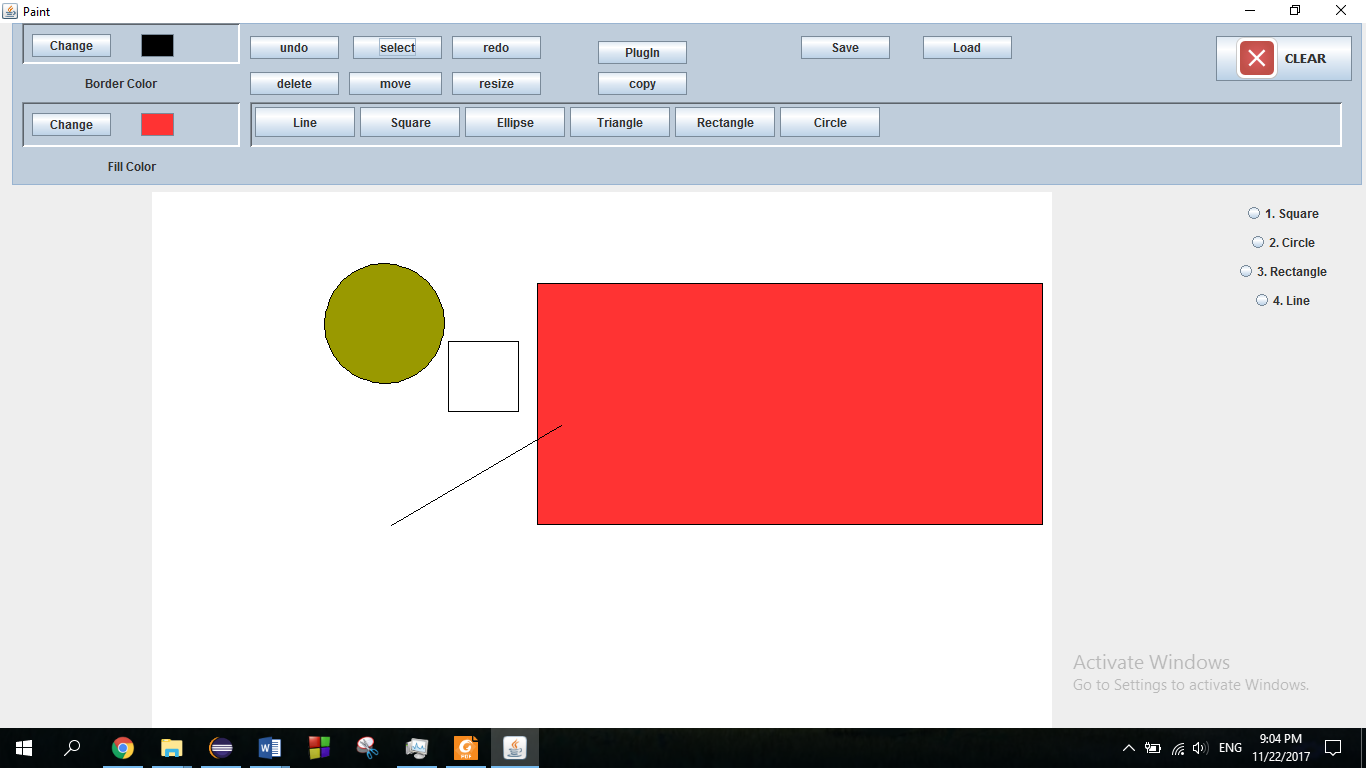
This button allows user to change the color that fills the next shapes.

When fill color or border color, a popup window appears for the user to choose suitable color as shown.



1. **Select**:

This button is pressed before modifying any shape on the drawing area. When it is pressed, a list of all available shapes appear, user can choose any number of shapes to modify all of them together.

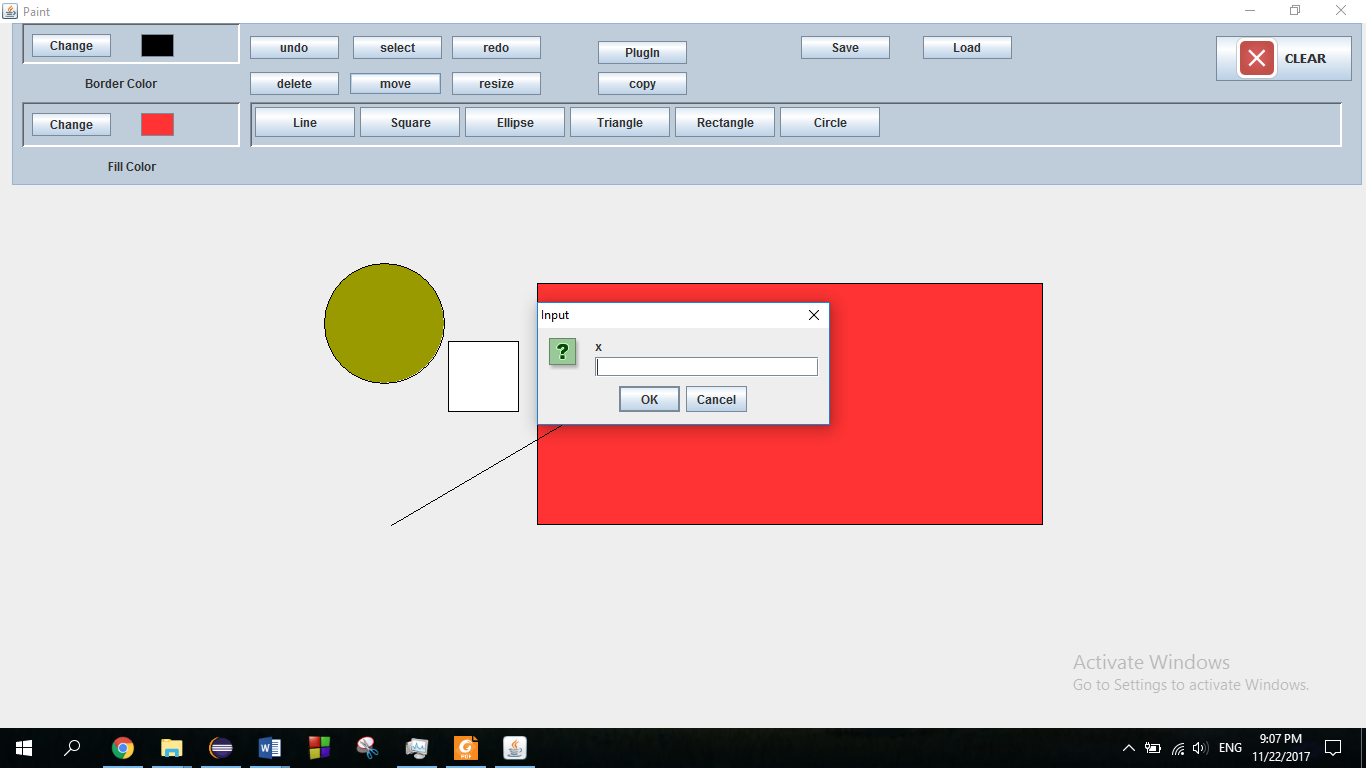
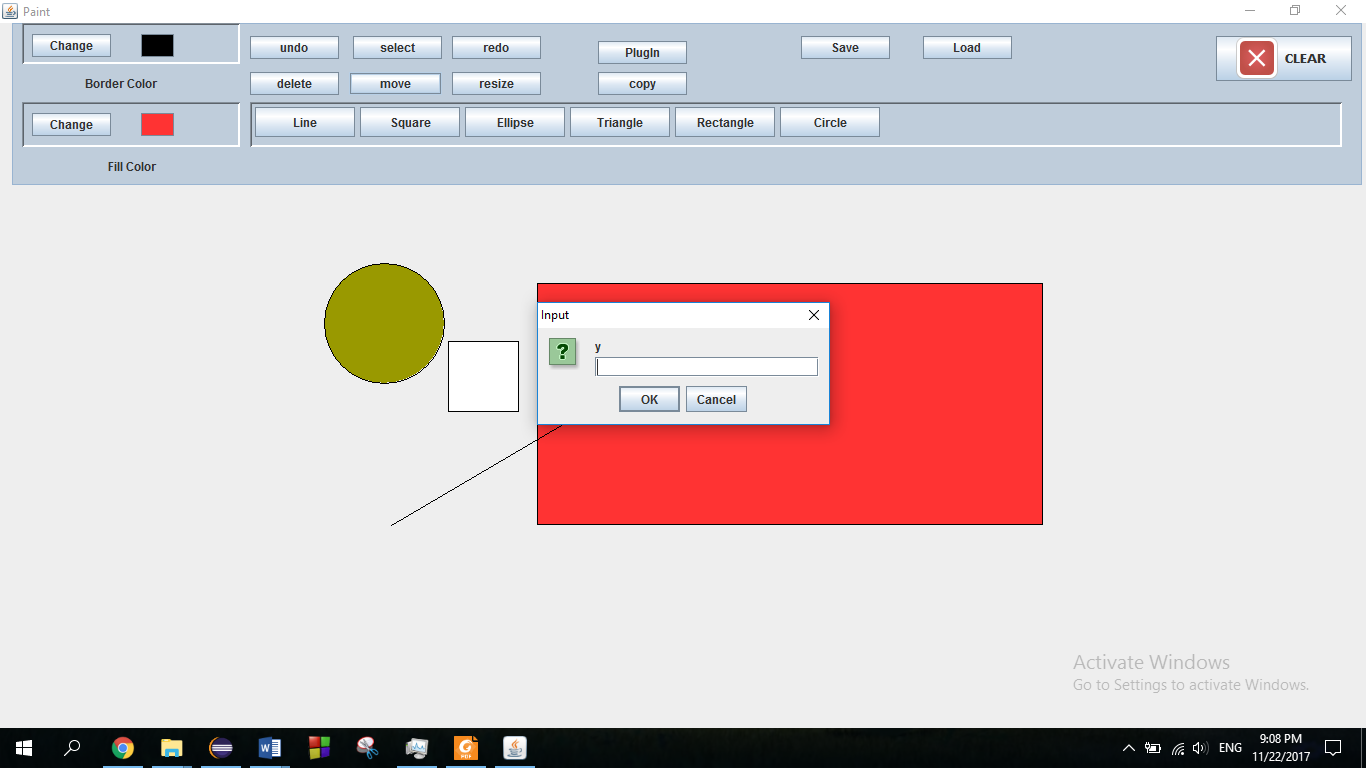


1. **Delete**:

After selecting shapes, user can delete all of them just by clicking on Delete button.

1. **Move**:

After selecting shapes user can also move them using this button. When button is clicked, a popup window appears for the user to choose the distance on the x and y axis for shapes to be moved:

1. **Resize**:

This button allows user to re-enter the properties of any selected shape he wants. When button is clicked, popup windows will appear asking for the new attributes of the selected shapes to update.

1. **Undo**:

This button allows user to undo anything he have done up to 20 step.

1. **Redo**:

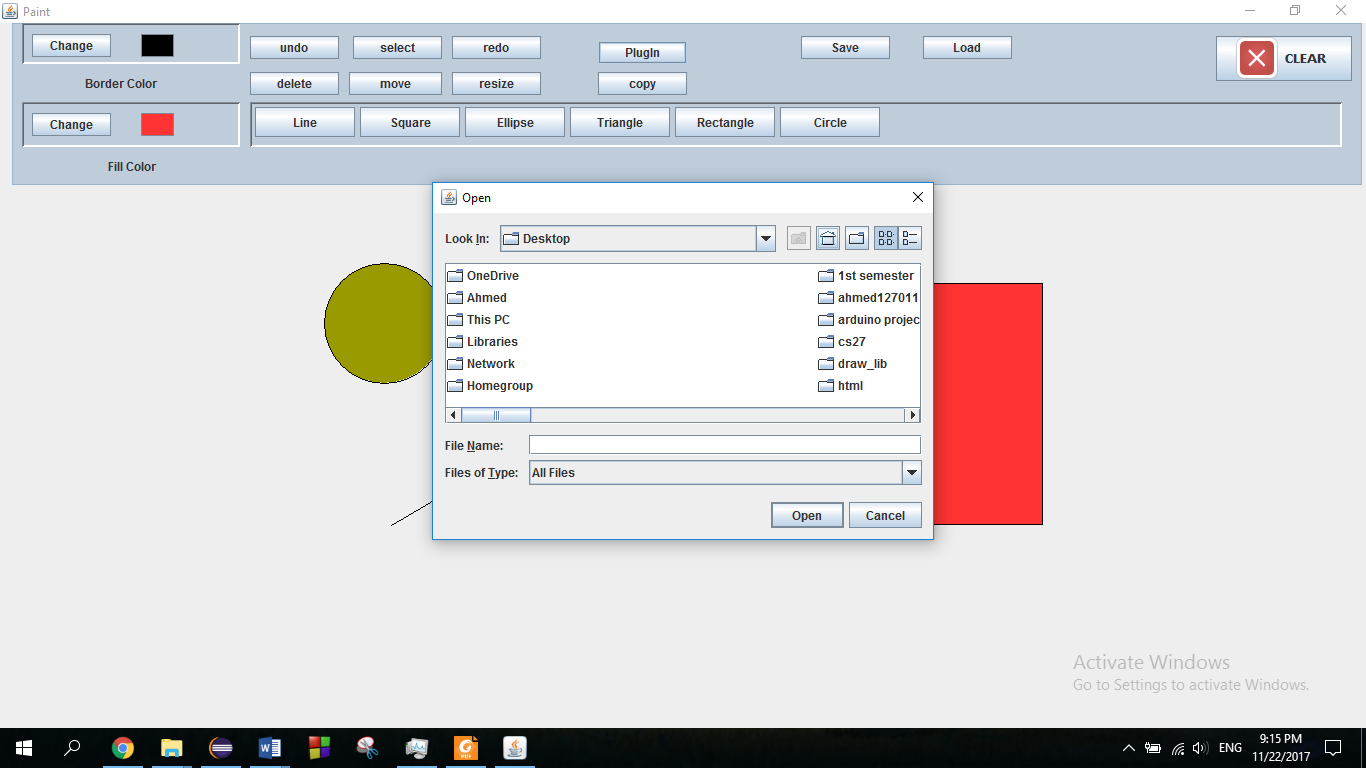
This button allows user to redo anything he have done up to 20 step.

1. **Copy**:

After selecting shapes user can also copy them using this button. When button is clicked, a popup window appears for the user to choose the distance on the x and y axis for shapes to be copied. It is the same as move but old shapes are not removed.

1. **Plugin**:

This button is used to add new shapes by allowing the user to choose their kjr file for the window directory.



1. **Save and load:**

Those two buttons are used to save the drawn shapes in an XML file to load from it in the future. Those buttons allows user to choose the path of the file to save in and the file you want to load.

1. **Clear**:

This button when clicked it removes all shapes from drawing area for a fresh restart. But be aware that after clicking on it you can’t undo or redo anything.

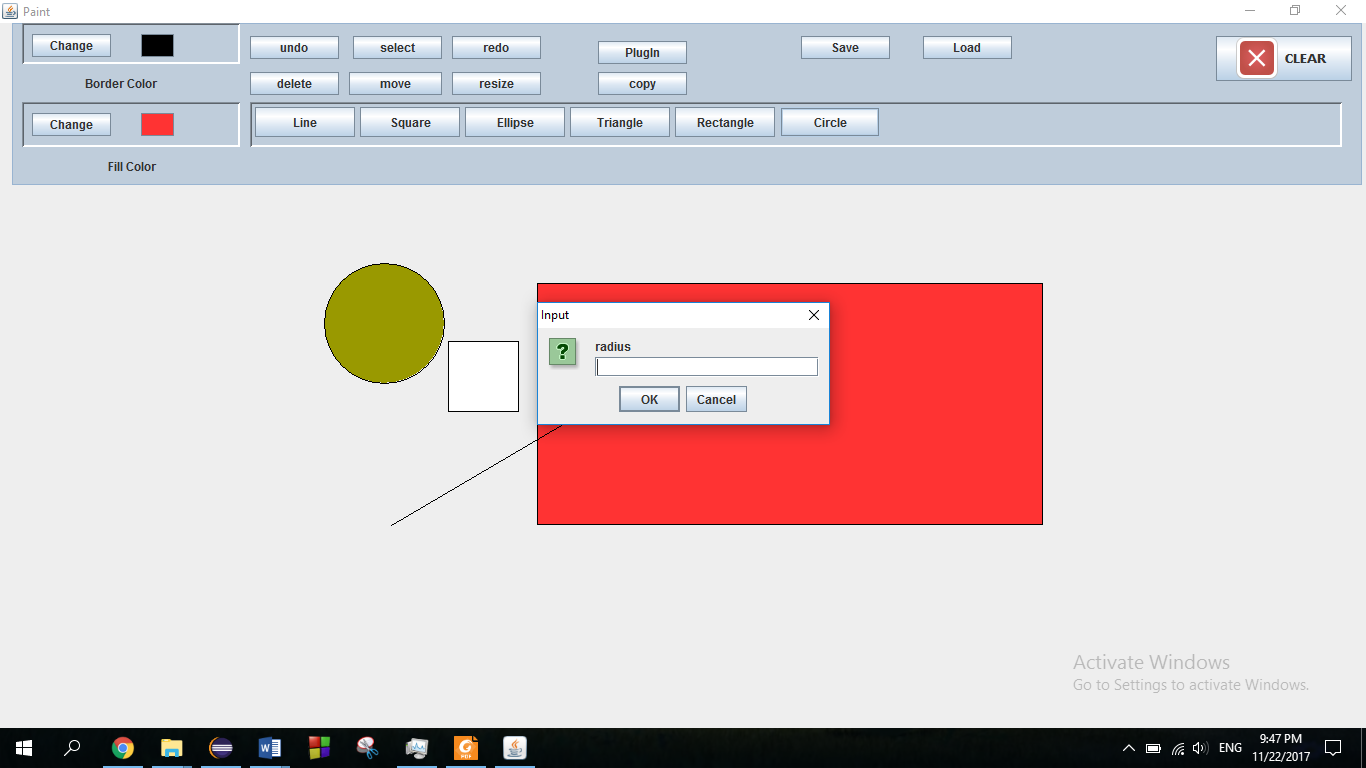
**Design assumption**

Some points needs to be clarified in this section:

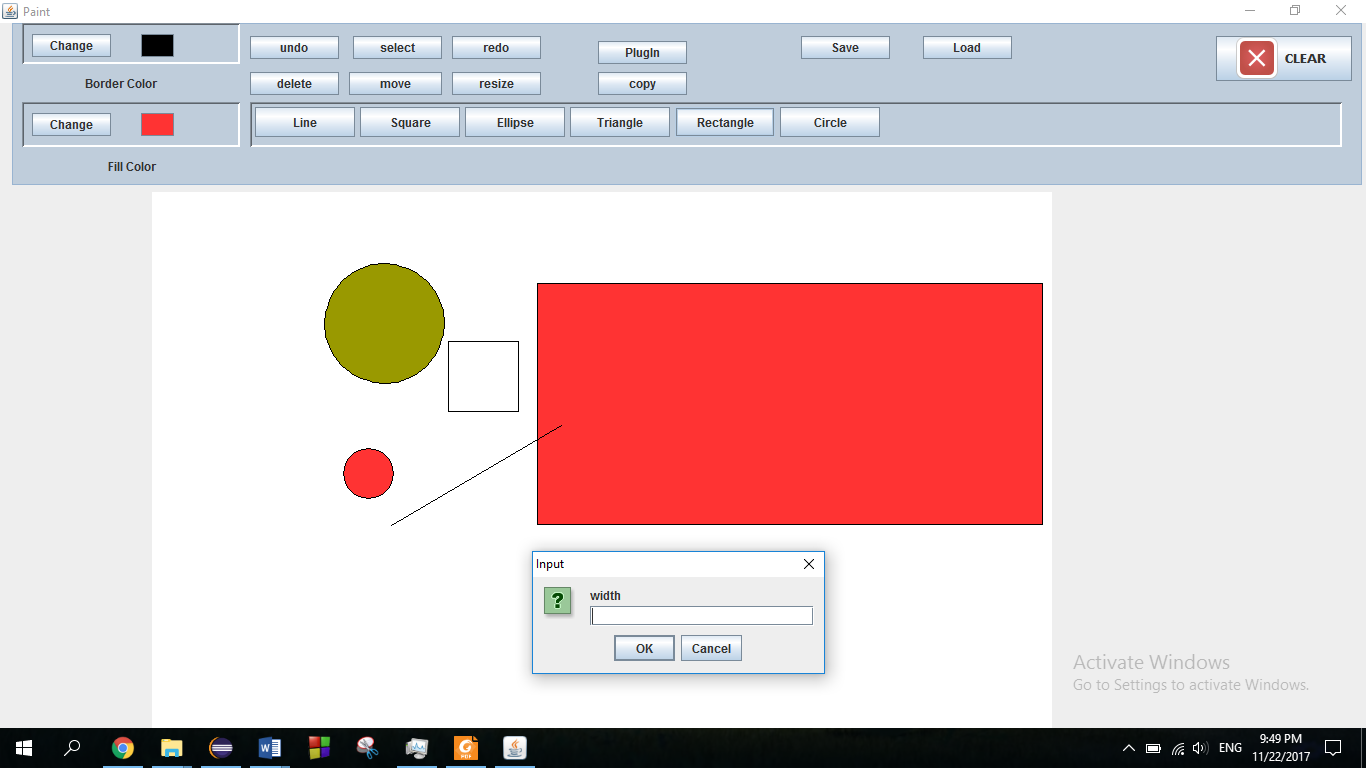
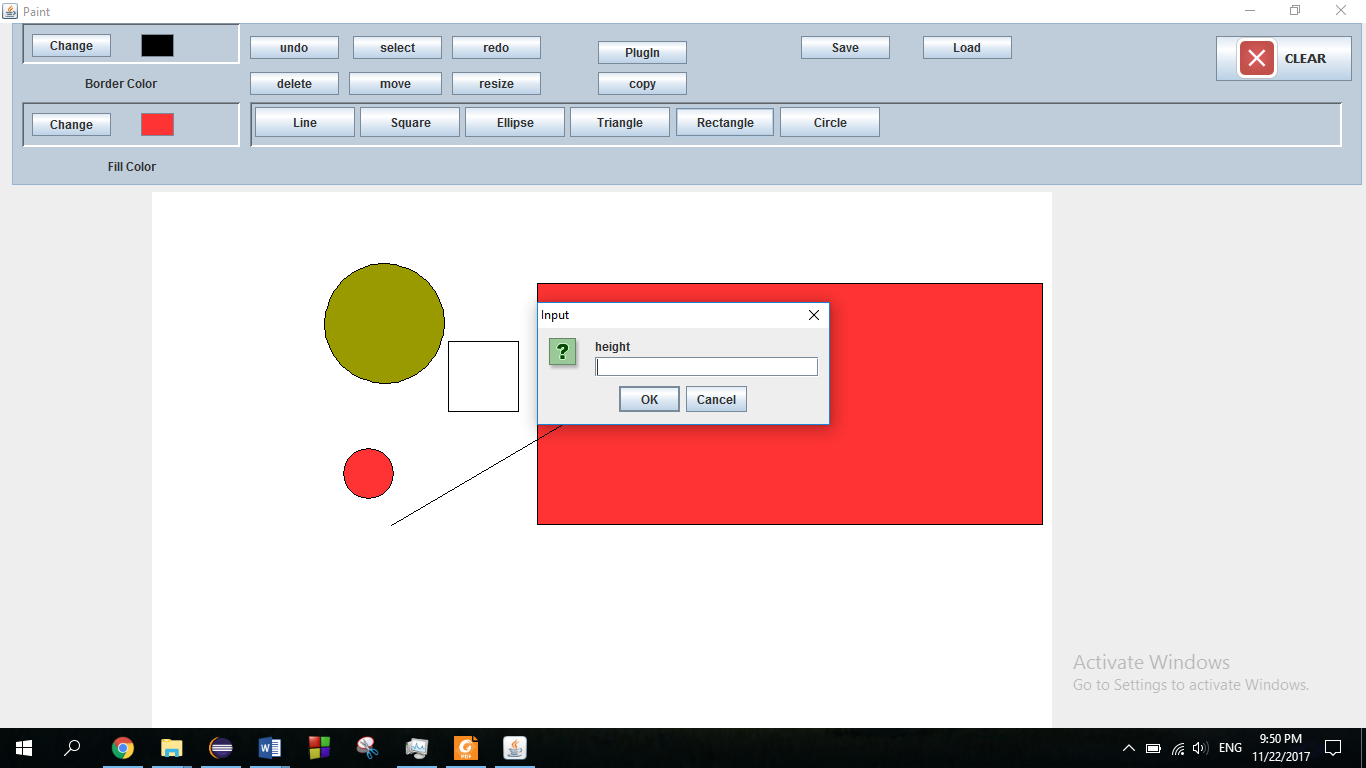
* **Drawing process:**

Drawing process consists of 3 steps:

1. Choosing the shape: this happens by clicking on the shape button
2. Entering the shape properties: number of popup windows will appear after clicking on the shape button. This number depends on the number of properties of the shape. For example, Circle has only one property, radius, so only one window will appear.



On the other hand, rectangle has two properties, width and height, so 2 popups will appear.

1. Choosing the position of the shape: this happens by clicking on the drawing area. When clicking, the new shape will be drawn immediately.

* User can add any number of new shapes using the plugin button. But after closing the program, user can not use them again. He should re-add them to the program.

**DataStructur and classes**

in this program we used 2 intefaces and 10 classes in addition to the gui.

Interfaces:

1. **Shape inteface** is for the shapes to be drawn.
2. **DrawingEngine** interface is for controlling the process of adding or removing shapes from the canvas while the program is running.

**Classes**:

* **drawShape** class: this class implements Shape inteface. It contains all its methods but draw and set properties.
* 7 **classes of shapes**: those classes extends drawShape class. Each class contains its own constructor,draw method, and set properties method. For example, circle class draws the circle using the radius passed on the properties map and position point.
* **drawEngine** class: this class implements drawingEngine interface. It contains all the logic of saving, loding, adding shapes and so on.
* **Mypanel** class: it extends jpanel and used to do all the drawing stuf. It deals with our drawing area directly and used to draw, delete and update shapes on the canvas.

**Datastructures:**

Most of datastructues used are on the drawing engine class:

1. **Shapeslist** <Linked list of linked list>: this linked list is used to save shapes drawn while the program isrunning. Everytime a new shape is drawn, all the shapes on the canvas, including the new shape, are added to the shapeslist as a linked list. This makes undoing and redoing easier by only using a pointer.
2. **Classeslist** <Linked list of classes>: this linked list contains all available classes to be drawn with. When a new class is added using the plugin, it is added to the linked list.

In addition, some linked lists are added to the gui to make it dynamic. For example:

1. **Selecte** <Array list of shapes>: it contains all the selected shapes that user selected to move, copy, delete, or resize. It is cleared after every operation of the previous ones.
2. **buttons** <Arraylist of buttons> : those buttons are generated according to the available classes on the drawEngine class.
3. **Rbtns**< Arraylist of radio buttons>: those buttons are generated according to the drawn shapes on the canvas.

**User manual**

**Draw a shape:**

1. Click on the shape button.
2. Enter the shape attributes acoording to the popup window.
3. Select the position of the shape on the drawing area.

**move or copy shapes:**

1. Click on select button
2. Choose shapes to be moved
3. Click on move button
4. Enter the x and y distance to be moved

**Delete shape:**

1. Click on select button
2. Choose shapes to be deleted
3. Click on delete button

**Resize shape:**

1. Click on select button
2. Choose shapes to be resized
3. Click on resize button
4. Enter the new properties of the selected shapes.

**add new shape class:**

1. Click on the plug in button.
2. Choose the .jar file of the class.
3. The button will show up next to the shapes buttons.