

**Alexandria University**

**Faculty of Engineering**

**.Computer and Systems Engineering Dept**

**CS221.**

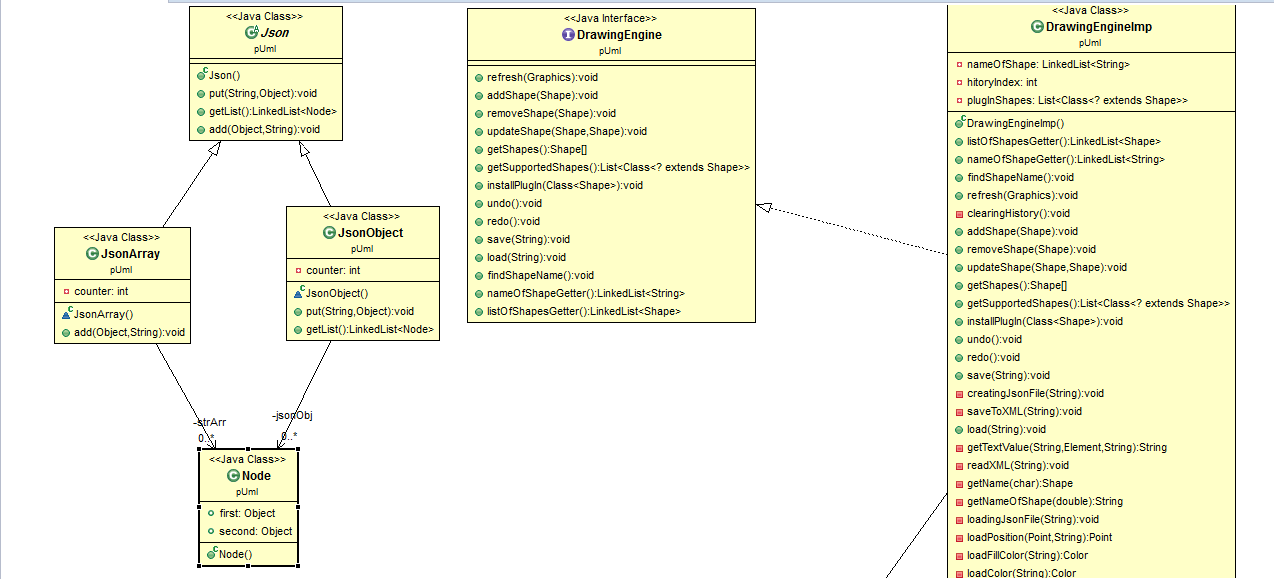
https://docs.google.com/drawings/d/s6Da6LqJKW7ZbombaTZCxKg/image?w=584&h=3&rev=1&ac=1

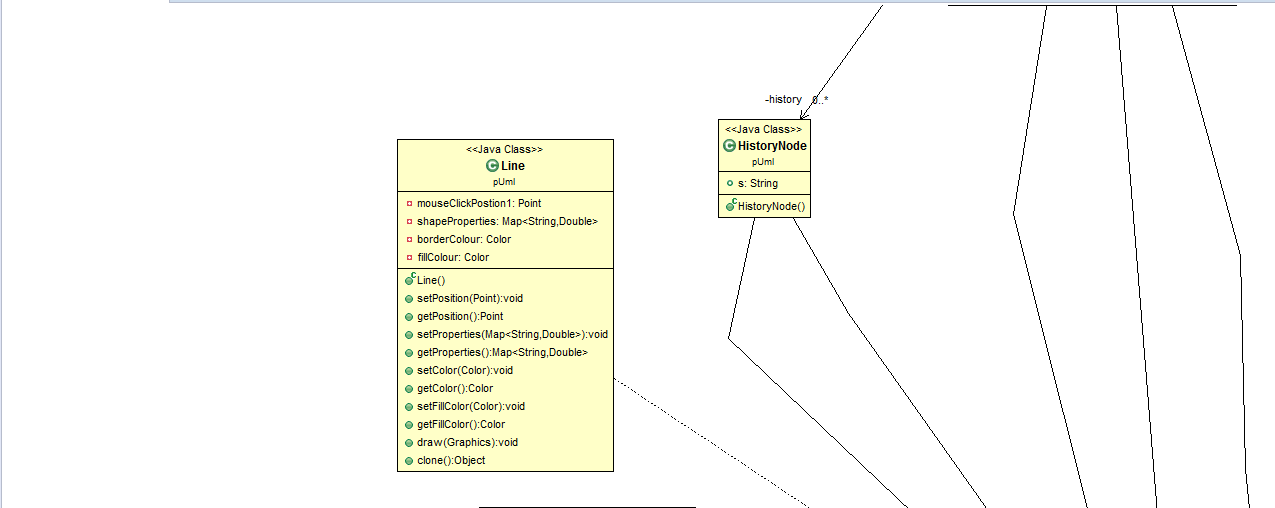
**Paint**

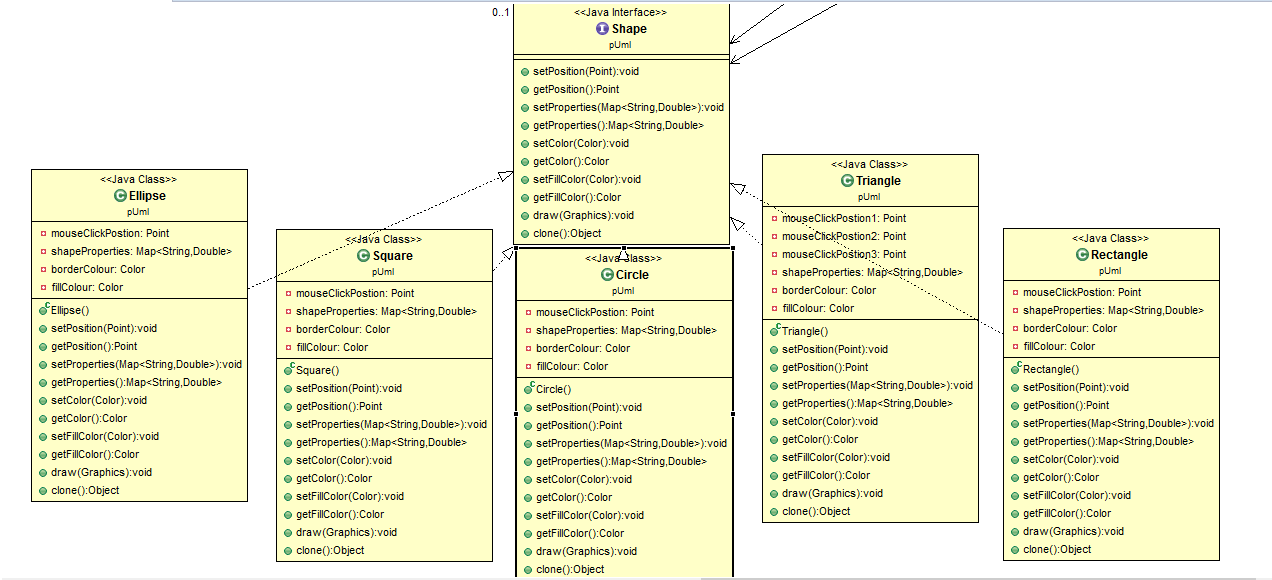
Names:                                           Ids:

* Sara Ehab Wahdan.                          30.
* Belal Adel El-Haddad.                       19.

**Paint Uml Diagram:**

****

****

****

**Design Description:**

By using the MVC design pattern , we implemented our code in three ***packages*** :

One for the view (**v**) , the other one for controller(**c**)and the last for the model(**M**)

**In package (v)** : we put the main for the GUI and the design for the GUI

**In package (M)** : we put the implementation for both interfaces 'Shapes' and 'Drawing engine' and the factory design pattern for generating instances from the different shapes

**In package (C)** : we put the controller which attached the implementation with the view

: A linked list '**history**' used for recording the user actions in the GUI

When the user presses a button for drawing shapes it calls the handler event method for the button which called 'createShape' which create a string holding the shape type to be sent to the factory design pattern to create an object

After creating a shape the user can drag , move and resize it . the drag done by selecting the shape properties like the fill color and the border color then the user press the mouse on the canvas and drag it to draw the shape in different sizes until releasing the final shape is added to the shape list and to the history, a toggle button is added to the list view to present the drown shape

**Moving** process done by creating a copy from the selected shape with the same properties then by using the same dragging technique only its position is changed

**Resizing** is the opposite process of move as the position is fixed and the changing done on the proprieties

**Removing** shape is to delete it from the canvas and its done by deleting it from the list of shapes which contain the drown shapes then clearing the canvas and drawing the list again and of course recording this event in the history

**Undo** is canceling the last event the user takes that’s why we have created the history list so we can remember what the user have done before so undo is to go backward in this list and doing the opposite the user has done

**Redo** has the same idea of the undo but the opposite as we take the action the user took and doing it again

**Update** shape is a process of creating a new shape with new properties and replacing it with the old shape that the user needs to update and recording it in the history list so when the user does undo or redo the history list will carry both the old shape and the new shape to switch between them

**Save** : the user can save his canvas by two ways (XML file , JSON file)

First , choosing where to save his files then choosing a file between the two kinds and the code will know his way to save with the right method for each type

Then , we can find the file where the user has chosen , with the name he decided and of course ready to be loaded at any time

**Load** : after saving the shapes in XML file or JSON file the user can load any one of them using the same way of saving , the user can chooses the file with any type of them both and the code knows what to do next to draw the shapes in the file into the canvas again

***The user guide :***

**Drawing a new shape** : the user has to press the button of the shape he wants to draw then choosing its colors and dragging the mouse in the canvas will draw the shape ha wants until he releases the mouse the final shape will be the one he decided to be added to the shapes and a toggle button will appear in the view list to represent this shape

**Save & load** : when the user wants to save or load in any time of the program he only need to press their button an d a pc window will appear to choose where to save or load his file

**Plugin** : if the user wants to draw a shape which our program didn’t support ,think what, we can do that by loading its jar to our program and he can draw by the same way he draws the shapes and that is done by pressing the newPlugIn button to open the pc window to select the jar file and a button will appear to support this new plug

**Copy & Remove** : the user first needs to select the shapes he wants to do this action on them by pushing their buttons and then pressing the action button he wants to take

**Move & Resize** : after selecting the shape he wants and pushing the action button he wants to take , ha has to drag on the canvas to move the shape or resizing it then when the mouse is released the final shape will be the done one

**Undo & Redo** : the user only need to press this buttons to cancel or redoing he actions ha has taken before

***The idea of drawing with drag*** : is we have two canvases one for the drag and one to draw the final shape

When dragging the shape the first point the canvas gets will be the shape position and with dragging the canvas is cleared and measuring the size of the shape then its drown in the canvas ….. doing the drag more and more the more repeated those steps will be done until releasing

***Design patterns :***

***MVC*** *:* used to organize our code

**Package M :** for the module (the implementation)

**Package C** : for the controlling class

**Package V** : for the GUI design done by the FXML file and for the main

***Factory design pattern*** : used to create an instance from the shapes

When the user presses a button to draw a shape the factory class is called to create an object from the chosen shape

***The GUI snapshots :***

