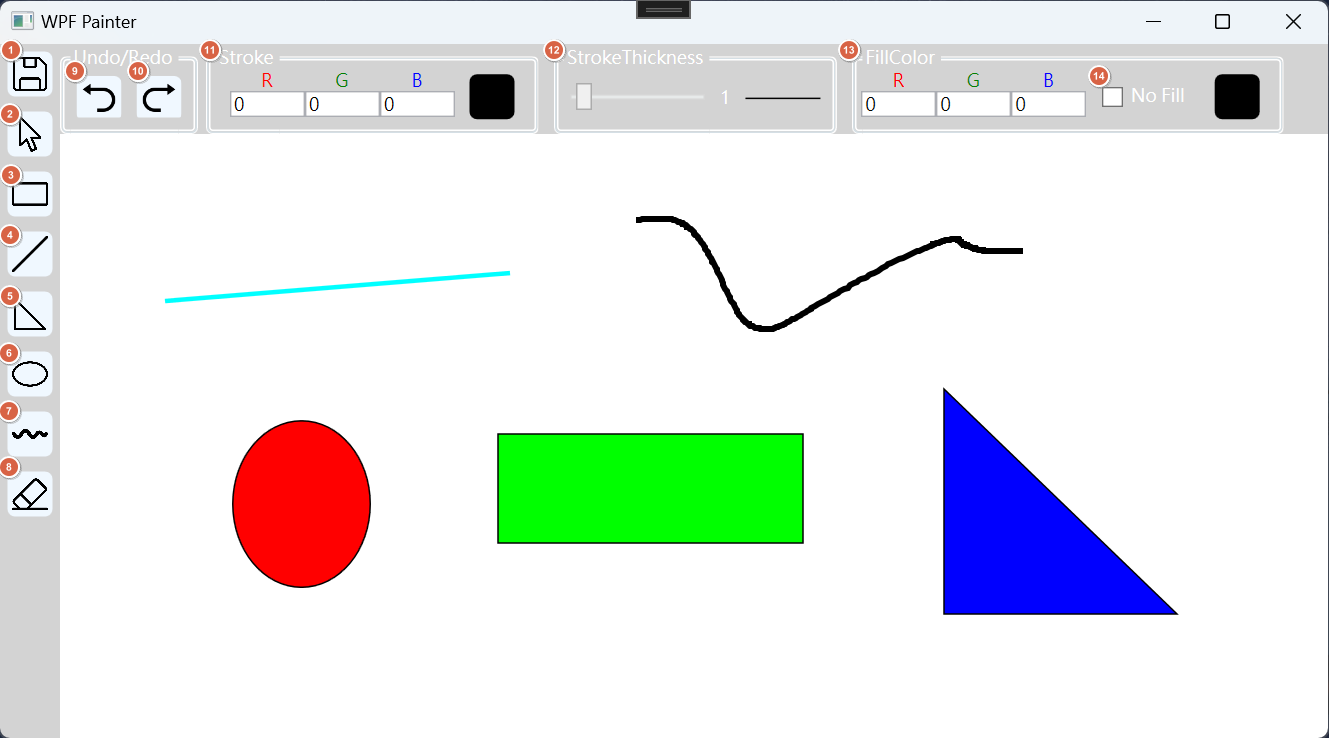
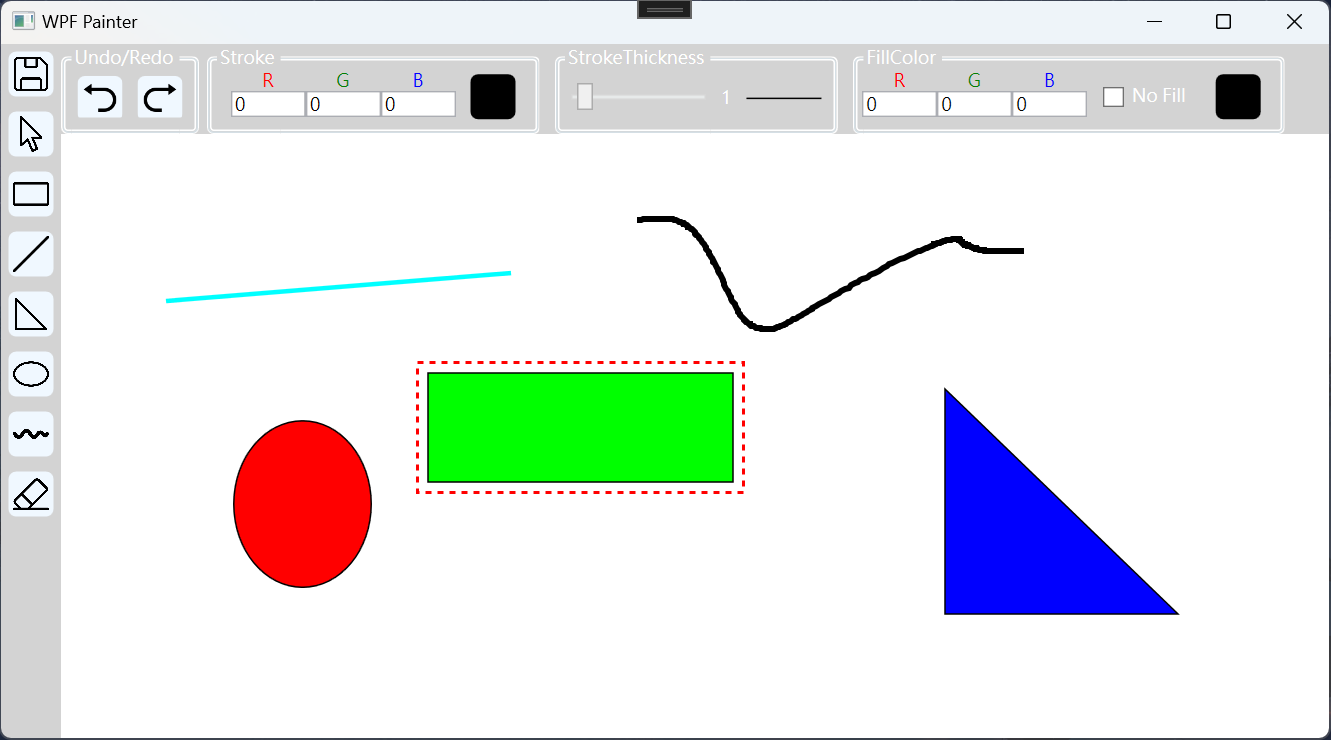
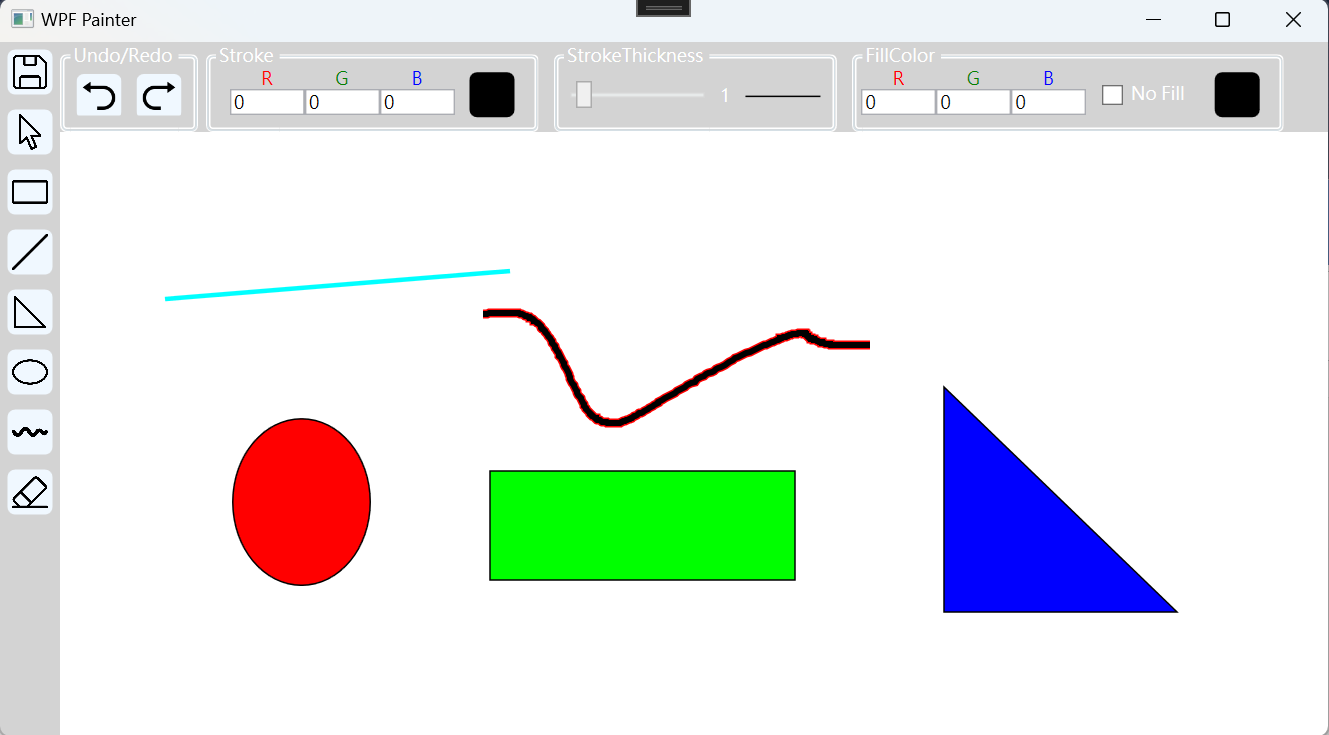
WPF Painter Function Explain

1. Save current drawn shapes (save to local drive is supported)
2. Select Mode
3. Drawing Mode for Rectangle
4. Drawing Mode for Line
5. Drawing Mode for Triangle
6. Drawing Mode for Ellipse
7. Drawing Mode for Brush
8. Erase Mode
9. Undo
10. Redo
11. You can change stroke color here
12. You can change stroke thickness here
13. You can change fill color here
14. If you don’t want any fill color (i.e. transparent), you can check this box

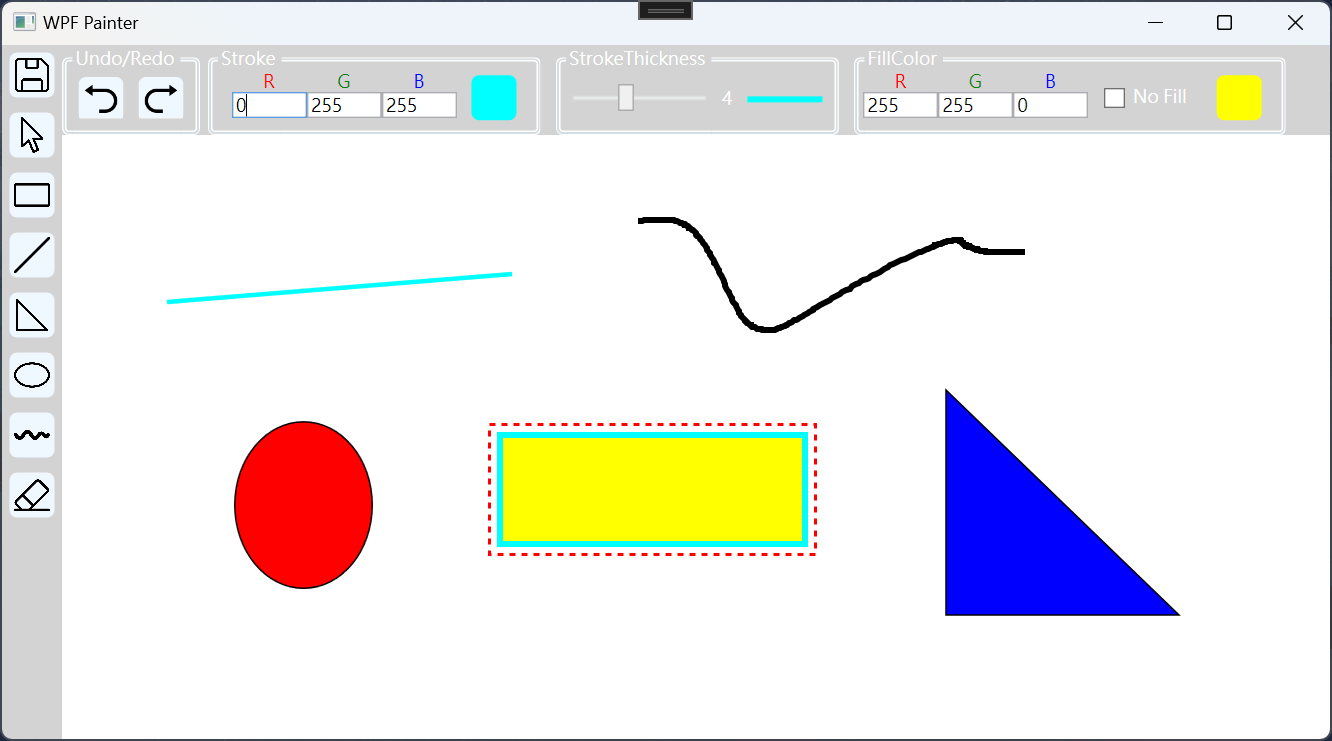


In Select Mode, you can move drawn shape freely.

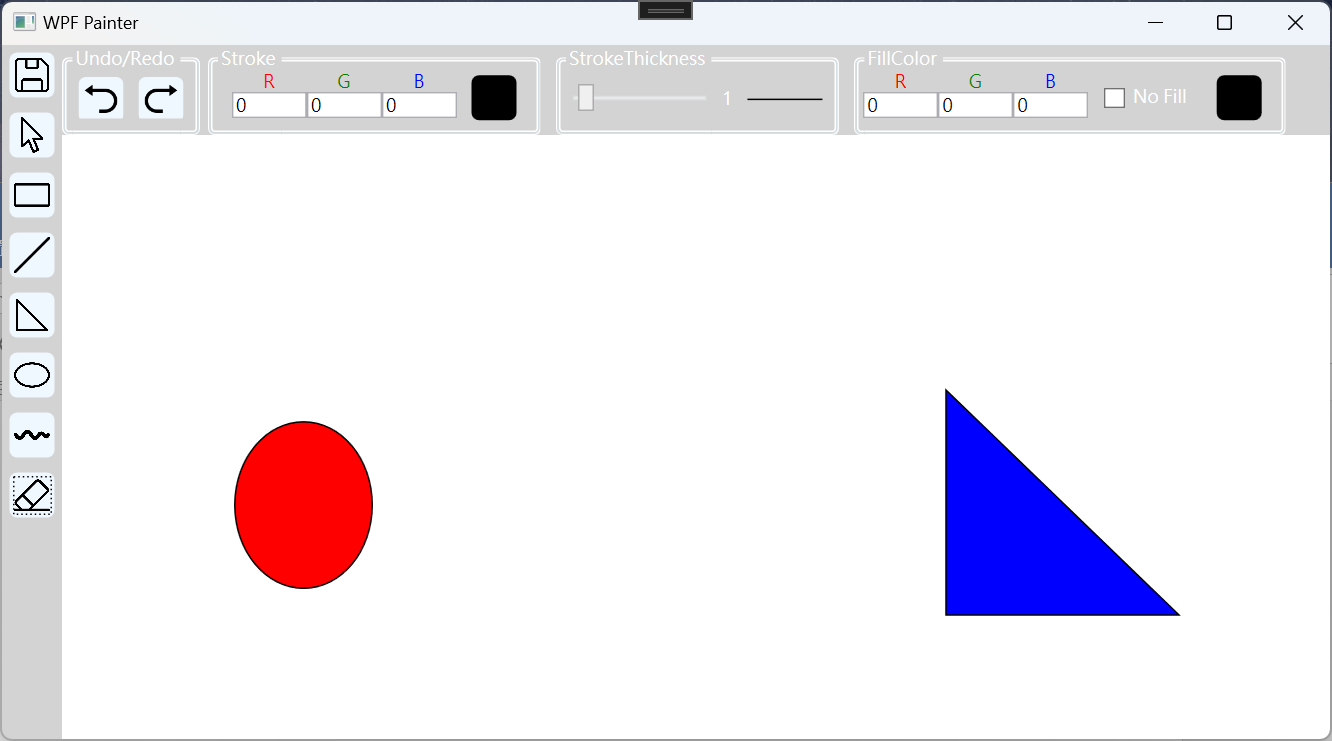




You can change properties of shape (stroke、strokethickness、color) by select it first, then you can change these properties using top panel.



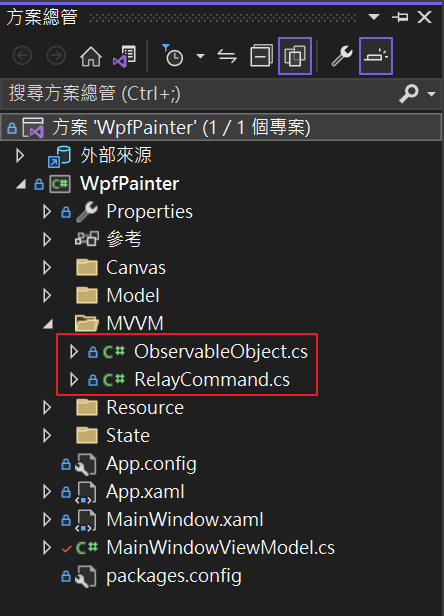
In Erase Mode, program will remove any shape that cursor is passed.



Architecture Explain

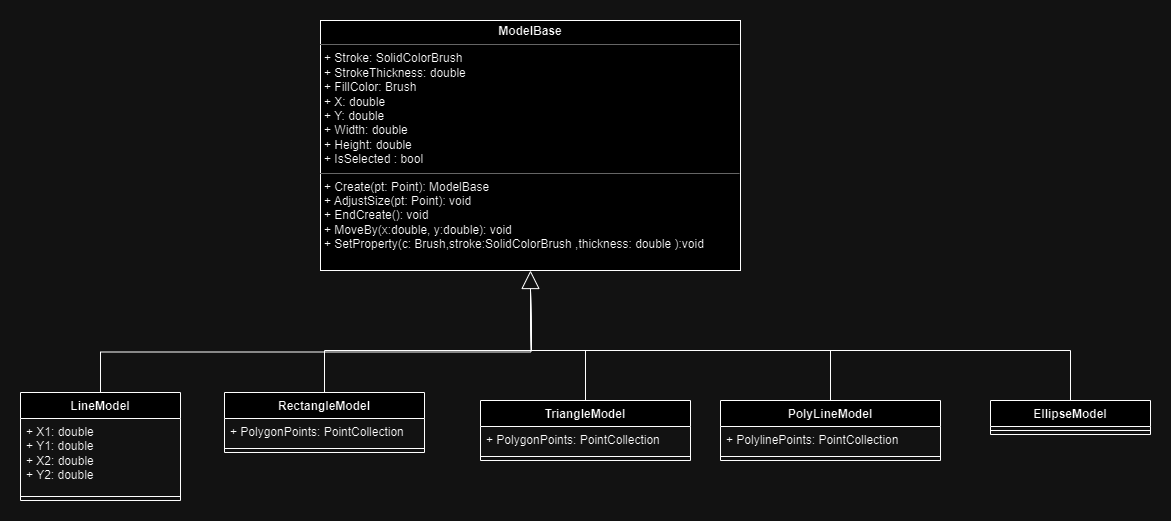
**MVVM**

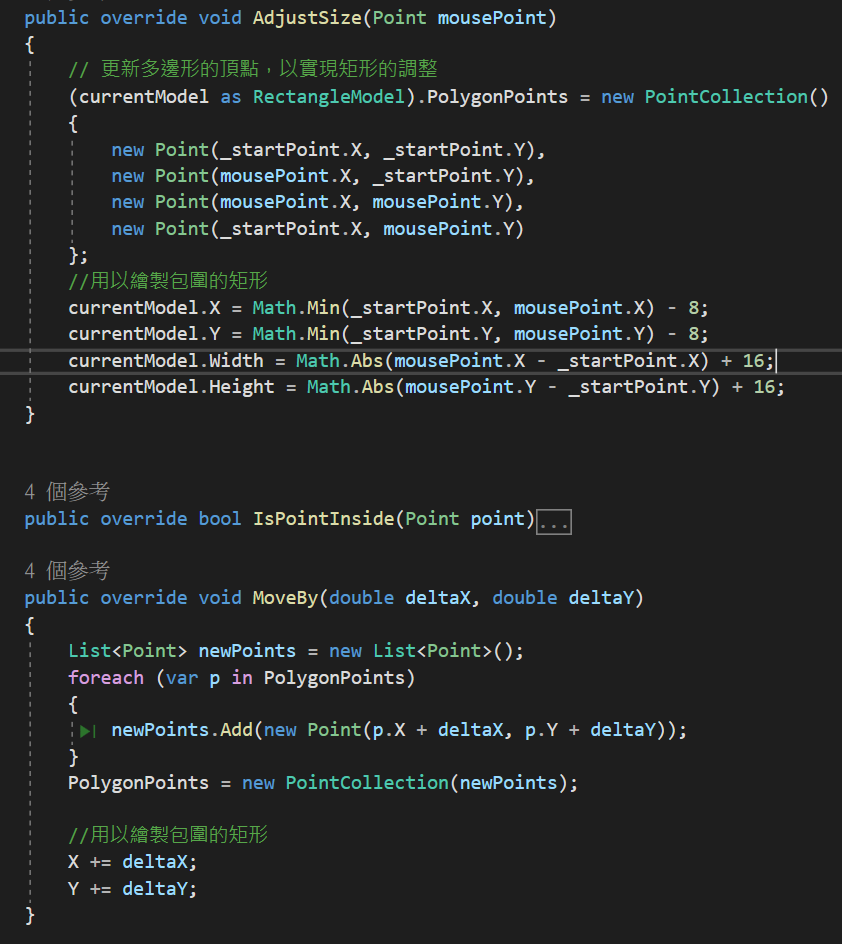
This program is implemented using MVVM architectural pattern, so I include ObservableObeject and RelayCommand these two class for basic binding and raising property change.

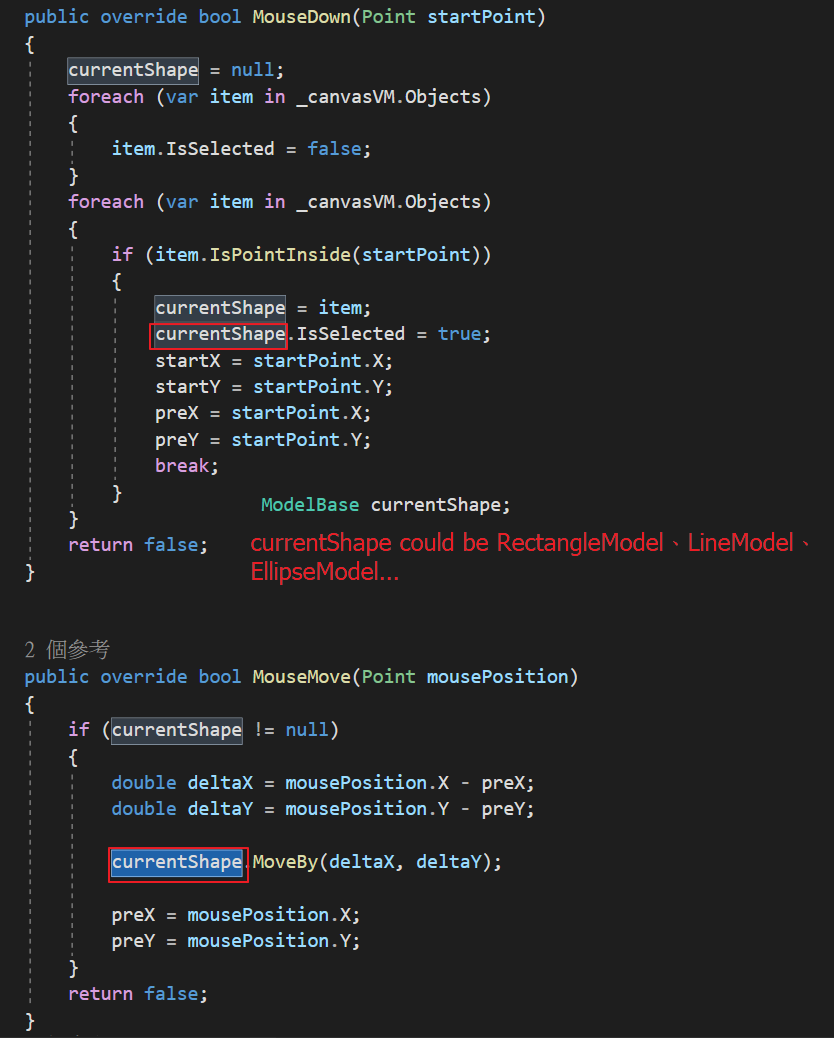


**Shape**

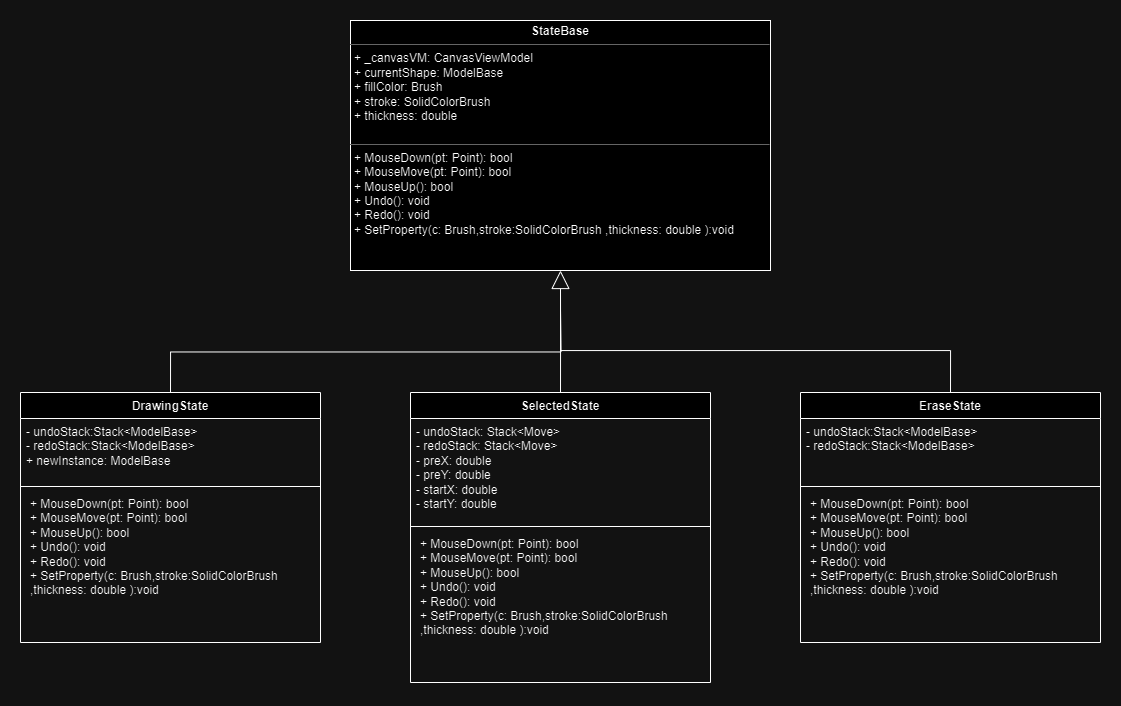
All shapes drawn on the Canvas inherit from the ModelBase class. ModelBase encompasses several attributes common to various shapes, such as width, height, fill color, border color, and border width. It also includes methods that are universally applicable to all shapes, such as Create, AdjustSize, MoveBy, and others.

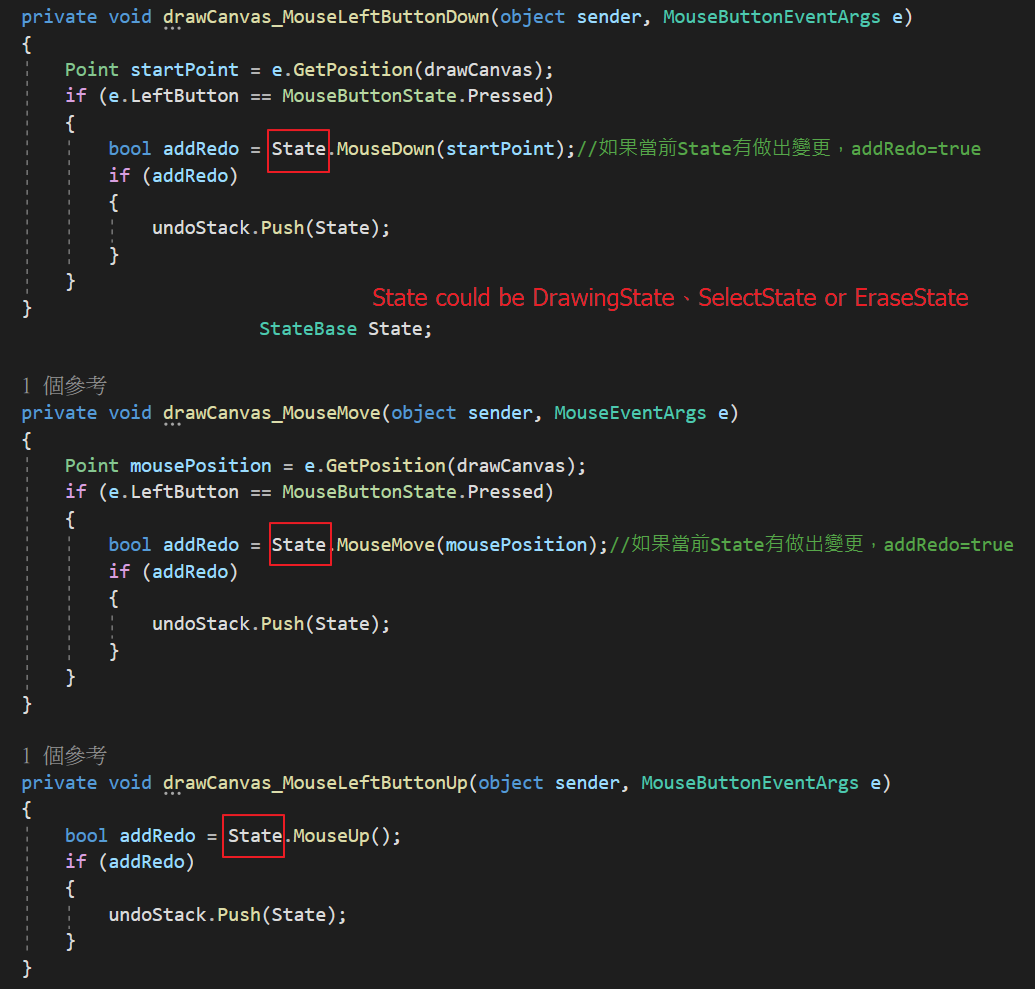
If child classes exhibit different behaviors from parent class, they will override corresponding methods. If there are properties required by specific child classes but not present in the parent class, these properties are added individually.

Taking RectangleModel as an example, the methods for dynamically adjusting width and height during creation, as well as updating internal points during movement (MoveBy), are as follows. Child classes, such as Ellipse, will implement and override these logic independently, as the calculation methods may differ.

For example in SelectState class, currentShape could be RectangleModel、LineModel、EllipseModel…，each shape has its own MoveBy method

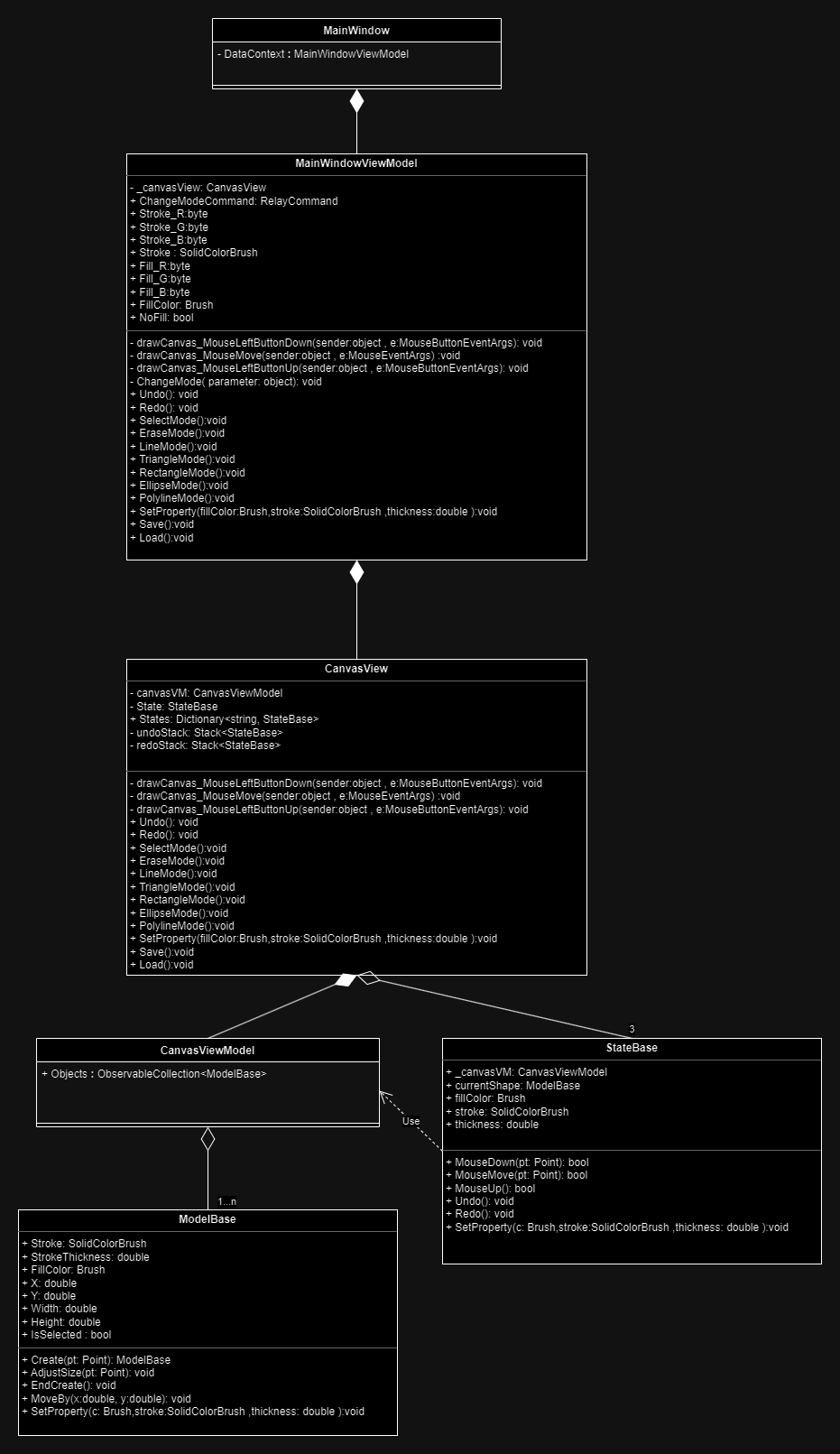
**State**

I have categorized the current states into three types: Drawing State, Select State, and Erase State. Each state has its own logic for handling mouse events when the left button is pressed, when the mouse is moved, and when the left button is released.



**Overview**

The overall structure of the program is shown below:



The primary interaction logic is implemented in the CanvasView, which is bind to the CanvasViewModel. The CanvasViewModel stores a collection of all shapes.

The CanvasView has three states (Drawing State, Select State, Erase State) responsible for drawing, moving, removing, and other functionalities. When these actions are performed, CanvasViewModel is modified accordingly.

Finally, CanvasView is injected into MainWindow. MainWindow is only responsible for handling events related to buttons, color adjustments, etc. MainWindow is bind to the MainWindowViewModel.

