National Taipei University of Technology

Windows Programming (Fall 2021)

Homework #7

Deadline: 12/27(Mon), before 24:00

壹、 注意

請遵守以下規則，否則單次成績以 0 分計算

1. 準時繳交作業，逾時不候
2. 不准抄襲他人作業，請自己完成

貳、 主題

根據上一次的作業內容，本次新增一個新的 Shape 為線以及 Command ， Redo 和 Undo 等功能，如圖 1 所示，Windows Store App 比照辦理。

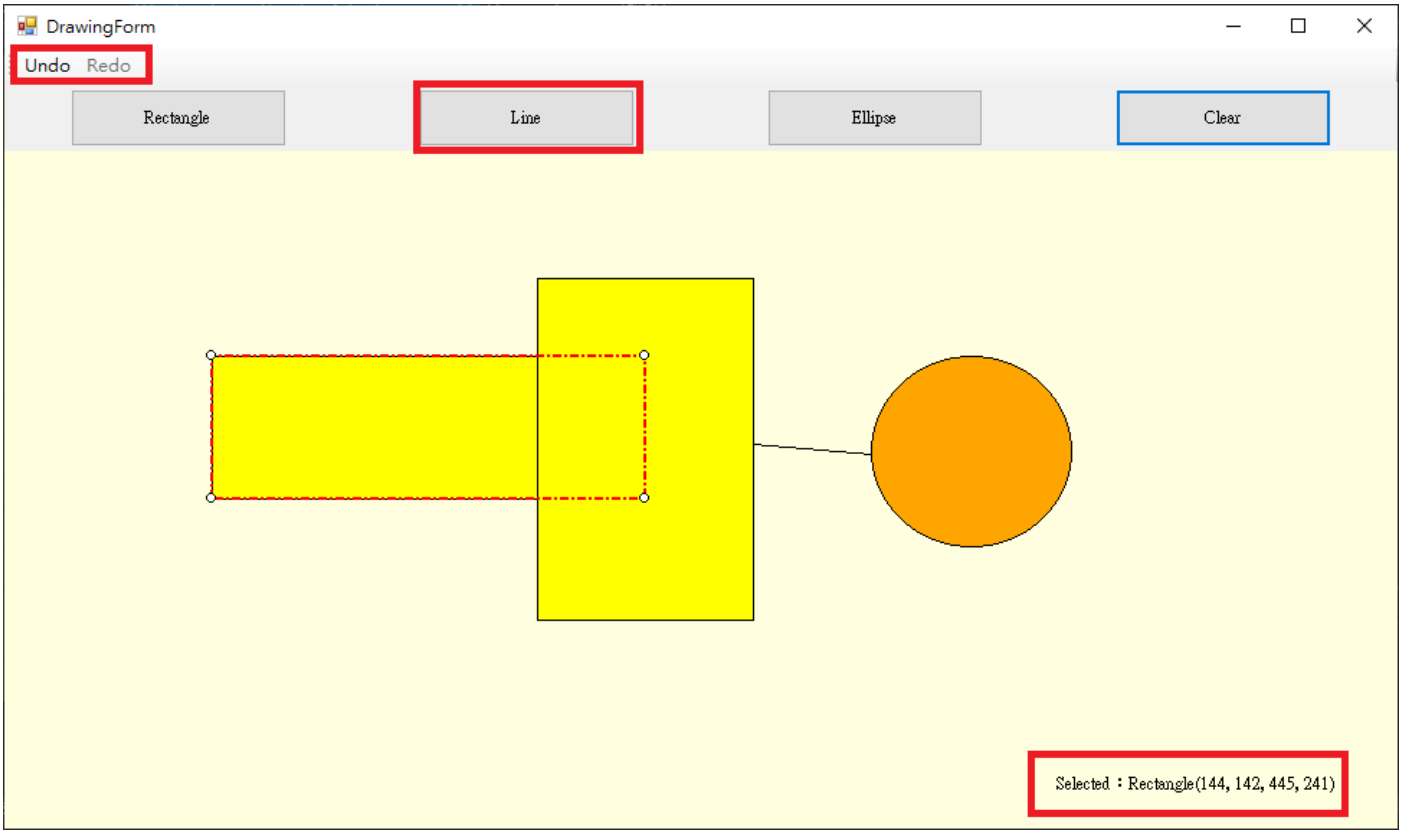

圖 1 Windows Form GUI

參、 題目介紹

一、[3 pts] Drawing Line

請在兩個 View 分別新增 Line Button，點擊 Line Button 時，你可以畫出線。注意：點擊任何畫圖按鈕時，須將其他畫圖按鈕設定為 enable，且將本身按鈕設定為 disable，例如點擊 Line 後，Rectangle 與 Ellipse 應該被 enable，而 Line 則被 disable。以下針對畫線功能做說明:

- Line 只能用來連接兩個圖形，不能單獨畫在畫布上，也不能與其他 Line 連接。
  - Line 的繪圖規則與其他圖形大致相同，但必須在某個已存在的圖形上按下滑鼠，才會開始進行繪製，在另一個圖形上釋放滑鼠後，再將 Line 儲存至 Model。Line 的兩個端點應為其連接圖形的中心點，如果在空白區域 (沒有其他圖形的地方) 按下滑鼠或釋放滑鼠，則放棄該 Line，但是仍然留在 Line 繪圖模式。請注意，未來(下一次作業)可能會有移動圖形的功能，使用者不能直接移動 Line，但是當 Line 的其中一個端點圖形被移動時，Line 也應該同時被移動，因此，實作 Line 功能時請避免直接在 Line 的 class 中儲存其端點圖形的座標，比較好的作法是在 Line 的 class 中儲存其兩個端點圖形的 reference，並在畫 Line 時(runtime)，取出端點圖形的座標，再依座標繪圖。

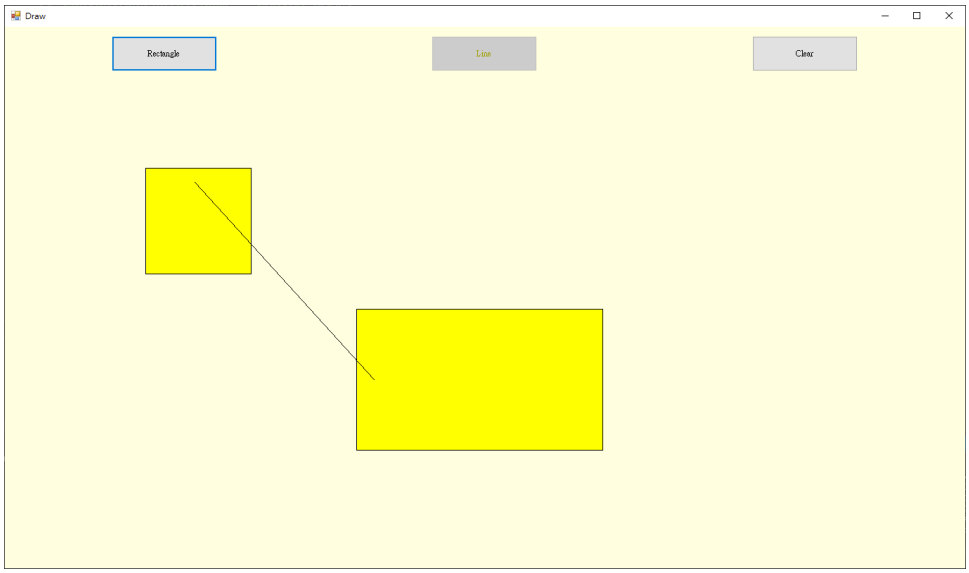

圖 2 畫線中

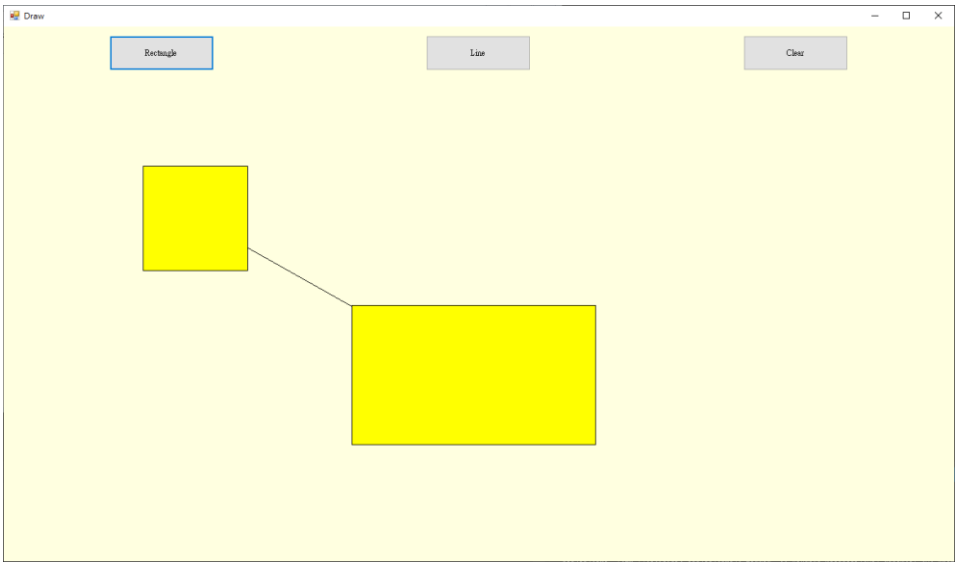

圖 3 成功畫線

- 我們希望使用者看到的 Line 畫在下層，也就是說 Line 不會蓋住其端點圖形，而是端點圖形蓋住 Line，如圖 4 所示，紅色虛線部分即為 Line 被其端點蓋過的部分。假設所有的圖形

(包含 Rectangle, Ellipse, Line 等)都儲存在同一個 List，則 OnPaint 時，可以分兩個 pass，第一個 pass 先繪出所有的 Line，第二個 pass 再繪出所有非 Line 的圖形，讓這些圖形蓋過 Line。請注意，你的實作應該充分運用多型(也就是呼叫 Shape class 的多型函數做運算或判斷)，避免直接依 instance 的 type 判斷是不是 Line(例如用 C#的 is 或 as 等 operator)。同理，在 Model 中，請避免儲存 2 個不同的 List (一個儲存 Line，另一個儲存非 Line 的圖形)，因為這樣做的話以後就不好擴充不同的 shape 了。

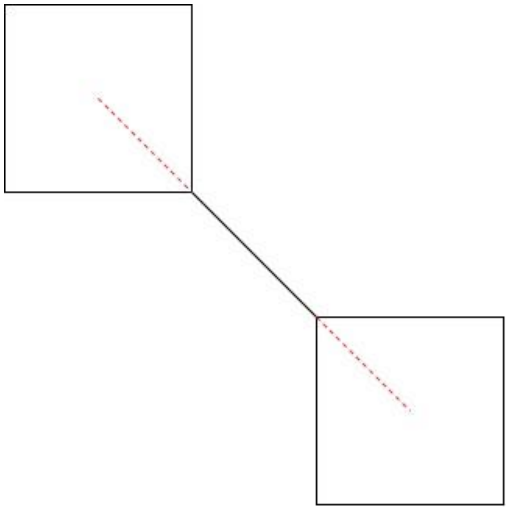

圖 4 Line 示意圖

二、[1 pts] Factory Pattern

在此作業中，可以在 Canvas 上繪製三種形狀（線、矩形和橢圓形）。當繪製形狀時，請使用簡單工廠(simple factory)模式來創建不同形狀的 instance。

三、[3 pts] 點選 Shape

使用者可以在畫面上點選 Shape。當使用者點擊某個形狀時，將選擇該形狀，被選擇到的形狀必須有虛線外框以及四個點。(未來被選擇到的形狀可能會新增不同行為，例如:放大、移動、刪除等等)。

- 虛線外框必須在最上層，不管點選的圖片是否為最上層，如圖 1。
- 畫面右下角新增一個 Label，點選到的圖形請在該 Label 上顯示被點選圖形之形狀及座標(Windows Form 與 Store 兩個視窗都必須顯示)，例如顯示 「Rectangle (1,1,2,2)，表示該形狀為矩形且其左上角座標為(1,1)、右下角 座標為(2,2)，請參考圖 1。

四、[8 pts] Undo and Redo

請使用 Command Pattern 來實作 redo 和 undo 功能，請參考圖 2。

- 如果是第一步，undo button 為 disable。
- 如果是最後一步，redo button 為 disable。
- 畫圖需要 redo、undo 功能，點選則不需 redo、undo。

五、[6 pts] Unit Testing

請為每個 Model class 的每個 method 撰寫 test case，且必須全部通過測試，Unit Test 是根據 test-coverage 去評分。

六、[3 pts] GUI Testing

請使用自動化 UI 測試為你的 GUI 編寫測試，測試案例(Test case)的規定如下：

- 只有 Windows Form 需要做 GUI Testing。

- 請將畫面上所有按鈕分別測試一次(畫出正方形、畫出橢圓形、畫出線、清空畫布)，畫出一個形狀後，請點選這個形狀，並 **Assert 畫面上的 Label 所顯示的值與所繪的圖形吻合** (assert 圖 1 的 Label 為正確的)。
- 請整合所有按鈕畫出一個具體的圖形，並執行適當的 Assert 後清除並關閉。

七、[2 pts] Observer (或 DataBinding)

請使用 C# 的 event-delegation model 或 observer pattern 來實現 MVC 架構。

八、[2 pts] Adaptor Pattern

在本作業中，您需要使用同一 Model 實現兩個不同的 view，而這兩個 view 的圖形界面並不相同，請使用 Adaptor Pattern 來解決該問題。

九、[3 pts] MVC Pattern

你的設計將根據 MVC 架構評分，讓 UI 盡可能薄，必須將 UI 以及 Model 切開，強制實行單向依賴，建議遵循圖 5 Class Diagram 設計。

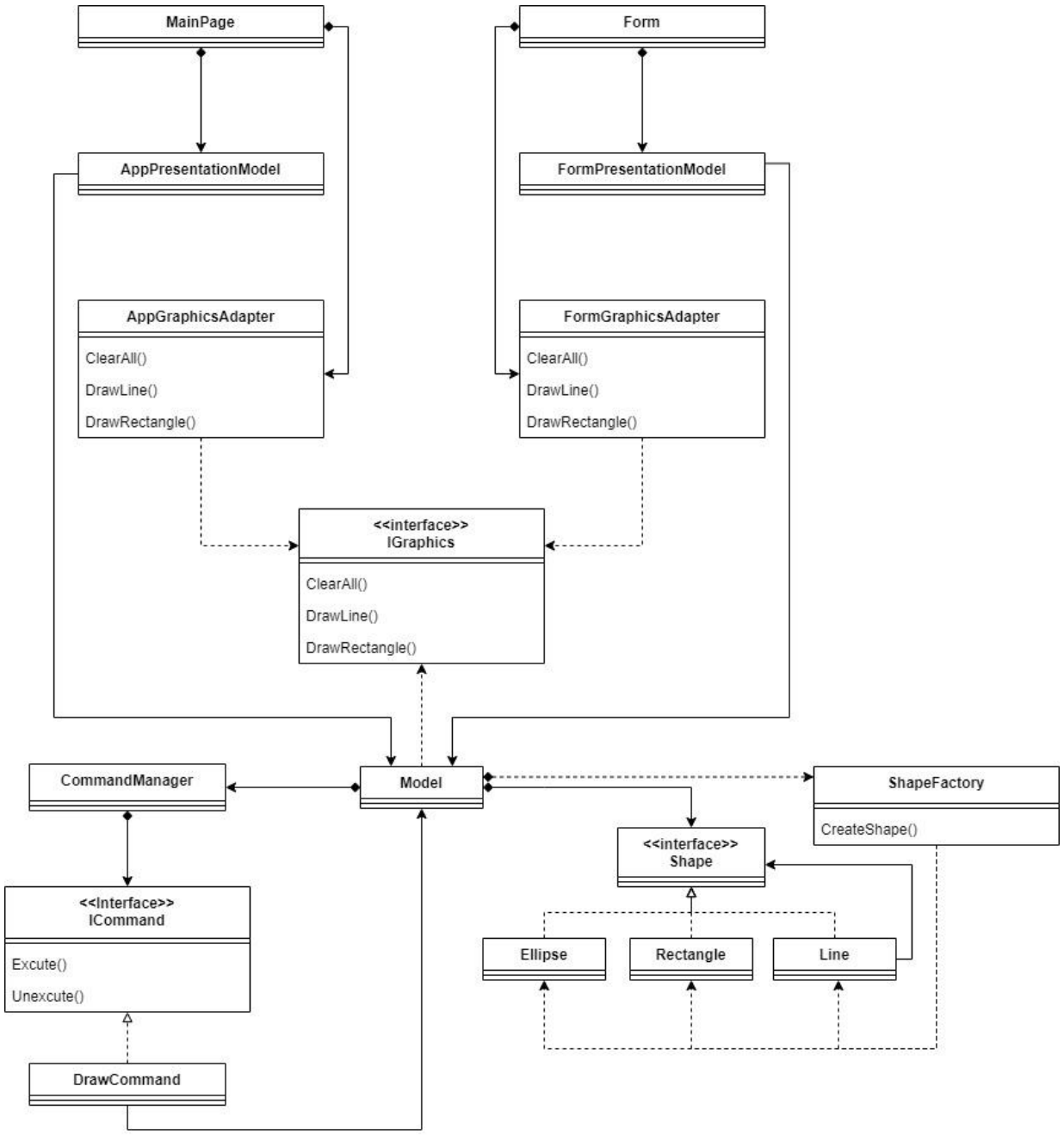

圖 5 Class Diagram

十、[9 pts] Code Quality

請使用 Dr.Smell 檢查你的程式，Code Quality 成績取決於你的程式碼是否有 bad smell，分數由你的 code smell 密度決定。請注意，當你使用 Windows Form Designer 產生程式碼時，其預設的變數命名、副程式命名等可能違反本課程之 coding standard，請修改以符合本課程之規定。

十一、 [2 pts] Summary

每一次寫完作業時都必須完成 homework summary，必須填寫本次作業花費時間，請在網站下載範本檔。

Additional Information

1. Drawing a rectangle in Windows Form – the following code can be used.

```
public DrawingForm()
{
    InitializeComponent();
    _canvas.AccessibleName = "_canvas";
    _canvas.Dock = DockStyle.Fill;
    _canvas.BackColor = System.Drawing.Color.LightYellow;
}
public void DrawRectangleRectangle(PaintEventArgs e)
{
    // Create pen.
    Pen blackPen = new Pen(Color.Black, 3);

    // Create rectangle.
    Rectangle rect = new Rectangle(0, 0, 200, 200);

    // Draw rectangle to screen.
    e.Graphics.DrawRectangle(blackPen, rect);
}
```

2. UI testing for drawing – add class Robot (the class can be downloaded from the instructor's web site).

   You can use the following code to do UI testing. You may also change the above class as you wish.

```
[TestMethod]
public void DrawingRectangle()
{
    // BUTTON_NAME 為畫圖按鈕，CANVAS_NAME 則為 canvas 的 AccessibleName
    _robot.ClickButton(BUTTON_NAME);
    _robot.DragAndDrop(CANVAS_NAME, 500, 400, 850, 750);
    …
}
```