# 統整小畫家筆記

## 一、專案定位與目的

本作業為 WPF Canvas 的繪圖工具示例，練習：

- WPF 的事件模型（MouseDown/Move/Up）
- 動態建立/更新 Shape（Rectangle, Ellipse, Line, Polyline）
- Canvas 的 Render → Image 儲存（RenderTargetBitmap）
- Undo/Redo 與橡皮擦的設計（使用 Stack 儲存操作）
  （原始程式碼與 XAML 片段已上傳，包含 ToolMode、Mouse 事件、SaveCanvas 實作）。

---

## 二、功能清單（User-visible features）

1. 選擇工具：矩形、橢圓、直線、自由筆（Polyline）、文字、橡皮擦、移動/選取（Select）。
2. 顏色/筆寬選擇、填充選項（Fill / Stroke）、線條樣式。
3. Undo / Redo（stack-based），刪除時能一次還原多元素（marker pattern）。
4. 儲存成 PNG/JPEG（RenderTargetBitmap + BitmapEncoder），並可改為輸出 XAML（vector）。

---

## 三、核心資料結構與物件模型

- ToolMode enum：代表目前使用工具（Select, Rectangle, Ellipse, Line, FreeDraw, Text, Eraser 等）。
- isDrawing: bool、currentShape: Shape、startPoint: Point：繪圖狀態追蹤，用以在 MouseMove 更新 shape 大小。
- Stack<UIElement> undoStack / redoStack：儲存操作以還原或重做。

刪除多元素時使用一個 marker 元件來包裝刪除清單，使 undo 可一次恢復多項。

---

## 四、重要事件/演算法詳解

### 1）繪圖流程（MouseLeftButtonDown / Move / Up）

- MouseLeftButtonDown：
  - 記錄 startPoint。根據 ToolMode 建立對應 Shape（例如 new Rectangle() => 設定 Stroke、Fill、LineWidth），將它 Canvas.Children.Add(currentShape)，並 CaptureMouse()。
- MouseMove（若 isDrawing）：
  - 計算寬高與 Canvas.Left/Top（以 startPoint 為基準），或更新 Line.X2/Y2、Polyline.Points.Add(currentPoint)。建議使用 Dispatcher 或 requestAnimationFrame 類型節流在大量事件時優化。
- MouseLeftButtonUp：
  - 解除 CaptureMouse()、isDrawing=false，並把 currentShape push 到 undoStack。若是文字或特殊物件，可能會包成 TextBox 再 finalize 為 TextBlock。

### 2）橡皮擦（Eraser）

- 原理：計算橡皮擦區域（矩形）→ 搜尋 Canvas.Children 中與該區域相交（IntersectsWith）的元素 → 將匹配元素移除並把被刪清單 push 到 undoStack（使用 marker 封包），以便 Undo。此方法能對 vector 元件精準刪除而非用 raster 覆蓋。

### 3）儲存（保存成影像）

- 使用 RenderTargetBitmap：先做 canvas.Measure、canvas.Arrange、canvas.UpdateLayout 以確保正確尺寸 → rtb.Render(canvas) → PngBitmapEncoder 或 JpegBitmapEncoder 寫出檔案。面試會問為何要 Measure/Arrange（因 WPF layout 可能未計算，render 出來會是空白

或不完整）。