```
<Application x:Class="_2024_WpfApp4.App"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:local="clr-namespace:_2024_WpfApp4"
             StartupUri="MainWindow.xaml">
    <Application.Resources>

    </Application.Resources>
</Application>

```

```
1  using System.Windows;
2  using System.Windows.Controls;
3  using System.Windows.Input;
4  using System.Windows.Media;
5  using System.Windows.Shapes;
6
7  namespace _2024_WpfApp4
8  {
9
10     public partial class MainWindow : Window
11     {
12         Point start = new Point { X = 0, Y = 0 };
13         Point dest = new Point { X = 0, Y = 0 };
14         Color strokeColor = Colors.Red;
15         Color fillColor = Colors.Aqua;
16         int strokeThickness = 1;
17         string shapeType = "line";
18         string actionType = "draw";
19
20         public MainWindow()
21         {
22             InitializeComponent();
23             strokeColorPicker.SelectedColor = strokeColor;
24             fillColorPicker.SelectedColor = fillColor;
25         }
26
27         private void MyCanvas_MouseEnter(object sender, MouseEventArgs e)
28         {
29             if (actionType == "erase") myCanvas.Cursor = Cursors.Hand;
30             else myCanvas.Cursor = Cursors.Pen;
31         }
32
33         private void MyCanvas_MouseLeftButtonDown(object sender,
             MouseButtonEventArgs e)
34         {
35             myCanvas.Cursor = Cursors.Cross;
36             start = e.GetPosition(myCanvas);
37
38             if (actionType == "draw")
39             {
40                 switch (shapeType)
41                 {
42                     case "line":
43                         Line line = new Line
44                         {
45                             X1 = start.X,
46                             Y1 = start.Y,
47                             X2 = dest.X,
48                             Y2 = dest.Y,
49                             StrokeThickness = 1,
50                             Stroke = Brushes.Gray
51                         };
52                         myCanvas.Children.Add(line);
```

```
53                          break;
54
55                      case "rectangle":
56                          Rectangle rect = new Rectangle
57                          {
58                              Stroke = Brushes.Gray,
59                              Fill = Brushes.LightGray
60                          };
61                          myCanvas.Children.Add(rect);
62                          rect.SetValue(Canvas.LeftProperty, start.X);
63                          rect.SetValue(Canvas.TopProperty, start.Y);
64                          break;
65
66                      case "ellipse":
67                          Ellipse ellipse = new Ellipse
68                          {
69                              Stroke = Brushes.Gray,
70                              Fill = Brushes.LightGray
71                          };
72                          myCanvas.Children.Add(ellipse);
73                          ellipse.SetValue(Canvas.LeftProperty, start.X);
74                          ellipse.SetValue(Canvas.TopProperty, start.Y);
75                          break;
76
77                      case "polyline":
78                          Polyline polyline = new Polyline
79                          {
80                              Stroke = Brushes.Gray,
81                              Fill = Brushes.LightGray
82                          };
83                          myCanvas.Children.Add(polyline);
84                          break;
85                  }
86              }
87
88              DisplayStatus();
89          }
90
91          private void MyCanvas_MouseMove(object sender, MouseEventArgs e)
92          {
93              dest = e.GetPosition(myCanvas);
94
95              switch (actionType)
96              {
97                  case "draw":
98                      if (e.LeftButton == MouseButtonState.Pressed)
99                      {
100                         Point origin;
101                         origin.X = Math.Min(start.X, dest.X);
102                         origin.Y = Math.Min(start.Y, dest.Y);
103                         double width = Math.Abs(start.X - dest.X);
104                         double height = Math.Abs(start.Y - dest.Y);
105
```

```
106                         switch (shapeType)
107                         {
108                             case "line":
109                                 var line = myCanvas.Children.OfType<Line>
                        ().LastOrDefault();
110                                 line.X2 = dest.X;
111                                 line.Y2 = dest.Y;
112                                 break;
113
114                             case "rectangle":
115                                 var rect = myCanvas.Children.OfType<Rectangle>
                        ().LastOrDefault();
116                                 rect.Width = width;
117                                 rect.Height = height;
118                                 rect.SetValue(Canvas.LeftProperty, origin.X);
119                                 rect.SetValue(Canvas.TopProperty, origin.Y);
120                                 break;
121
122                             case "ellipse":
123                                 var ellipse = myCanvas.Children.OfType<Ellipse>
                        ().LastOrDefault();
124                                 ellipse.Width = width;
125                                 ellipse.Height = height;
126                                 ellipse.SetValue(Canvas.LeftProperty, origin.X);
127                                 ellipse.SetValue(Canvas.TopProperty, origin.Y);
128                                 break;
129
130                             case "polyline":
131                                 var polyline = myCanvas.Children.OfType<Polyline>
                        ().LastOrDefault();
132                                 polyline.Points.Add(dest);
133                                 break;
134                         }
135                     }
136                     break;
137
138                 case "erase":
139                     var shape = e.OriginalSource as Shape;
140                     myCanvas.Children.Remove(shape);
141                     if (myCanvas.Children.Count == 0)
142                         myCanvas.Cursor = Cursors.Arrow;
143                     break;
144             }
145
146             DisplayStatus();
147         }
148
149         private void MyCanvas_MouseLeftButtonUp(object sender,
              MouseButtonEventArgs e)
150         {
151             Brush strokeBrush = new SolidColorBrush(strokeColor);
152             Brush fillBrush = new SolidColorBrush(fillColor);
153
```

```
154             switch (actionType)
155             {
156                 case "draw":
157                     switch (shapeType)
158                     {
159                         case "line":
160                             var line = myCanvas.Children.OfType<Line>
                     ().LastOrDefault();
161                             line.Stroke = strokeBrush;
162                             line.StrokeThickness = strokeThickness;
163                             break;
164 
165                         case "rectangle":
166                             var rect = myCanvas.Children.OfType<Rectangle>
                     ().LastOrDefault();
167                             rect.Stroke = strokeBrush;
168                             rect.Fill = fillBrush;
169                             rect.StrokeThickness = strokeThickness;
170                             break;
171 
172                         case "ellipse":
173                             var ellipse = myCanvas.Children.OfType<Ellipse>
                     ().LastOrDefault();
174                             ellipse.Stroke = strokeBrush;
175                             ellipse.Fill = fillBrush;
176                             ellipse.StrokeThickness = strokeThickness;
177                             break;
178 
179                         case "polyline":
180                             var polyline = myCanvas.Children.OfType<Polyline>
                     ().LastOrDefault();
181                             polyline.Stroke = strokeBrush;
182                             polyline.Fill = fillBrush;
183                             polyline.StrokeThickness = strokeThickness;
184                             break;
185                     }
186                     break;
187 
188                 case "erase":
189                     break;
190             }
191         }
192 
193         private void DisplayStatus()
194         {
195             pointLabel.Content = $"({Convert.ToInt32(start.X)}, {Convert.ToInt32
                (start.Y)}) -  ({Convert.ToInt32(dest.X)}, {Convert.ToInt32
                (dest.Y)})";
196             shapeLabel.Content = shapeType;
197             int lineCount = myCanvas.Children.OfType<Line>().Count();
198             int rectCount = myCanvas.Children.OfType<Rectangle>().Count();
199             int ellipseCount = myCanvas.Children.OfType<Ellipse>().Count();
200             int polylineCount = myCanvas.Children.OfType<Polyline>().Count();
```

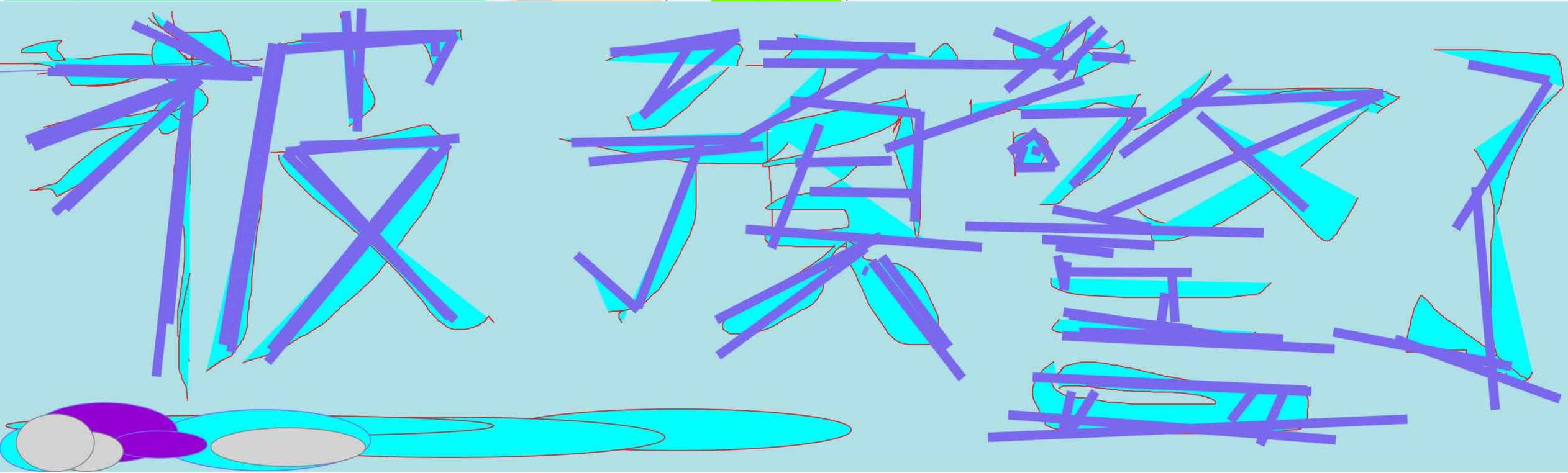
```
201
202             statusLabel.Content = $"工作模式：{actionType}, Line:{lineCount},
                  Rectangle:{rectCount}, Ellipse:{ellipseCount}, Polyline:
                  {polylineCount}";
203         }
204
205         private void StrokeThicknessSlider_ValueChanged(object sender,
              RoutedPropertyChangedEventArgs<double> e)
206         {
207             strokeThickness = Convert.ToInt32(strokeThicknessSlider.Value);
208         }
209
210         private void ShapeRadioButton_Checked(object sender, RoutedEventArgs e)
211         {
212             var targetRadioButton = sender as RadioButton;
213             shapeType = targetRadioButton.Tag.ToString();
214             actionType = "draw";
215             DisplayStatus();
216         }
217
218         private void StrokeColorPicker_SelectedColorChanged(object sender,
              RoutedPropertyChangedEventArgs<Color?> e)
219         {
220             strokeColor = strokeColorPicker.SelectedColor.Value;
221         }
222
223         private void FillColorPicker_SelectedColorChanged(object sender,
              RoutedPropertyChangedEventArgs<Color?> e)
224         {
225             fillColor = fillColorPicker.SelectedColor.Value;
226         }
227
228         private void ClearButton_Click(object sender, RoutedEventArgs e)
229         {
230             myCanvas.Children.Clear();
231             DisplayStatus();
232         }
233
234         private void EraseButton_Click(object sender, RoutedEventArgs e)
235         {
236             actionType = "erase";
237             if (myCanvas.Children.Count > 0)
238             {
239                 myCanvas.Cursor = Cursors.Hand;
240             }
241             DisplayStatus();
242         }
243     }
244 }
```

```
1  using System.Windows;
2
3  [assembly: ThemeInfo(
4      ResourceDictionaryLocation.None,            //where theme specific resource
       dictionaries are located
5                                                  //(used if a resource is not found
                       in the page,
6                                                  // or application resource
                       dictionaries)
7      ResourceDictionaryLocation.SourceAssembly   //where the generic resource
       dictionary is located
8                                                  //(used if a resource is not found
                       in the page,
9                                                  // app, or any theme specific
                       resource dictionaries)
10 )]
11
```

```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>WinExe</OutputType>
    <TargetFramework>net8.0-windows</TargetFramework>
    <RootNamespace>_2024_WpfApp4</RootNamespace>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
    <UseWPF>true</UseWPF>
  </PropertyGroup>

  <ItemGroup>
    <None Remove="ellipse.png" />
    <None Remove="eraser.png" />
    <None Remove="line.png" />
    <None Remove="polyline.png" />
    <None Remove="rectangle.png" />
    <None Remove="trashcan.png" />
  </ItemGroup>

  <ItemGroup>
    <PackageReference Include="Extended.Wpf.Toolkit" Version="4.6.1" />
  </ItemGroup>

  <ItemGroup>
    <Resource Include="ellipse.png" />
    <Resource Include="eraser.png" />
    <Resource Include="line.png" />
    <Resource Include="polyline.png" />
    <Resource Include="rectangle.png" />
    <Resource Include="trashcan.png" />
  </ItemGroup>

</Project>
```

檔案

筆刷色彩 M 填滿色彩 10 線 方 圓 接 擦 全部不要



line (811, 65) - (504, 499) 工作模式：draw, Line:95, Rectangle:0, Ellipse:17, Polyline:52

110 % 找不到任何問題 行: 11 字元: 14 SPC LF