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

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

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

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

```
202             int polylineCount = myCanvas.Children.OfType<Polyline>().Count();
203
204             statusLabel.Content = $"工作模式：{actionType}, Line:{lineCount},
                  Rectangle:{rectCount}, Ellipse:{ellipseCount}, Polyline:
                  {polylineCount}";
205         }
206
207         private void StrokeThicknessSlider_ValueChanged(object sender,
              RoutedPropertyChangedEventArgs<double> e)
208         {
209             strokeThickness = Convert.ToInt32(strokeThicknessSlider.Value);
210         }
211
212         private void ShapeRadioButton_Checked(object sender, RoutedEventArgs e)
213         {
214             var targetRadioButton = sender as RadioButton;
215             shapeType = targetRadioButton.Tag.ToString();
216             actionType = "draw";
217             DisplayStatus();
218         }
219
220         private void StrokeColorPicker_SelectedColorChanged(object sender,
              RoutedPropertyChangedEventArgs<Color?> e)
221         {
222             strokeColor = strokeColorPicker.SelectedColor.Value;
223         }
224
225         private void FillColorPicker_SelectedColorChanged(object sender,
              RoutedPropertyChangedEventArgs<Color?> e)
226         {
227             fillColor = fillColorPicker.SelectedColor.Value;
228         }
229
230         private void ClearButton_Click(object sender, RoutedEventArgs e)
231         {
232             myCanvas.Children.Clear();
233             DisplayStatus();
234         }
235
236         private void EraseButton_Click(object sender, RoutedEventArgs e)
237         {
238             actionType = "erase";
239             if (myCanvas.Children.Count > 0)
240             {
241                 myCanvas.Cursor = Cursors.Hand;
242             }
243             DisplayStatus();
244         }
245     }
246 }
```