```
// 定义编辑模式的枚举类型
enum EditMode
{
    SelectElement,    // 选择要素模式
    SelectVertex,     // 选择节点模式
    MoveVertex,       // 移动节点模式
    AddVertex,        // 新增节点模式
    DeleteVertex      // 删除节点模式
}

EditMode editMode = EditMode.SelectElement; // 设置默认的编辑模式

// 声明全局变量
public IMap m_pMap;                    // 地图对象
private IActiveView pActiveView;    // 活动视图

private IFeature selectedFeature;   // 选中的要素
private IPoint point;                 // 选中的点要素
private IPoint clickPoint;            // 鼠标点击的点
private IPoint startPoint;            // 鼠标按下时的起始点
private bool isDragging = false;   // 标识是否在拖动状态

private IPoint selectedVertex;      // 选中的顶点
private IPointCollection pointCollection; // 顶点集合
private int selectedVertexIndex = -1; // 选中的顶点索引
private IElement highlightElement; // 标记选中的顶点

public IWorkspaceEdit pWorkspaceEdit; // 当前编辑工作空间
public ILayer m_pCurrentLayer;            // 当前图层

private void btnLoadMxd_Click(object sender, EventArgs e)
{
    loadMapDocument();
}

// 加载地图文档的方法
private void loadMapDocument()
{
    // 创建文件对话框,用于选择地图文档文件
    OpenFileDialog openFileDialog = new OpenFileDialog();
    openFileDialog.Title = "打开地图文档";
```

```
        openFileDialog.Filter = "map documents(*.mxd)|*.mxd";

        if (openFileDialog.ShowDialog() == DialogResult.OK)
        {
            string filePath = openFileDialog.FileName; // 获取用户选择的文件路径

            // 检查文件是否为有效的地图文档
            if (axMapControl1.CheckMxFile(filePath))
            {
                axMapControl1.LoadMxFile(filePath, 0, Type.Missing); // 加载地图文件
            }
            else
            {
                MessageBox.Show(filePath + " 不是有效的地图文档");
            }
        }
    }

    // 点击选择按钮,启动要素选择功能
    private void btnSelectElement_Click(object sender, EventArgs e)
    {
        editMode = EditMode.SelectElement; // 设置为选择要素模式
        MessageBox.Show("当前模式: 选择要素");
    }

    // 点击选择顶点按钮,启动顶点选择功能
    private void btnSelectVertex_Click(object sender, EventArgs e)
    {
        editMode = EditMode.SelectVertex; // 设置为选择顶点模式
        MessageBox.Show("当前模式: 选择顶点");
    }

    // 点击移动顶点按钮,启动顶点移动功能
    private void btnMoveVertex_Click(object sender, EventArgs e)
    {
        editMode = EditMode.MoveVertex; // 设置为移动顶点模式
        MessageBox.Show("当前模式: 移动顶点");
    }

    // 点击增加顶点按钮,启动新增顶点功能
    private void btnAddVertex_Click(object sender, EventArgs e)
```

```
{
    editMode = EditMode.AddVertex; // 设置为新增顶点模式
    MessageBox.Show("当前模式: 新增顶点");
}

// 点击删除顶点按钮, 启动删除顶点功能
private void btnDeleteVertex_Click(object sender, EventArgs e)
{
    editMode = EditMode.DeleteVertex; // 设置为删除顶点模式
    DeleteVertex();
    ClearRedPoint();
    ReselectCurrentElement();
    MessageBox.Show("删除点要素成功");
}


// 根据不同的编辑模式处理鼠标点击事件
private void axMapControl1_OnMouseDown(object sender, IMapControlEvents2_OnMouseDownEvent e)
{
    m_pCurrentLayer = axMapControl1.get_Layer(0); // 获取第一个图层作为当前图层
    IFeatureLayer pFeatureLayer = (IFeatureLayer)m_pCurrentLayer;
    if (pFeatureLayer.FeatureClass == null) this.Close();
    IDataset pDataset = (IDataset)pFeatureLayer.FeatureClass;
    if (pDataset == null) this.Close();

    // 获取当前工作空间并开始编辑
    pWorkspaceEdit = (IWorkspaceEdit)pDataset.Workspace;
    if (!pWorkspaceEdit.IsBeingEdited())
    {
        pWorkspaceEdit.StartEditing(true);
        pWorkspaceEdit.EnableUndoRedo();
    }

    switch (editMode)
    {
        case EditMode.SelectElement:
            // 选择要素模式
            SelectElement(e);
            break;
        case EditMode.SelectVertex:
            // 选择顶点模式
```

```
                SelectVertex(e);
                break;
            case EditMode.MoveVertex:
                // 移动顶点模式
                MoveVertex(e);
                break;
            case EditMode.AddVertex:
                // 新增节点模式
                AddVertex(e);
                break;
            case EditMode.DeleteVertex:
                // 删除节点模式
                DeleteVertex();
                break;
        }
    }

    // 选择要素功能
    private void SelectElement(IMapControlEvents2_OnMouseDownEvent e)
    {
        IMap pMap = axMapControl1.Map;
        IActiveView pActiveView = (IActiveView)pMap;
        IEnvelope pEnv = axMapControl1.TrackRectangle(); // 获取矩形选择框
        pMap.SelectByShape(pEnv, null, false);                    // 根据矩形框选择要素
        pActiveView.PartialRefresh(esriViewDrawPhase.esriViewGeoSelection, null, null); // 局部刷新

        IEnumFeature enumFeature = (IEnumFeature)pMap.FeatureSelection;
        selectedFeature = enumFeature.Next();

        if (selectedFeature != null)
        {
            // 判断是否为点要素图层
            if (selectedFeature.Shape is IPoint)
            {
                MessageBox.Show("已选择点要素");
                point = (IPoint)selectedFeature.Shape;
                HighlightVertex(point);   // 高亮显示点要素
            }
            else if (selectedFeature.Shape is IPointCollection)
            {
                pointCollection = (IPointCollection)selectedFeature.Shape;
```

```
            }
        }
    }

    // 选择顶点功能
    private void SelectVertex(IMapControlEvents2_OnMouseDownEvent e)
    {
        clickPoint = axMapControl1.ActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(e.x, e.y);

        if (selectedFeature != null && selectedFeature.Shape is IPointCollection)
        {
            selectedVertexIndex = FindNearestVertexIndex(clickPoint, pointCollection);

            if (selectedVertexIndex != -1)
            {
                selectedVertex = pointCollection.get_Point(selectedVertexIndex);  // 获取最近顶点
                HighlightVertex(selectedVertex); // 高亮显示选中的顶点
            }
        }
    }

    // 新增顶点功能
    private void AddVertex(IMapControlEvents2_OnMouseDownEvent e)
    {
        if (pointCollection != null)
        {
            pWorkspaceEdit.StartEditOperation();  // 开始编辑操作
            clickPoint=axMapControl1.ActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(e.x, e.y);
            pointCollection.AddPoint(clickPoint); // 添加新顶点
            SaveFeature(); // 保存修改
            pWorkspaceEdit.StopEditOperation();  // 结束编辑操作
            RefreshMap();  // 刷新地图
        }
        else if (selectedFeature.Shape is IPoint)
        {
            MessageBox.Show("无法对单点要素执行新增节点操作");
        }
    }

    // 删除顶点功能
    private void DeleteVertex()
```

```
{
    if (selectedVertexIndex != - 1 && pointCollection != null)
    {
        pWorkspaceEdit.StartEditOperation();  // 开始编辑操作
        pointCollection.RemovePoints(selectedVertexIndex, 1); // 删除选中的顶点
        SaveFeature(); // 保存修改
        pWorkspaceEdit.StopEditOperation();   // 结束编辑操作
        selectedVertexIndex = - 1;
        selectedVertex = null;
        highlightElement = null;
        RefreshMap();  // 刷新地图
    }
    else if (selectedFeature.Shape is IPoint)
    {
        MessageBox.Show("无法对单点要素执行删除节点操作");
    }
}

// 移动顶点功能
private void MoveVertex(IMapControlEvents2_OnMouseDownEvent e)
{
    startPoint = axMapControl1.ActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(e.x, e.y);

    if (selectedVertex != null && pointCollection != null && selectedVertexIndex != - 1)
    {
        double distance = ((IProximityOperator)selectedVertex).ReturnDistance(startPoint);
        //if (distance < 1000) // 设置阈值
        //{
        // 开始移动顶点的编辑操作
        pWorkspaceEdit.StartEditOperation();
        isDragging = true;  // 开始拖动
        //}
    }
    else if (selectedFeature.Shape is IPoint)
    {
        MessageBox.Show("无法移动单点要素");
    }
}

// 鼠标移动事件,用于拖动顶点
private void axMapControl1_OnMouseMove(object sender, IMapControlEvents2_OnMouseMoveEvent e)
```

```
{
    if (isDragging && selectedVertex != null && pointCollection != null && selectedVertexIndex != - 1)
    {
        IActiveView pActiveView = axMapControl1.ActiveView;

        // 将屏幕坐标转换为地图坐标
        IPoint newPoint = pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(e.x, e.y);

        // 更新顶点位置
        selectedVertex.PutCoords(newPoint.X, newPoint.Y);
        pointCollection.UpdatePoint(selectedVertexIndex, selectedVertex);

        // 更新红色标记位置
        if (highlightElement != null)
        {
            highlightElement.Geometry = selectedVertex;
            axMapControl1.ActiveView.GraphicsContainer.UpdateElement(highlightElement);
        }

        // 局部刷新显示
        pActiveView.PartialRefresh(esriViewDrawPhase.esriViewGeography, null, null);
    }
}

// 鼠标抬起事件,用于保存编辑操作
private void axMapControl1_OnMouseUp(object sender, IMapControlEvents2_OnMouseUpEvent e)
{
    if (isDragging && pointCollection != null && selectedVertexIndex != - 1)
    {
        IMap pMap = axMapControl1.Map;
        IActiveView pActiveView = (IActiveView)pMap;

        IEnumFeature enumFeature = (IEnumFeature)pMap.FeatureSelection as IEnumFeature;
        IFeature selectedFeature = enumFeature.Next();

        if (selectedFeature != null)
        {
            // 更新要素形状并保存
            selectedFeature.Shape = (IGeometry)pointCollection;
            selectedFeature.Store();  // 保存修改
            // 刷新地图显示
```

```
                pActiveView.PartialRefresh(esriViewDrawPhase.esriViewGeography, null, null);
            }

            pWorkspaceEdit.StopEditOperation();    // 结束编辑操作

            // 停止拖动并重置状态
            isDragging = false;
            selectedVertex = null;
            selectedVertexIndex = -1;
            highlightElement = null;


            //删除红色标记
            ClearRedPoint();
            //刷新选中要素框
            ReselectCurrentElement();
        }
    }

    // 保存要素修改
    private void SaveFeature()
    {
        if (selectedFeature != null)
        {
            selectedFeature.Shape = (IGeometry)pointCollection;
            selectedFeature.Store();  // 保存修改
        }
    }

    // 刷新地图
    private void RefreshMap()
    {
        axMapControl1.ActiveView.PartialRefresh(esriViewDrawPhase.esriViewGeography, null, null);
    }

    // 高亮显示选中的顶点或点要素
    private void HighlightVertex(IPoint vertex)
    {
        IElement element = new MarkerElement();
        element.Geometry = vertex;
        IMarkerElement markerElement = (IMarkerElement)element;
```

```
        markerElement.Symbol = CreateRedMarkerSymbol();  // 使用红色标记

        IGraphicsContainer graphicsContainer = axMapControl1.ActiveView.GraphicsContainer;

        // 如果已经存在高亮标记,先删除它
        if (highlightElement != null)
        {
            graphicsContainer.DeleteElement(highlightElement);
        }

        // 刷新选中框(先刷新选中框,再添加红点)
        axMapControl1.ActiveView.PartialRefresh(esriViewDrawPhase.esriViewGeoSelection, null, null);

        // 添加红点标记元素
        graphicsContainer.AddElement(element, 0);

        // 保存当前红点标记
        highlightElement = element;

        // 使用 esriViewGraphics 刷新红点,确保红点在最顶层
        axMapControl1.ActiveView.PartialRefresh(esriViewDrawPhase.esriViewGraphics, null, null);
    }

    // 根据点击点寻找最近的顶点索引
    private int FindNearestVertexIndex(IPoint clickPoint, IPointCollection pointCollection)
    {
        int nearestIndex = -1;
        double minDistance = double.MaxValue;

        for (int i = 0; i < pointCollection.PointCount; i++)
        {
            IPoint vertex = pointCollection.get_Point(i);
            double distance = ((IProximityOperator)vertex).ReturnDistance(clickPoint);
            if (distance < minDistance)
            {
                minDistance = distance;
                nearestIndex = i;
            }
        }
        return nearestIndex;
    }
```

```
// 创建红色标记符号,用于高亮显示顶点
private ISimpleMarkerSymbol CreateRedMarkerSymbol()
{
    ISimpleMarkerSymbol markerSymbol = new SimpleMarkerSymbol();
    markerSymbol.Style = esriSimpleMarkerStyle.esriSMSCircle;
    markerSymbol.Size = 7;
    markerSymbol.Color = GetRgbColor(255, 0, 0);
    return markerSymbol;
}

// 清除红点标记的方法
private void ClearRedPoint()
{
    // 获取地图的图形容器
    IGraphicsContainer graphicsContainer = axMapControl1.ActiveView.GraphicsContainer;

    // 删除所有红色标记元素
    graphicsContainer.DeleteAllElements();

    // 刷新地图显示以移除红点
    axMapControl1.ActiveView.PartialRefresh(esriViewDrawPhase.esriViewGraphics, null, null);

    // 重置红点相关变量
    highlightElement = null;
    selectedVertex = null;
    selectedVertexIndex = -1;
}

// 重新选择并刷新当前选中要素的方法
private void ReselectCurrentElement()
{
    // 取消当前地图的所有选择
    axMapControl1.Map.ClearSelection();

    // 检查当前选中的要素是否存在
    if (selectedFeature != null)
    {
        // 重新选择当前要素
        axMapControl1.Map.SelectFeature(m_pCurrentLayer, selectedFeature);
```

```
            // 刷新地图以更新选中状态
            axMapControl1.ActiveView.PartialRefresh(esriViewDrawPhase.esriViewGeoSelection, null, null);
        }
    }

    // 返回 RGB 颜色对象
    private IRgbColor GetRgbColor(int r, int g, int b)
    {
        IRgbColor rgbColor = new RgbColor();
        rgbColor.Red = r;
        rgbColor.Green = g;
        rgbColor.Blue = b;
        return rgbColor;
    }
```