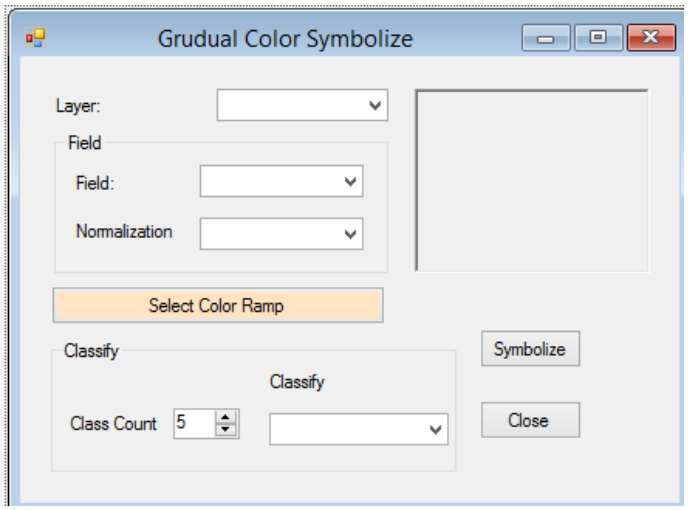
Chapter 14 Gradual Color Symbolize

1. Add a Windows Form “GraduatedColorSymbolize” to the project:



GUI implementation detail could be found at “GraduatedColorSymbolize.Designer.cs” file.

2. Implementation of GraduatedColorSymbolize:

- Class member and Construction method:

```
IHookHelper m_hookHelper = null;
IActiveView m_activeView = null;
IMap m_map = null;
1 reference | Yuhui Wu, 14 hours ago | 1 change
public GraduatedColorSymbolize(IHookHelper hookHelper)
{
    InitializeComponent();
    m_hookHelper = hookHelper;
    m_activeView = m_hookHelper.ActiveView;
    m_map = m_hookHelper.FocusMap;
}
```

- Add GetLayers, `CbxLayersAddItems`, GetFeatureLayer, ,CbxFieldAddotems, Load event of form, Click Event of “Cancel”:

```
private IEnumLayer GetLayers()
{
    UID uid = new UIDClass();
    uid.Value = "{40A9E885-5533-11d0-98BE-00805F7CED21}";
    if (m_map.LayerCount != 0)
    {
        IEnumLayer layers = m_map.get_Layers(uid, true);
        return layers;
    }
    return null;
}
```

```
private void CbxLayersAddItems()
{
    if (GetLayers() == null) return;
    IEnumLayer layers = GetLayers();
    layers.Reset();
    ILayer layer = layers.Next();
    while (layer != null)
    {
        if (layer is IFeatureLayer)
        {
            cbxLayers2Symbolize.Items.Add(layer.Name);
        }
        layer = layers.Next();
    }
}
```

```
private IFeatureLayer GetFeatureLayer(string layerName)
{
    if (GetLayers() == null) return null;
    IEnumLayer layers = GetLayers();
    layers.Reset();

    ILayer layer = null;
    while ((layer = layers.Next()) != null)
    {
        if (layer.Name == layerName)
            return layer as IFeatureLayer;
    }
    return null;
}
```

```
private void CbxFieldAdditems(IFeatureLayer featureLayer)
{
    IFields fields = featureLayer.FeatureClass.Fields;
    cbxFields.Items.Clear();
    cbxNormalization.Items.Clear();
    cbxNormalization.Items.Add("None");
    for (int i = 0; i < fields.FieldCount; i++)
    {
        if ((fields.get_Field(i).Type == esriFieldType.esriFieldTypeDouble) ||
            (fields.get_Field(i).Type == esriFieldType.esriFieldTypeInteger) ||
            (fields.get_Field(i).Type == esriFieldType.esriFieldTypeSingle) ||
            (fields.get_Field(i).Type == esriFieldType.esriFieldTypeSmallInteger))
        {
            cbxFields.Items.Add(fields.get_Field(i).Name);
            cbxNormalization.Items.Add(fields.get_Field(i).Name);
        }
    }
    cbxFields.SelectedIndex = 0;
    cbxNormalization.SelectedIndex = 0;
}
```

```
private void GraduatedColorSymbolize_Load(object sender, EventArgs e)
{
    CbxLayersAddItems();
}
```

```
private void btnCancel_Click(object sender, EventArgs e)
{
    this.Close();
}
```

- selectedIndexChanged event of cbxLayers2Symbolize , cbxFields, and : cbxNormalization

```
IFeatureLayer layer2Symbolize = null;
string strRendererField = string.Empty;
1 reference | Yuhui Wu, 15 hours ago | 1 change
private void cbxLayers2Symbolize_SelectedIndexChanged(object sender, EventArgs e)
{
    if (cbxLayers2Symbolize.SelectedItem != null)
    {
        string strLayer2Symbolize = cbxLayers2Symbolize.SelectedItem.ToString();
        layer2Symbolize = GetFeatureLayer(strLayer2Symbolize);
        CbxFieldAdditems(layer2Symbolize);
        strRendererField = cbxFields.Items[0].ToString();
    }
}
1 reference | Yuhui Wu, 15 hours ago | 1 change
private void cbxFields_SelectedIndexChanged(object sender, EventArgs e)
{
    if (cbxFields.SelectedItem != null)
    {
        strRendererField = cbxFields.SelectedItem.ToString();
    }
}
string strNormalizeField = string.Empty;
1 reference | Yuhui Wu, 15 hours ago | 1 change
private void cbxNormalization_SelectedIndexChanged(object sender, EventArgs e)
{
    if (cbxNormalization.SelectedItem != null)
    {
        strNormalizeField = cbxNormalization.SelectedItem.ToString();
    }
}
```

- Click Event of “Select Color Ramp”:

```
IColorRamp colorRamp = null;
1 reference | Yuhui Wu, 15 hours ago | 1 change
private void btnSelectColorRamp_Click(object sender, EventArgs e)
{
    GetSymbol symbolForm = new GetSymbol(esriSymbologyStyleClass.esriStyleClassColorRamps);
    symbolForm.ShowDialog();
    IStyleGalleryItem styleGalleryItem = symbolForm.m_styleGalleryItem;
    if (styleGalleryItem == null)
        return;
    colorRamp = styleGalleryItem.Item as IColorRamp;
    symbolForm.Dispose();
}
```

- ValueChanged event of nudClassCount:

```
int gClassCount = 5;

1 reference | Yuhui Wu, 15 hours ago | 1 change
private void nudClassCount_ValueChanged(object sender, EventArgs e)
{
    gClassCount = Convert.ToInt32(nudClassCount.Value);
}
```

- SelectedIndexChanged Event of classifyCBX:

```
string strClassifyMethod = "Natural Breaks";
1 reference | Yuhui Wu, 15 hours ago | 1 change
private void classifyCBX_SelectedIndexChanged(object sender, EventArgs e)
{
    if (classifyCBX.SelectedItem != null)
    {
        strClassifyMethod = classifyCBX.SelectedItem.ToString();
    }
}
double[] gClassbreaks = null;
```

- classify, CreateClassBreaksRenderer and Renderer

```
double[] gClassbreaks = null;
private void classify()
{
    if (layer2Symbolize == null) return;
    IFeatureClass featureClass = layer2Symbolize.FeatureClass;
    ITable pTable = featureClass as ITable;
    ITableHistogram pTableHistogram = new BasicTableHistogramClass();
    IBasicHistogram pHistogram = pTableHistogram as IBasicHistogram;
    pTableHistogram.Field = strRendererField;
    if (strNormalizeField.ToLower() != "none")
        pTableHistogram.NormField = strNormalizeField;
    pTableHistogram.Table = pTable;
    object dataFrequency;
    object dataValues;
    pHistogram.GetHistogram(out dataValues, out dataFrequency);
    IClassifyGEN pClassify = new NaturalBreaksClass();
    switch (strClassifyMethod)
    {
        case "Equal Interval":
            pClassify = new EqualIntervalClass();
            break;
        case "Quantile":
            pClassify = new QuantileClass();
            break;
        case "Natural Breaks":
            pClassify = new NaturalBreaksClass();
            break;
        case "Geometrical Interval":
            pClassify = new GeometricalIntervalClass();
            break;
        default:
            break;
    }
    int numDesiredClass = gClassCount;
    pClassify.Classify(dataValues, dataFrequency, ref numDesiredClass);
    gClassbreaks = (double[])pClassify.ClassBreaks;
}
IClassBreaksRenderer m_classBreaksRenderer = null;
private IClassBreaksRenderer CreateClassBreaksRenderer
    (IFeatureClass featureClass)
{
    if (colorRamp == null)
    {
        MessageBox.Show("Choose Ramp",
```

```
                "Message", MessageBoxButtons.OK, MessageBoxIcon.Warning);
            return null;
        }
        classify();
        int ClassesCount = gClassbreaks.GetUpperBound(0);
        if (ClassesCount == 0) return null;
        nudClassCount.Value = ClassesCount;
        IClassBreaksRenderer pClassBreaksRenderer =
            new ClassBreaksRendererClass();
        pClassBreaksRenderer.Field = strRendererField;
        if (strNormalizeField.ToLower() != "none")
            pClassBreaksRenderer.NormField = strNormalizeField;
        pClassBreaksRenderer.BreakCount = ClassesCount;
        pClassBreaksRenderer.SortClassesAscending = true;

        colorRamp.Size = ClassesCount;
        bool createRamp;
        colorRamp.CreateRamp(out createRamp);
        IEnumColors enumColors = colorRamp.Colors;
        enumColors.Reset();
        IColor pColor = null;
        ISymbol symbol = null;
        for (int i = 0; i < ClassesCount; i++)
        {
            pColor = enumColors.Next();
            switch (featureClass.ShapeType)
            {
                case esriGeometryType.esriGeometryPoint:
                    ISimpleMarkerSymbol simpleMarkerSymbol =
                        new SimpleMarkerSymbolClass();
                    simpleMarkerSymbol.Color = pColor;
                    symbol = simpleMarkerSymbol as ISymbol;
                    break;
                case esriGeometryType.esriGeometryPolyline:
                    ISimpleLineSymbol simpleLineSymbol = new SimpleLineSymbolClass();
                    simpleLineSymbol.Color = pColor;
                    symbol = simpleLineSymbol as ISymbol;
                    break;
                case esriGeometryType.esriGeometryPolygon:
                    ISimpleFillSymbol simpleFillSymbol = new SimpleFillSymbolClass();
                    simpleFillSymbol.Color = pColor;
                    symbol = simpleFillSymbol as ISymbol;
                    break;
                default:
                    break;
            }
            pClassBreaksRenderer.set_Symbol(i, symbol);
            pClassBreaksRenderer.set_Break(i, gClassbreaks[i + 1]);
        }
        return pClassBreaksRenderer;
    }
    private void Renderer()
    {
        IGeoFeatureLayer pGeoFeatureL = (IGeoFeatureLayer)layer2Symbolize;
        IFeatureClass featureClass = pGeoFeatureL.FeatureClass;
        int lfieldNumber = featureClass.FindField(strRendererField);
        if (lfieldNumber == -1)
        {
```

```
                MessageBox.Show("Can't find field called " + strRendererField);
                return;
            }
            m_classBreaksRenderer = CreateClassBreaksRenderer(featureClass);
            if (m_classBreaksRenderer == null) return;
            pGeoFeatureL.Renderer = (IFeatureRenderer)m_classBreaksRenderer;
            m_activeView.PartialRefresh(esriViewDrawPhase.esriViewGeography,
                null, m_activeView.Extent);
        }
```

- Click Event of “Symbolize”:

```
private void btnSymbolize_Click(object sender, EventArgs e)
{
    if (layer2Symbolize == null)
        return;
    Renderer();
}
```

3. Add A Base Command Class GraduatedColorSymbolsCmd, and implement OnClick method:

```
public override void OnClick()
{
    // TODO: Add GraduatedColorSymbolsCmd.OnClick implementation
    if (m_hookHelper == null) return;
    if (m_hookHelper.FocusMap.LayerCount > 0)
    {
        GraduatedColorSymbolize symbol =
            new GraduatedColorSymbolize(m_hookHelper);
        symbol.Show(m_hookHelper as
            System.Windows.Forms.IWin32Window);
    }
}
```

4. Back to MainForm, add a menu content and its click event:

```
private void graduatedColorSymbolizeToolStripMenuItem_Click(object sender, EventArgs e)
{
    ICommand command = new GraduatedColorSymbolsCmd();
    command.OnCreate(axMapControl1.Object);
    command.OnClick();
}
```