第2章

ArcGIS Engine中的控件

为了快速构建一个 GIS 应用程序，ArcGIS Engine 给开发者提供了一些可视化控件，如制图控件、3D 控件、框架控件等。ArcGIS 控件可以通过两种方式建立应用程序，其一，ArcGIS 控件可以嵌入到现有的应用程序中以增强制图功能；其二，ArcGIS 控件可用于创建新的独立应用程序。

制图控件，如 MapControl、PageLayoutControl，其中 MapControl 控件主要用于地理数据的显示和分析，PageLayoutControl 用于生成一幅成品地图。MapControl 封装了 Map 对象，而 PageLayoutControl 则封装了 PageLayout 对象。这两个控件都实现了 IMxContents 接口，因此不仅可以读取 ArcMap 创建的地图文档，而且可以将自身的地图内容写到一个新的地图文档中。

三维控件，如 GlobeControl、SceneControl 都具有导航功能，允许终端用户操作三维视图，而不必使用控件命令或自定义命令。通过设置 Navigate 属性，用户就可以操作三维视图，如前后左右移动、放大缩小等。

框架控件，如 TOCControl、ToolbarControl，需要与其他的控件协作使用。例如在 TOCControl 控件属性页中设置与 MapControl 关联，单在 MapControl 中删除一个图层时，该图层也从 TOCControl 中删除。

本章详细介绍这 6 个控件，并给出 C#的实际使用。

2.1 制图控件介绍

2.1.1 地图控件

MapControl 控件封装了 Map 对象，并提供了其他的属性、方法和事件，用于管理控件的外观、显示属性和地图属性，管理、添加数据图层，装载地图文档，显示、绘制跟踪图层。MapControl 上存在着诸如 TrackRectangle、TrackPolygon、TrackLine 和 TrackCircle 等帮助方法，用于追踪或

“橡皮圈住（rubber banding）”显示上的几何图形（Shape）。VisibleRegion 属性可用于更改 MapControl 显示区内的几何图形。MapControl 控件实现的主要接口有 IMapControlDefault、IMapControl2、IMapControl3、IMapControlEvents2 等，如图 2-1 所示。

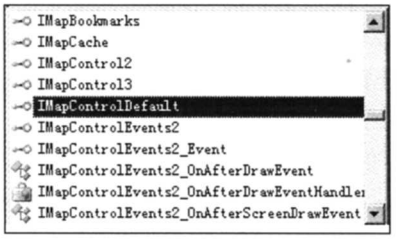


图 2-1　MapControl 控件实现的主要接口

2.1.1.1　IMapControlDefault 接口

IMapControlDefault 接口是地图控件缺省接口。当将 MapControl 控件拖放到容器上时，会自动创建一个 axMapControl1 的对象，该对象全部继承父类接口的属性和方法。下面的代码演示了如何使用这个接口。

```
IMapControlDefault mapControlDefault;
MapControl mapControl;
MapControl = axMapControl1.Object as IMapControlDefault;
```

2.1.1.2　IMapControl2 接口

IMapControl2 接口提供了一系列的属性和方法，如设置控件外观，设置 Map 对象或控件的显示属性，添加和管理数据图层、地图文挡，在控件上绘制图形和返回几何对象等，如图 2-2 所示。

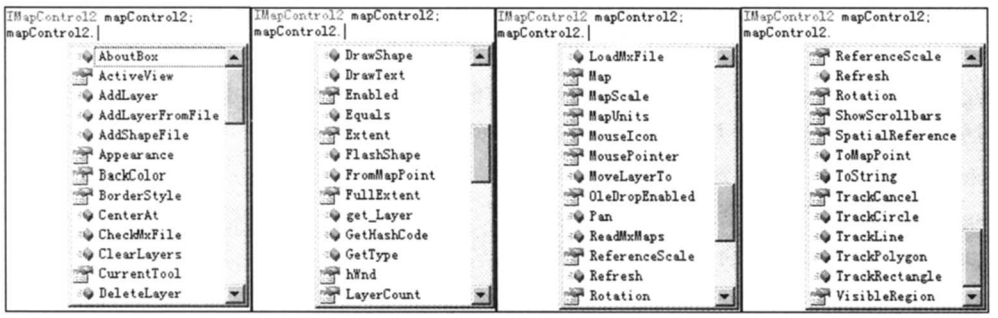


图 2-2　IMapControl2 接口提供的属性和方法

2.1.1.3　IMapControl3 接口

该接口继承自 IMapControl2，并增加了 8 个属性和一个方法。

- CustomProperty：设置自定义控件属性。
- DocumentFilename：返回 MapControl 装入的地图文档的文件名。
- DocumentMap：返回 MapControl 最后装入的地图名称。
- KyeIntercept：返回或设置 MapControl 截取键盘按键信息。
- Object：返回 MapControl 控件。

```
IMapControl2 mapControl2;
```

```
mapControl2=axMapControl1.Object as IMapControl2 ;
mapControl2 =axMapControl1.GetOcx() as IMapControl2 ;
IMapControl3 mapControl3;
mapControl3 = axMapControl1.Object as IMapControl3;
mapControl3 = axMapControl1.GetOcx() as IMapControl3;
```

- ShowMapTips：确定是否显示地图的 Map Tips。
- TipDelay：设置 Map Tips 的延迟时间。
- TipStyle：设置 Map Tips 的显示样式。
- SuppressResizeDrawing()：当控件尺寸发生变化时阻止数据实时重绘。

2.1.1.4 IMapControlEvents2 接口

IMapcontrolEvents2 定义了 MapControl 能够处理的全部事件，如图 2-3 所示。其中，OnBeforeScreenDraw 事件是屏幕绘制前触发的事件，OnViewRefreshed 是视频刷新触发事件。



图 2-3 MapControl 能够处理的全部事件

2.1.2 页面布局控件

PageLayoutControl 控件主要用于页面布局与制图。该控件封装了 PageLayout 类，提供了布局视图中控制元素的属性和方法，以及其他的事件、属性和方法。

Printer 属性提供了处理地图打印的设置。

Page 属性提供了处理控件的页面效果。

Element 属性则用于管理控件中的地图元素。

PageLayoutControl 控件不能添加地图图层或地理数据，必须通过使用 MXD 文件来加载需要处理的数据。PageLayoutControl 控件主要实现 IPageLayoutControlDefault、IPageLayoutControl、IPageLayoutControl2、IPageLayoutControlevents 等接口，如图 2-4 所示。

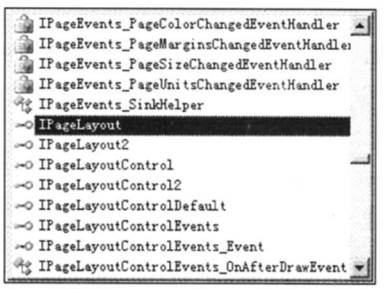


图 2-4 PageLayout Control 控件主要实现的接口

2.1.2.1　IPageLayoutControlDefault 接口

IPagelayoutControlDefault 是界面调用的默认接口。该接口是纯接口，因此可以在新版本中使用。接口的方法和属性都是相同的最高版本的主界面上的 PageLayoutControl，例如 IPageLayoutControlDefalut 相当于 IPageLayoutControl2，但在以后的新版本中可能成为 IPageLayoutControl3。通过使用 IPageLayoutControlDefault 接口，可以保证使用的 PageLayoutControl 是最新版本。

2.1.2.2　IPageLayoutControlevents 接口

2.2　3D 控件介绍

SceneControl 和 GlobeControl 都是嵌入式的开发组件，可以通过开发环境增加到窗体中或者对话框中，它们都提供了设置属性，并且能直接点控件来增加功能，当然这个属性页也提供了快捷的选择控件属性和方法，并允许开发者编写自己的代码来进行相关的操作。3D 控件主要用到两个类：**3DAnalyst 和 GlobeCore。**

3DAnalyst 类

包含了 3D scenes 的对象，当然也包含了 ArcGlobe 应用程序运用的类库，主要的开发组件是在 GlobeCore 类库中的定义，定义文件为 esri3DAnalyst.olb，主要定义了 SceneControl、Scene、SceneGraph、3Dproperties、SceneExporter3D、SceneViewer、3Dsymbol 和 AnimationTrack 等。

GlobeCore 类

是 ArcGlobe 应用程序主要运用的类库，同时进行基于 global 数据的 ArcGIS 3D 分析。在 GlobeCore 类库中，定义了 2D 和 3D 数据在 globe 的表面显示及数据操作，定义文件为 esriGlobeCore.olb，主要定义了 GlobeControl、Globe、GlobeDisplay、GlobeCamera、GlobeViewer、GlobeLayerProperties、GlobeLayer 和 AnimationTrack 等。

3DAnalyst 和 GlobeCore 类库包括了 SceneControl 和 GlobeControl 开发组件，并且提供了基于 Scene 和 Globe 的命令并可协同工作，SceneControl 和 GlobeControl 也可以与 ToolbarControl 和 TOCControl 开发组件关联。它所定义的类库如图 2-5 所示。

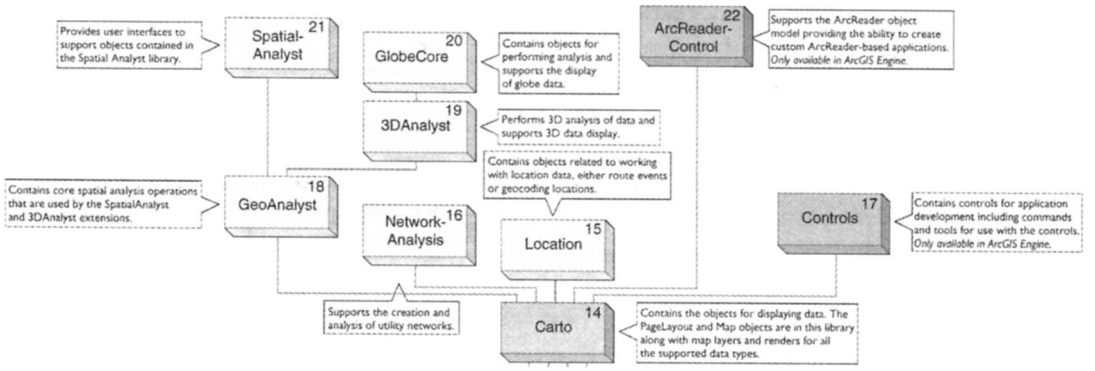


图 2-5　定义的类库

但是需要注意的是：SceneControl 和 GlobeControl 对图层的定义有所区别。Globe 的图层分为 esriGlobeLayerTypeDraped 、esriGlobeLayerTypeFloating、esriGlobeDataElevation 等 3 种，更适合三维用户使用。而在 SCENE 中就没有这种概念，这也是很多 SCENE 用户转向 GLOBE 时比

较迷茫的。Elevation 图层一经定义，所有的 Draped 图层将依附于 Elevation 图层表面上。如果要单独定义图层的 Elevation，就必须定义该图层为 Draped 图层，这一点就与 SCENE 一样了。

2.2.1 场景控件——SceneControl

SceneControl 是一个高性能的嵌入式的开发组件，提供给开发者建立和扩展 Scene 程序，当然也提供了基于 ArcScene™的功能给用户，以便进行绘图等操作。控件 SceneControl 相当于 ArcScene Desktop 应用程序中的 3D 视图，并且提供了显示和增加空间数据到 3D 的方法等。

SceneControl 是单一的开发进程，并且提供了粗粒度 ArcEngine 组件对象，也提供了强大纹理着色的功能。SceneControl 通过对象接口 ISceneViewer 来表现，ISceneViewer 接口提供了一个 Camera 对象，该对象由视角（Observer）和目标（Target）构成。SceneControl 控件提供一些属性和方法操作三维对象。例如，Camera、Scene、SceneGraph 和 SceneViewer 等属性。LoadSxFile 方法，用于导入 scene 文档。SceneControl 是进行三维开发最基本的控件。

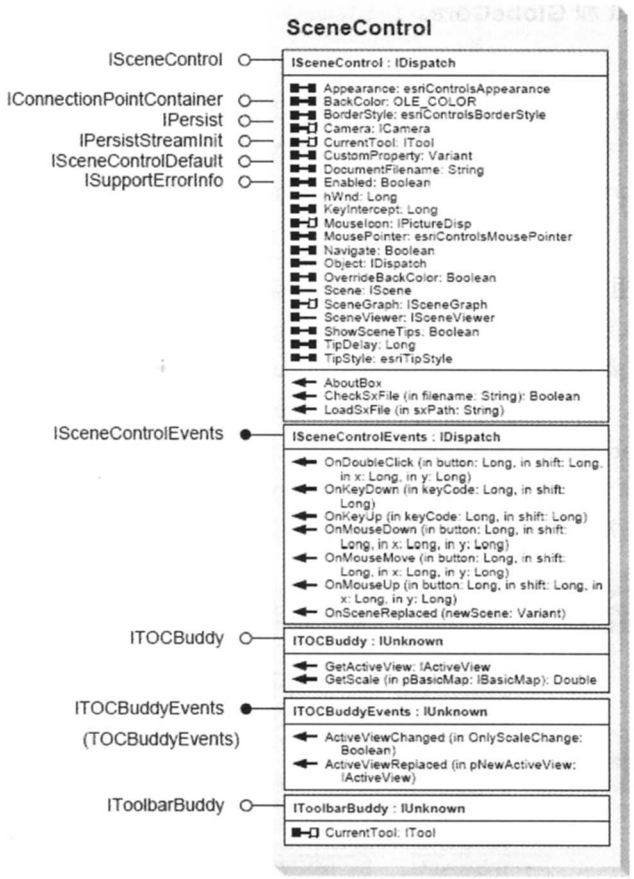


图 2-6 SceneControl 开发进程

3DAnalyst 类库包含了命令、工具和工具集，可与 SceneControl 控件共同工作来进行操作。例如，scene navigation 命令和工具用来移动对象到新的位置（camera），Scene 命令包含了 GUID 和命令描述，详细信息读者可以参考帮助中的“Names and IDs of the ControlCommands”部分，

帮助信息中介绍了 Commands 和 tools 使用，例如：Pan, zoom, fly, set observer, select Toolbars，我们可以看看下面所有 SceneControl 中已经提供了命令、工具，如图 2-7 所示。

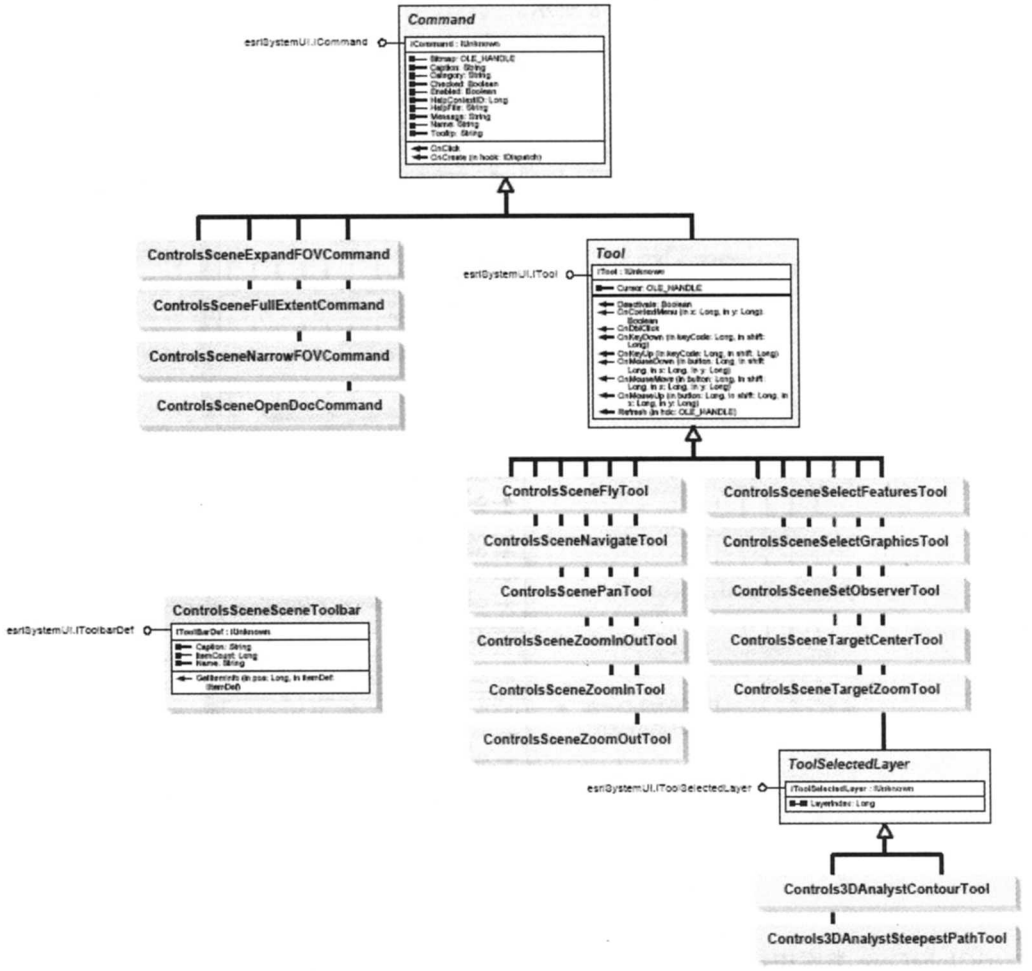


图 2-7　SceneControl 提供的命令、工具

2.2.2　Globe 控件

GlobeControl 是一个高性能的嵌入式的开发组件，提供给开发者建立和扩展 ArcGlobe 程序，当然也提供了基于 ArcGlobe ™的功能给用户，以便进行绘图等操作。GlobeControl 显示 3D 视图，并能提供全球表现的位置，而且是基于 3D 数据。GlobeControl 控件对应于 ArcGlobe 桌面应用程序的三维视图。GlobeControl 封装了 GlobeViewer 对象，可以加载 ArcGlobe 应用程序创作的 Globe 文档。

GlobeControl 也是单一的开发进程，并且提供了粗粒度 ArcEngine 组件对象，当然也提供了强大纹理着色的 ArcEngine 组件。GlobeControl 通过对象接口来操作 IGlobe 视图，用户可以通过 IGlobeViewer 对象来操作 ArcGlobe 应用程序。IGlobeViewer 接象包含一个 GlobeDisplay，GlobeDisplay 又包含一个 Globe。GlobeControl 提供了经常使用的属性和方法，例如，GlobeControl 有 GlobeCamera、Globe、GlobeDisplay 和 GlobeViewer 等属性，当然 GlobeControl 也能执行一些

方法或任务，例如，GlobeControl 有 Load3dFile 方法用于导入 globe 文档。GlobeControl 是进行三维开发最基本的控件，因为其提供了用户界面，所以更容易进行开发，当然使用对象模型也能很容易地理解及开发三维功能，如图 2-8 所示。

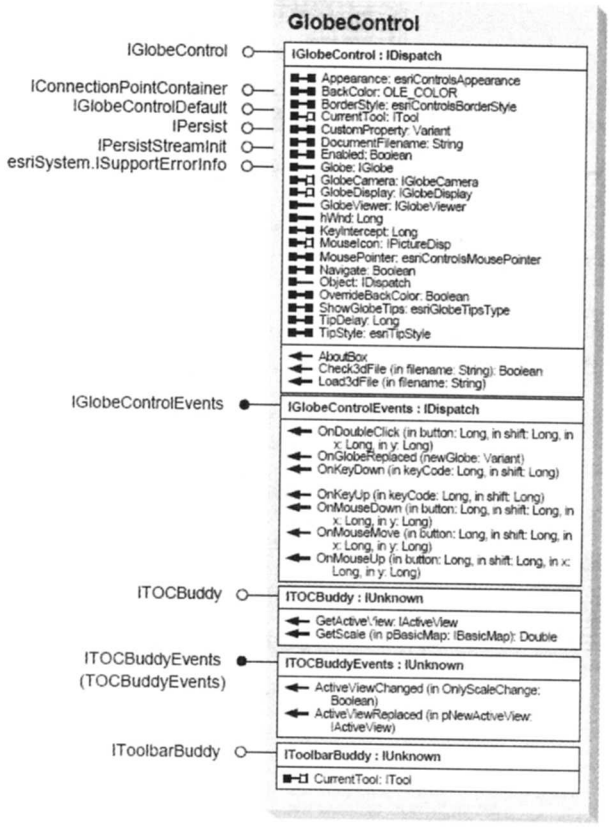


图 2-8　Globe Control 开发进程

esriGlobeCore 类库中包含了工具、命令和工具集与 GlobeControl 控件共同工作来执行操作，详细信息读者可以参考帮助中的“Names and IDs of the ControlCommands”部分，我们可以看看下面所有 GlobeControl 中已经提供了命令、工具，如图 2-9 所示。

2.2.3　SceneControl 和 GlobeControl 的异同

SceneControl 支持下面的主要特征：

3D 线符号有 Tubes、walls 和 textured lines

TIN 数据显示和分析

基于内存 Memory 的显示原理

支持立体和平面视图

表面分析工具，比如最短路径和等高线的生成 1

Layer 支持，比如图层坐标转换

输出到 3D 格式（vrml)

动态阴影效果

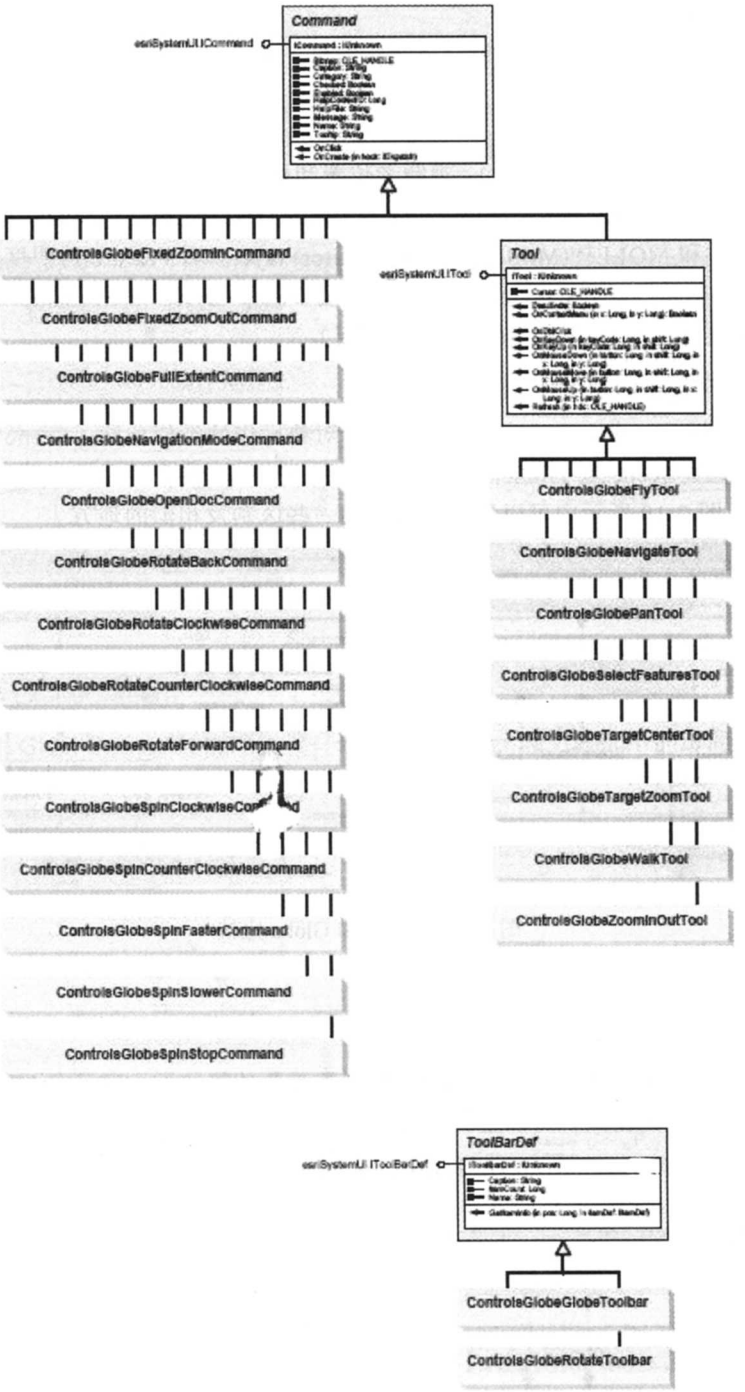


图 2-9　Globe Control 提供的命令、工具

GlobeControl 支持下面的主要特征：

所有的数据源必须具有空间参考

空间参考可以是地理坐标系统或工程坐标系统

页面显示，提供多级显示机制，对于大数据量支持 caching 的方式

超链接（hyperlinks)

导航和分析工具，比如量测、步行、放大、缩小、移动目标到中心、缩放目标

地图符号支持光栅要素图层

旋转工具栏

各式各样的显示目标（正面朝上）、观测者位置和指北针、剪切控制面板

可以创建一个模板图层，比如 MapServer layers、ArcIMS Image layers、feature annotation layers、WMS layers 和 MOLE™(Military Overlay Editor) layers。对 Feature 透明度，在 GlobeControl 中有一些数据不被支持，比如 Tracking Analyst 图层、测量 layers 和 Geostatistics 图层。另外，TINs 数据不直接支持，所以要转换成栅格（rasters）。

Scene 和 Globe

SceneControl 和 GlobeControl 有一些对于 3D 对象操作的类似功能，Scene 和 Globe 都支持 2D 和 3D 数据图层，Scene 或 Globe 中的对象都允许进行 3D 控制，其中 MapControl 中的 Map CoClass 相似，从图 2-10 所示中可以看到这两者的一些区别及相似的地方。

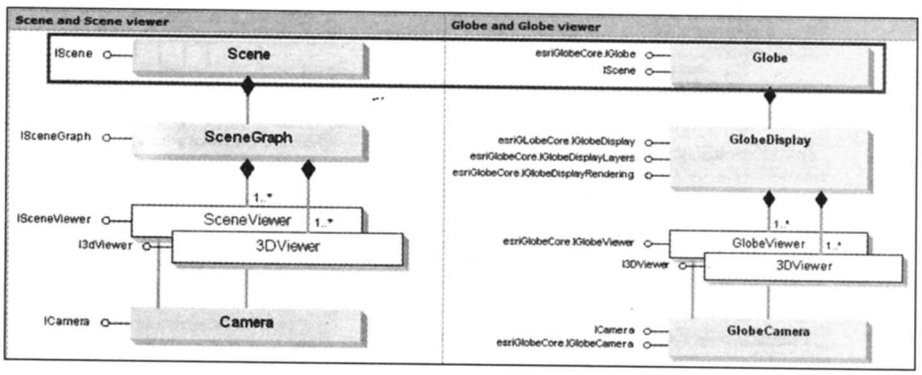


图 2-10　Scene 和 Globe 比较

下面进一步说明其内在的联系及区别。

SceneGraph 和 GlobeDisplay

SceneGraph 和 GlobeDisplay 都是进行 3D 绘制和着色功能的类，当然在 3D 制图程序方面还是有一些不同。对于视图窗口，可以进行 Add、remove、set active、refresh 等操作，也可以改变一些属性，比如 Vertical exaggeration、extent、contrast，以及对 caches 的控制。它们在模型中的位置，如图 2-11 所示。

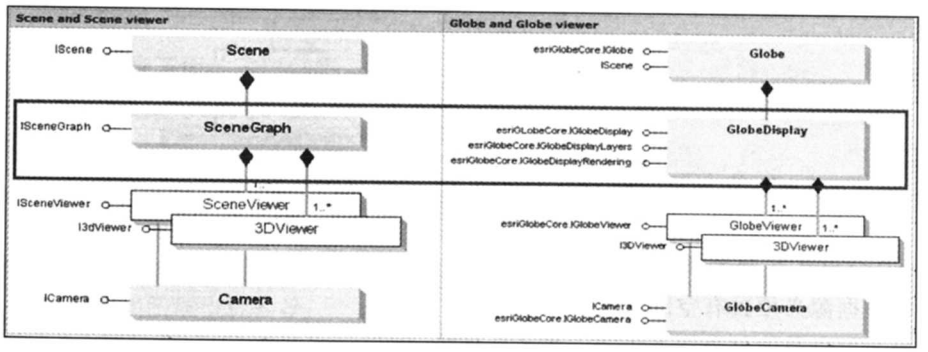


图 2-11　Scene Graph 和 Globe Display

SceneViewer 和 GlobeViewer

SceneViewer 和 GlobeViewer 是 SceneControl 和 GlobeControl 中的 3D 显示窗口，当然它们都各自支持自身的控件。这个当前的视图还可以通过 ActiveViewer 得到。I3DViewer 提供了一些共同的属性和方法来对 scene 和 globe 的视图进行操作，并且比如全屏等都可以在 Scene 和 Globe 中使用，它们都是非模态 3D 显示窗口，也能通过不同的视角来显示 3D 数据。它们在模型中的位置，如图 2-12 所示。

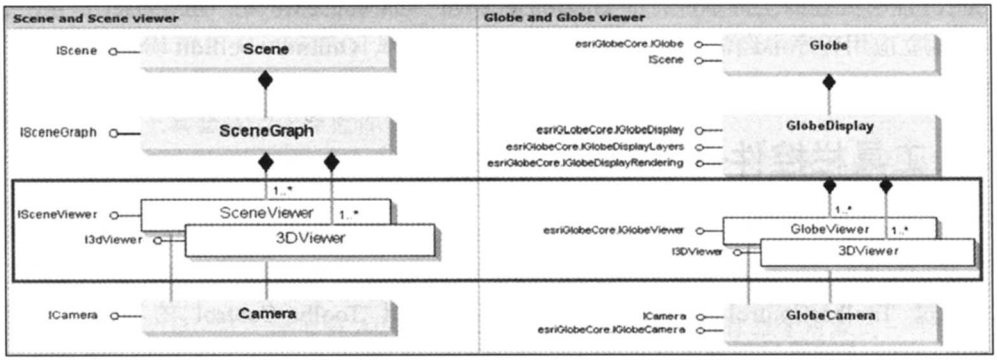


图 2-12　SceneViewer 和 Globe Viewer

Camera 和 GlobeCamera

Camera 和 GlobeCamera 分别控制着一个 scene viewer 和 Globe 中的不同的观测点。它们在模型中的位置，如图 2-13 所示。

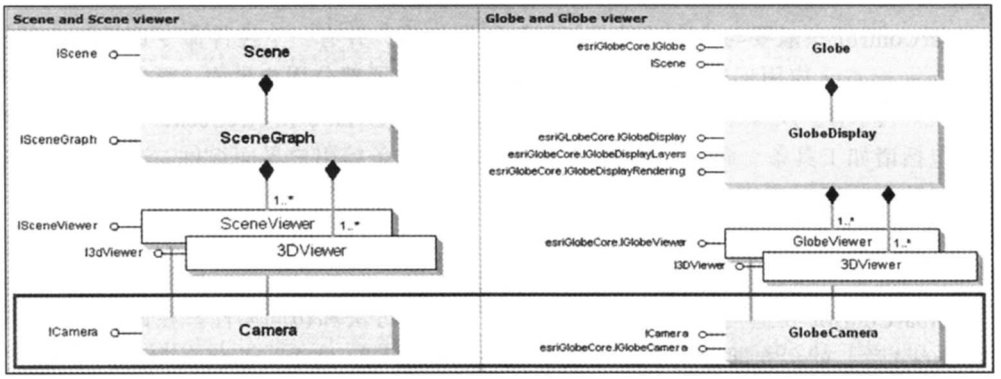


图 2-13　Camera 和 Globe Camera

2.3　框架控件介绍

2.3.1　图层树控件——TOCControl

TOCControl 是用来管理图层的可见性和标签的编辑。TOCControl 需要一个“伙伴控件”，或实现了 IActiveView 接口的对象协同工作。“伙伴控件”可以是 MapControl、PageLayoutControl 、ReaderControl、SceneControl 或 GlobeControl。“伙伴控件”可以通过 TOCControl 属性页设置，或者在驻留 TOCControl 的容器被显示时用 SetBuddyControl 方法通过

编程设置。TOCControl 的每个“伙伴控件”都实现了 ITOCBuddy 接口。TOCControl 用“伙伴控件”来显示其地图、图层和符号体系内容的一个交互树视图，并保持其内容与“伙伴控件”同步。TOCControl 通过 ITOCBuddy 接口来访问其“伙伴控件”。

TOCControl 的主要接口有两个，一个是 ITOCControl，一个是 ITOCControlEvents。ITOCControl 接口是任何与 TOCControl 有关的任务的出发点，如设置控件的外观、设置“伙伴控件”、管理图层的可见性和标签的编辑等。ITOCControlEvents 是一个事件接口，它定义了 TOCControl 能够处理的全部事件，如 OnMouseDown、OnMouseMove、OnMouseUp 等，这些事件在构建独立应用程序中经常使用，如 OnBeginLabelEdit、OnEndLabelEdit 分别是 TOCControl 中的标签开始编辑、结束编辑时触发。

2.3.2 工具栏控件——ToolbarControl

ToolbarControl 包括 6 个对象及相关接口：ToolbarControl、ToolbarItem、ToolbarMenu、CommandPool、CustomizeDialog、MissingCommand。ToolbarControl 要与一个“伙伴控件”协同工作，通过 ToolbarControl 属性页设置，或者在驻留 ToolbarControl 的容器被显示时用 SetBuddyControl 方法通过编程设置。

ToolbarControl 的每个“伙伴控件”都实现了 IToolbarBuddy 接口，这个接口用于设置“伙伴控件”的 CurrentTool 属性。如通过设置 MapControl 作为其“伙伴控件”，当用户单击该 ToolbarControl 上的“拉框放大”工具时，该放大工具就会成为 MapControl 的 CurrentTool。放大工具的实现过程是：通过 ToolbarControl 获取其“伙伴控件”，然后在 MapControl 上提供显示终端用户拉动鼠标所画的框，并改变 MapControl 的显示范围。

ToolbarControl 一般要与一个“伙伴控件”协同工作，并有一个控件命令选择集，以便快速提供功能强大的 GIS 应用程序。ToolbarControl 不仅提供了部分用户界面，而且还提供了部分应用程序框架。ArcGIS Desktop 应用程序，如 ArcMap、ArcGlobe 和 ArcScene 等具有强大而灵活的框架，包括诸如工具条、命令、菜单、泊靠窗口和状态条等用户界面组件，这些框架使终端用户可以通过改变位置、添加和删除这些用户界面组件来定制应用程序。

ArcGIS Engine 提供了几套使用 ArcGIS 控件的命令，以便执行某种特定动作，开发人员可通过创建执行特定任务的定制命令来扩展这套控件命令。所有的命令对象都实现了 ICommand 接口，ToolbarControl 在适当的时候要使用该接口来调用方法和访问属性。在命令对象被驻留到 ToolbarControl 后，就会立即调用 ICommand::OnCreate 方法，这个方法将一个句柄（Handle）或钩子（hook）传递给该命令操作的应用程序。命令的实现一般都要经过测试，以查看该钩子（hook）对象是否被支持，如果不支持则该钩子自动失效，如果支持，命令则存储该钩子以便以后使用。ToolbarControl 使用钩子（hook）来联系命令对象和“伙伴控件”，并提供了属性、方法和事件用于管理控件的外观，设置伙伴控件，添加、删除命令项，设置当前工具等。ToolbarControl 的主要接口有：IToolbarControl、IToolbarControlDefault、IToolbarControlEvents。

（1）IToolbarControl：该接口是任何与 ToolbarControl 有关的任务的出发点，如设置“伙伴控件”的外观，设置伙伴控件，添加或去除命令、工具、菜单等。

（2）IToolbarControlDefault：该接口是自动暴露的缺省的 dispatch 接口，该接口的属性和方法与 ToolbarControl 的最高版本主接口的属性、方法相同。例如目前版本中的 IToolbarControlDefault 等同于 IToolbarControl，但在今后的新版本中，可能会变为 IToolbarControl2。在开发中使用 IToolbarControlDefault 接口，能够保证总是访问到最新版本的 ToolbarControl。

（3）IToolbarControlEvents：该接口是一个事件接口，定义了 ToolbarControl 能够处理的全部事件，如 OnDoubleClick、OnItemClick、OnKeyDown 等。

在 ToolbarControl 上可以驻留以下 3 类命令。

（1）实现了响应单击事件的 ICommand 接口的单击命令。用户单击事件，会导致对 ICommand::OnClick 方法的调用，并执行某种动作。通过改变 ICommand::Checked 属性的值，简单命令项的行为就像开关那样。单击命令是可以驻留在菜单中的惟一命令类型。

（2）实现了 ICommand 接口和 ITool 接口、需要终端用户与“伙伴控件”的显示进行交互的工具。ToolbarControl 维护 CurrentTool 属性。当终端用户单击 ToolbarControl 上的工具时，该工具就成为 CurrentTool。ToolbarControl 会设置“伙伴控件”的 CurrentTool 属性。当某个工具为 CurrentTool 时，该工具会从“伙伴控件”收到鼠标和键盘事件。

（3）实现了 ICommand 接口和 IToolControl 接口的工具控件。这通常是用户界面组件。ToolbarControl 驻留了来自 ItoolControl：hWnd 属性窗口句柄提供的一个小窗口，只能向 ToolbarControl 添加特定工具控件的一个例程。

可以使用 3 种方法向 ToolbarControl 添加命令，第 1 种是指定唯一识别命令的一个 UID，第 2 种是指定一个 progID，第 3 种是给 AddToolbarDef 方法提供某个现有命令对象的一个例程。下面给出样例代码。

```
    //Add a toolbardef by passing a UID
    UID uID = new UIDsClass();
    uID.Value = "esriControls.ControlsMapNavigationToolbar";
    axToolbarControl1.AddToolbarDef(uID,-1,false,0,esriCommandStyles.esriCommandSt
yleIconOnly);
    //Add a toolbardef by passing a ProgID
    string progID = "esriControls.ControlsMapNavigationToolbar";
    axToolbarControl1.AddToolbarDef(progID,-1,false,0,esriCommandStyles.esriComman
dStyleIconOnly);
    //Add a toolbardef by passing an IToolbarDef
    IToolBarDef toolBarDef = new ControlsMapNavigationToolbarClass();
    axToolbarControl1.AddToolbarDef(toolBarDef,-1,false,0,esriCommasndSLyles.esriC
ommandstyleIconOnly);
```

ToolbarControl 更新命令的使用。默认情况下，ToolbarControl 每半秒钟自动更新一次，以确保驻留在 ToolbarControl 上的每个工具条命令项的外观与底层命令的 Enabled、Bitmap、Caption 等属性同步。改变 UpdateInterval 属性可以更改更新频率。UpdateInterval 为 0 会停止任何自动发生的更新，可以通过编程调用 Update 方法以刷新每个工具条命令项的状态。

在应用程序中首次调用 Update 方法时，ToolbarControl 会检查每个工具条命令项的底层命令的 ICommand::OnCreate 方法是否已经被调用过，如果还没有调用过该方法，该 ToolbarCommand 将作为钩子被自动传递给 ICommand::OnCreate 方法。

2.4 控件使用实例

2.4.1 地图加载、保存

在 VS2005 中创建一个 Windows 应用程序项目，命名为“lesson1”，开发语言选择 C#（本书

所有的代码均为C#语言）。在左边的工具箱中选择“LicenseControl”控件，拖放到默认的“Form1”窗体上（本书所有的程序都需要此控件，以后不再说明）。用同样方法在窗体上添加“MapControl”控件，在窗体上放置两个按钮控件“button1”、“button2”，修改按钮控件文本如图2-14所示。

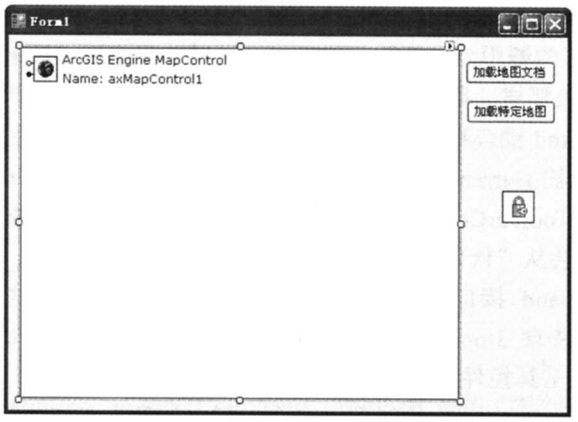


图2-14　新建程序项目lesson1

在代码窗口添加如下代码。

```
private void button1_Click(object sender, EventArgs e)
{
     loadMapDocument();
}
private void button2_Click(object sender, EventArgs e)
{
loadMapDocument2();
}
//加载地图文档
private void loadMapDocument()
{
   System.Windows.Forms.OpenFileDialog openFileDialog;
   openFileDialog = new OpenFileDialog();
   openFileDialog.Title = "打开地图文档";
   openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
   openFileDialog.ShowDialog();
   string filePath = openFileDialog.FileName;
   if (axMapControl1.CheckMxFile(filePath))
   {
      axMapControl1.MousePointer = esriControlsMousePointer.esriPointerHourglass;
      axMapControl1.LoadMxFile(filePath, 0, Type.Missing);
      axMapControl1.MousePointer = esriControlsMousePointer.esriPointerDefault;
   }
   else
   {
      MessageBox.Show(filePath + "不是有效的地图文档");
    }
}
//加载地图文档中的特定地图
private void loadMapDocument2()
```

```
    {
      System.Windows.Forms.OpenFileDialog openFileDialog;
      openFileDialog = new OpenFileDialog();
      openFileDialog.Title = "打开地图文档";
      openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
      openFileDialog.ShowDialog();
      string filePath = openFileDialog.FileName;
      if (axMapControl1.CheckMxFile(filePath))
      {
          IArray arrayMap = axMapControl1.ReadMxMaps(filePath, Type.Missing);
          int i;
          IMap map;
          for (i = 0; i < arrayMap.Count; i++)
          {
             map = arrayMap.get_Element(i) as IMap;
             if (map.Name == "Layers")
             {
              axMapControl1.MousePointer=esriControlsMousePointer.
esriPointerHourglass;
              axMapControl1.LoadMxFile(filePath, 0, Type.Missing);
              axMapControl1.MousePointer= esriControlsMousePointer.esriPointerDefault;
              break;
             }
          }
        }
        else
        {
           MessageBox.Show(filePath + "不是有效的地图文档");
        }
    }
```

创建一个新 Windows 应用程序项目，命名为“lesson2”，在左边工具箱中拖放“MapControl”控件到窗体上，在窗体上放置 3 个按钮控件“button1”、“button2”和“button3”，修改控件的文本如图 2-15 所示。

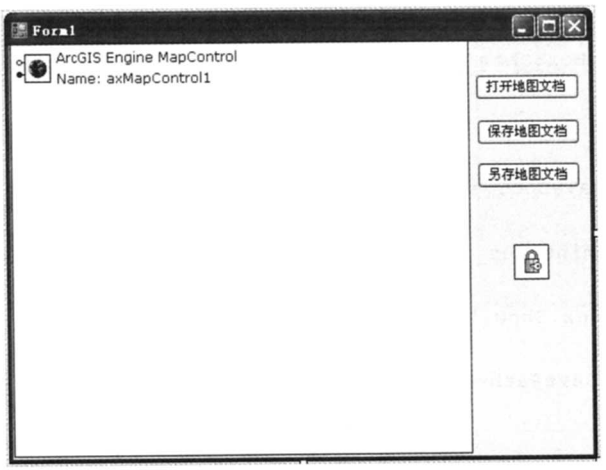


图 2-15　应用程序项目 lesson2

在代码窗口中添加如下代码。

```
private void button1_Click(object sender, EventArgs e)
{
   loadMapDoc();
}
private void button2_Click(object sender, EventArgs e)
{
   saveDocument();
}
private void button3_Click(object sender, EventArgs e)
{
   saveAsDocument();
}
IMapDocument mapDocument;
//加载地图文档
private void loadMapDoc()
{
    mapDocument = new MapDocumentClass();
    try
    {
        System.Windows.Forms.OpenFileDialog openFileDialog;
        openFileDialog = new OpenFileDialog();
        openFileDialog.Title = "打开地图文档";
        openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
        openFileDialog.ShowDialog();
        string filePath = openFileDialog.FileName;
        mapDocument.Open(filePath, "");
        for (int i = 0; i < mapDocument.MapCount; i++)
        {
            axMapControl1.Map = mapDocument.get_Map(i);
        }
        axMapControl1.Refresh();
    }
    catch (Exception e)
    {
        MessageBox.Show(e.ToString());
    }
}
//保存地图文档
private void saveDocument()
{
   if (mapDocument.get_IsReadOnly(mapDocument.DocumentFilename) == true)
   {
       MessageBox.Show("地图文档是只读的无法保存");
   }
   string fileSavePath=@"e:\new.mxd";
   try
   {
       mapDocument.Save(mapDocument.UsesRelativePaths, true);
       MessageBox.Show("保存地图文档成功");
```

```
        }
        catch (Exception e)
        {
            MessageBox.Show("保存地图文档失败!!! " + e.ToString());
        }
    }
    private void saveAsDocument()
    {
        if (mapDocument.get_IsReadOnly(mapDocument.DocumentFilename) == true)
        {
            MessageBox.Show("地图文档是只读的无法保存");
        }
        string fileSavePath = @"e:\new.mxd";
        try
        {
            mapDocument.SaveAs(fileSavePath, true, true);
            MessageBox.Show("另存地图文档成功");
        }
        catch (Exception e)
        {
            MessageBox.Show("另存地图文档失败!!! " + e.ToString());
        }
    }
```

2.4.2 图层操作

创建一个新 Windows 应用程序项目，命名为“lesson3”，在左边工具箱中拖放“MapControl”控件到窗体上，在窗体上放置 4 个按钮控件“button1”、“button2”、“button3”和“button4”，修改控件的文本如图 2-16 所示。

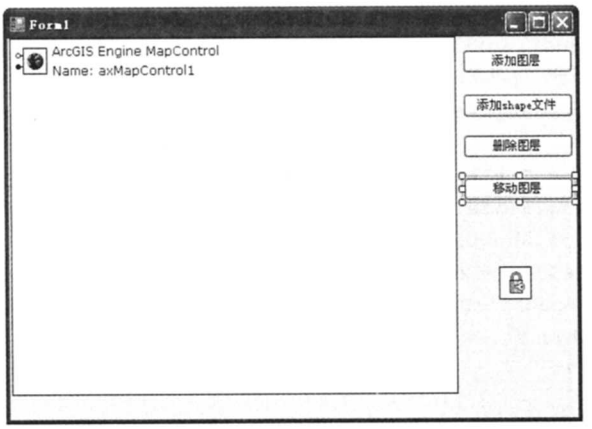


图 2-16　应用程序项目 lesson3

在代码窗口中添加如下代码。

```
    private void button1_Click(object sender, EventArgs e)
    {
        addLayerFile();
```

```
    }
    private void button2_Click(object sender, EventArgs e)
    {
       addShapeFile();
    }
    private void button3_Click(object sender, EventArgs e)
    {
       deleteLayer();
    }
    private void button4_Click(object sender, EventArgs e)
    {
       moveLayer();
    }
    //添加图层文件
    private void addLayerFile()
    {
        System.Windows.Forms.OpenFileDialog openFileDialog;
        openFileDialog = new OpenFileDialog();
        openFileDialog.Title = "打开图层文件";
        openFileDialog.Filter = "map documents(*.lyr)|*.lyr";
        openFileDialog.ShowDialog();
        string filePath = openFileDialog.FileName;
        try
        {
            axMapControl1.AddLayerFromFile(filePath);
        }
        catch (Exception e)
        {
           MessageBox.Show("添加图层失败!!! " + e.ToString());
        }
    }
    //添加 shape 文件
    private void addShapeFile()
    {
       System.Windows.Forms.OpenFileDialog openFileDialog;
       openFileDialog = new OpenFileDialog();
       openFileDialog.Title = "打开图层文件";
       openFileDialog.Filter = "map documents(*.shp)|*.shp";
       openFileDialog.ShowDialog();
       FileInfo fileInfo = new FileInfo(openFileDialog.FileName);
       String path=openFileDialog.FileName.Substring(0,openFileDialog.FileName.Length
file Info.Name.Length);
       try
       {
           axMapControl1.AddShapeFile (path,fileInfo.Name  );
       }
       catch (Exception e)
       {
           MessageBox.Show("添加图层失败!!! " + e.ToString());
       }
   }
```

```
//删除图层
private void deleteLayer()
{
    try
    {
        //删除地图中所有的图层
        for (int i = axMapControl1.LayerCount-1; i >= 0; i--)
        {
            axMapControl1.DeleteLayer(i);
        }
    }
    catch (Exception e)
    {
        MessageBox.Show("删除图层失败!!! " + e.ToString());
    }
}
//移动图层
private void moveLayer()
{
    if (axMapControl1.LayerCount > 0)
    {
        try
        {
            //将最下层图层文件移动到最上层
            axMapControl1.MoveLayerTo(axMapControl1.LayerCount - 1, 0);
        }
        catch (Exception e)
        {
            MessageBox.Show("移动图层失败!!! " + e.ToString());
        }
    }
}
```

创建一个新 Windows 应用程序项目，命名为“lesson4”，在左边工具箱中拖放“MapControl”控件到窗体上，在窗体上放置 4 个按钮控件“button1”、“button2”、“button3”和“button4”，修改控件的文本如图 2-17 所示。

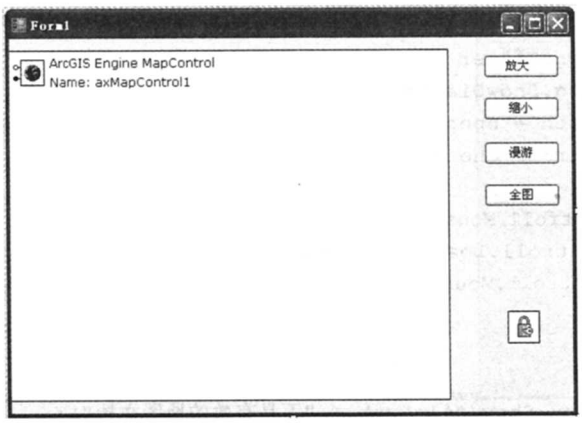


图 2-17　应用程序项目 lesson4

在代码窗口中添加如下代码。

```
private void Form1_Load(object sender, EventArgs e)
{
     //加载地图文档
    loadMapDocument();
}
//标记变量，标记地图操作
int flag = 0;
//放大
private void button1_Click(object sender, EventArgs e)
{
    axMapControl1.MousePointer = esriControlsMousePointer.esriPointerZoomIn;
     flag = 2;
}
//缩小
private void button2_Click(object sender, EventArgs e)
{
    axMapControl1.MousePointer = esriControlsMousePointer.esriPointerZoomOut;
    flag = 1;
}
//漫游
private void button3_Click(object sender, EventArgs e)
{
    axMapControl1.MousePointer = esriControlsMousePointer.esriPointerPan;
    flag = 3;
}
//全图
private void button4_Click(object sender, EventArgs e)
{
    axMapControl1.Extent  = axMapControl1.FullExtent ;
}
//加载地图文档
private void loadMapDocument()
{
    System.Windows.Forms.OpenFileDialog openFileDialog;
    openFileDialog = new OpenFileDialog();
    openFileDialog.Title = "打开地图文档";
    openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
    openFileDialog.ShowDialog();
    string filePath = openFileDialog.FileName;
    if (axMapControl1.CheckMxFile(filePath))
    {
         axMapControl1.MousePointer = esriControlsMousePointer.esriPointerHourglass;
         axMapControl1.LoadMxFile(filePath, 0, Type.Missing);
         axMapControl1.MousePointer = esriControlsMousePointer.esriPointerDefault;
    }
    else
    {
        MessageBox.Show(filePath + "不是有效的地图文档");
    }
```

```
    }
    private void axMapControl1_OnMouseDown(object sender, IMapControlEvents2_ OnMou-
seDownEvent e)
    {
        IEnvelope ipEnv;
        if (flag == 2)
        {
            ipEnv = axMapControl1.TrackRectangle();
            axMapControl1.Extent = ipEnv;
        }
        else if (flag == 1)
        {
            ipEnv = axMapControl1.TrackRectangle();
            ipEnv = axMapControl1.Extent;
            ipEnv.Expand(2, 2, true);
            axMapControl1.Extent = ipEnv;
        }
        else if (flag == 3)
        {
            ipEnv = axMapControl1.Extent;
            axMapControl1.Pan();
        }
    }
```

2.4.3　绘制图形

MapControl 控件提供了直接在控件上绘制图形和文字的方法，如 DrawShape、DrawText。这两个方法绘制的图形都是缓存（cache），而不是真正地保存，一旦出现窗口重绘，如地图刷新等，这些图形就将消失。

创建一个新 Windows 应用程序项目，命名为“lesson5”，在左边工具箱中拖放“MapControl”控件到窗体上，在窗体上放置 5 个按钮控件“button1”、“button2”、“button3” 、“button4”和“button5”，修改控件的文本如图 2-18 所示。

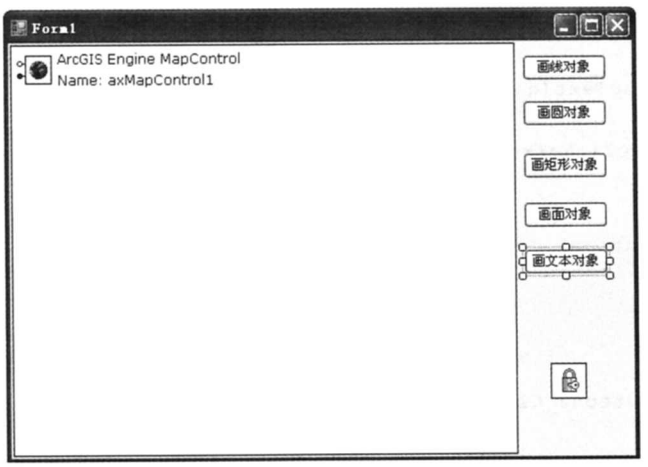


图 2-18　应用程序项目 lesson5

在代码窗口中添加如下代码。

```
    int flag = 0;
    private void axMapControl1_OnMouseDown(object sender, ESRI.ArcGIS.Controls.IMap
Control Events2_OnMouseDownEvent e)
    {
        axMapControl1.MousePointer =esriControlsMousePointer.esriPointerCrosshair;
        IGeometry geometry =null  ;
         //
        if (flag ==1)
        {
            geometry =axMapControl1.TrackLine() ;
        }
        else if (flag==2)
        {
            geometry =axMapControl1.TrackCircle ();
        }
        else if (flag==3)
        {
            geometry=axMapControl1.TrackRectangle ();
        }
        else if (flag==4)
        {
            geometry=axMapControl1.TrackPolygon ();
        }
        else if (flag==5)
        {
            IPoint point =new PointClass  ();
            point.X =e.x ;
            point.Y =e.y ;
            geometry =point as IGeometry ;
        }
        if (flag >= 1 && flag <= 4)
        {
            drawMapShape(geometry);
        }
        else if (flag == 5)
        {
            drawMapText(geometry);
        }
        axMapControl1.Refresh (esriViewDrawPhase.esriViewGeography, null, null);
    }
    //画线对象
    private void button1_Click(object sender, EventArgs e)
    {
        flag = 1;
    }
    //画圆对象
    private void button2_Click(object sender, EventArgs e)
    {
        flag = 2;
    }
```

```
//画矩形对象
private void button3_Click(object sender, EventArgs e)
{
    flag = 3;
}
//画面对象
private void button4_Click(object sender, EventArgs e)
{
    flag = 4;
}
//画文本对象
private void button5_Click(object sender, EventArgs e)
{
    flag = 5;
}
private void drawMapText(IGeometry geometry)
{
    IRgbColor color = new RgbColorClass();
    color.Red = 255;
    color.Blue = 0;
    color.Green = 0;
    ITextSymbol txtSystem = new TextSymbolClass();
    txtSystem.Color = color;
    //txtSystem.Text = "测试 DrawText";
    object symbol = txtSystem;
    axMapControl1.DrawText(geometry, "测试 DrawText", ref  symbol);
}
private void drawMapShape(IGeometry pGeom)
{
    IRgbColor pColor;
    pColor = new RgbColorClass();
    pColor.Red = 255;
    pColor.Green =255;
    pColor.Blue =0;
    object symbol=null;
    if (pGeom.GeometryType == esriGeometryType.esriGeometryPolyline)
    {
        ISimpleLineSymbol simpleLineSymbol;
        simpleLineSymbol = new SimpleLineSymbolClass();
        simpleLineSymbol.Color = pColor;
        simpleLineSymbol.Width = 5;
        symbol = simpleLineSymbol;
    }
    else
    {
        ISimpleFillSymbol simpleFillSymbol;
        simpleFillSymbol = new SimpleFillSymbolClass();
        simpleFillSymbol.Color = pColor;
        symbol = simpleFillSymbol;
    }
    axMapControl1.DrawShape(pGeom, ref symbol);
}
```

2.4.4 鹰眼图功能实现

使用鹰眼图，用户可以很直观地看到主视图中的地图范围在整个地图范围内的位置，犹如鸟瞰一样，本例就来实现这一功能。本例需要使用两个 AxMapControl 控件，即主控件 AxMapControl1 和鹰眼控件 AxMapControl2。要实现这一功能，主要是保持两个控件显示的数据一致，以及在鹰眼控件的显示方框中让两个控件的数据共享。

创建一个新 Windows 应用程序项目，命名为“lesson6”，在左边工具箱中拖放两个“MapControl”控件到窗体上，如图 2-19 所示。

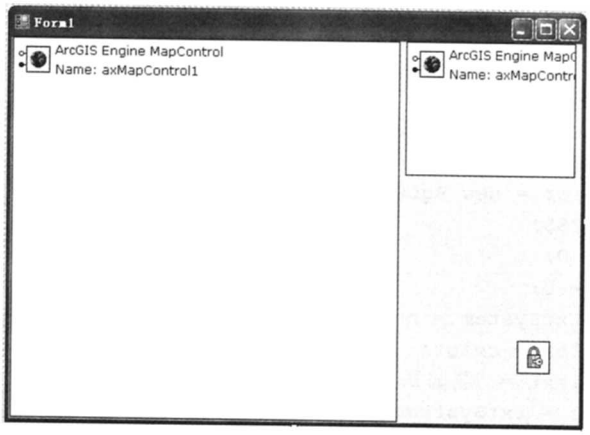


图 2-19 应用程序项目 lesson6

在 AxMapControl1 的 OnMapReplaced 事件中加入如下代码。

```
private void Form1_Load(object sender, EventArgs e)
{
    loadMapDocument();
}
//加载地图文档
private void loadMapDocument()
{
    System.Windows.Forms.OpenFileDialog openFileDialog;
    openFileDialog = new OpenFileDialog();
    openFileDialog.Title = "打开地图文档";
    openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
    openFileDialog.ShowDialog();
    string filePath = openFileDialog.FileName;
    if (axMapControl1.CheckMxFile(filePath))
    {
       axMapControl1.MousePointer = esriControlsMousePointer.esriPointerHourglass;
       axMapControl1.LoadMxFile(filePath, 0, Type.Missing);
       axMapControl1.MousePointer = esriControlsMousePointer.esriPointerDefault;
    }
    else
    {
       MessageBox.Show(filePath + "不是有效的地图文档");
    }
```

```
    }
    private void loadEagleEyeDocument()
    {
        System.Windows.Forms.OpenFileDialog openFileDialog;
        openFileDialog = new OpenFileDialog();
        openFileDialog.Title = "打开鹰眼地图文档";
        openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
        openFileDialog.ShowDialog();
        string filePath = openFileDialog.FileName;
        if (axMapControl2.CheckMxFile(filePath))
        {
            axMapControl2.MousePointer = esriControlsMousePointer.esriPointerHourglass;
            axMapControl2.LoadMxFile(filePath, 0, Type.Missing);
            axMapControl2.MousePointer = esriControlsMousePointer.esriPointerDefault;
        }
        else
        {
            MessageBox.Show(filePath + "不是有效的地图文档");
        }
    }
    private void axMapControl1_OnMapReplaced(object sender, ESRI.ArcGIS.Controls.
IMapControlEvents2_OnMapReplacedEvent e)
    {
        loadEagleEyeDocument();
    }
    private void axMapControl1_OnMouseDown(object sender, IMapControlEvents2_OnMouse
DownEvent e)
    {
        if (e.button == 1)
        {
            axMapControl1.Extent = axMapControl1.TrackRectangle();
            axMapControl1.Refresh(esriViewDrawPhase.esriViewBackground, null,null);
        }
        else if (e.button == 2)
        {
            axMapControl1.Pan();
            axMapControl1.Refresh(esriViewDrawPhase.esriViewBackground, null, null);
        }
    }

    private void axMapControl1_OnExtentUpdated(object sender, IMapControlEvents2_
OnExtentUpdatedEvent e)
    {
        IEnvelope envelope = (IEnvelope)e.newEnvelope;
        IGraphicsContainer graphicsContainer = axMapControl2.Map as IGraphicsContainer;
        IActiveView activeView = graphicsContainer as IActiveView;
        //在绘制前，清除 axMapControl2 中的任何图形元素
        graphicsContainer.DeleteAllElements ();
        IElement element = new RectangleElementClass();
        element.Geometry = envelope ;
        //设置鹰眼图中的红线
        //产生一个线符号对象
```

```
        ILineSymbol outLineSymbol = new SimpleLineSymbolClass();
        outLineSymbol.Width = 2;
        outLineSymbol.Color = GetColor(255, 0, 0, 255);
        //设置颜色属性
        //设置填充符号的属性
        IFillSymbol fillSymbol = new SimpleFillSymbolClass();
        fillSymbol.Color = GetColor(9, 0, 0, 0);
        fillSymbol.Outline = outLineSymbol ;
        IFillShapeElement fillShapeElement = element as IFillShapeElement;
        fillShapeElement.Symbol = fillSymbol;
        graphicsContainer.AddElement((IElement)fillShapeElement , 0);
        activeView.PartialRefresh(esriViewDrawPhase.esriViewGraphics, null, null);
    }
    private IRgbColor GetColor(int r, int g, int b, int t)
    {
        IRgbColor rgbColor = new RgbColorClass();
        rgbColor.Red = r;
        rgbColor.Green = g;
        rgbColor.Blue = b;
        rgbColor.Transparency = (byte)t;
        return rgbColor;
    }
    private void axMapControl2_OnMouseDown(object sender, IMapControlEvents2_OnMouse
DownEvent e)
    {
        IPoint point = new PointClass();
        point.PutCoords(e.mapX , e.mapY);
        axMapControl1.CenterAt(point);
    }
```

2.4.5 数据查询与选择

创建一个新 Windows 应用程序项目，命名为“lesson7”，在左边工具箱中拖放“MapControl”控件到窗体上，在窗体上放置 5 个按钮控件“button1”、“button2”、“button3” 、“button4”和“button5”，修改控件的文本如图 2-20 所示。

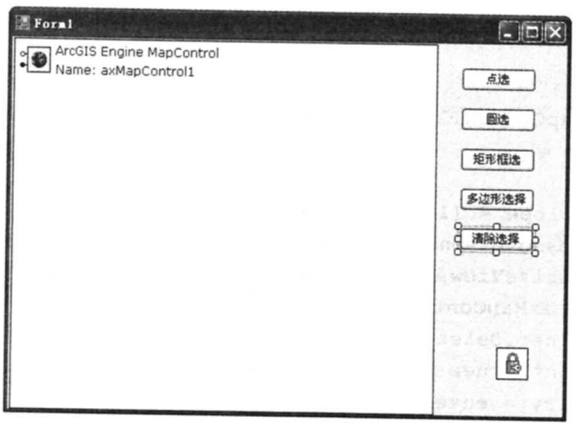


图 2-20 应用程序项目 lesson7

在代码窗口中添加如下代码。

```
private void Form1_Load(object sender, EventArgs e)
{
    loadMapDocument();
}
//加载地图文档
private void loadMapDocument()
{
    System.Windows.Forms.OpenFileDialog openFileDialog;
    openFileDialog = new OpenFileDialog();
    openFileDialog.Title = "打开地图文档";
    openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
    openFileDialog.ShowDialog();
    string filePath = openFileDialog.FileName;
    if (axMapControl1.CheckMxFile(filePath))
    {
        axMapControl1.MousePointer = esriControlsMousePointer.esriPointerHourglass;
        axMapControl1.LoadMxFile(filePath, 0, Type.Missing);
        axMapControl1.MousePointer = esriControlsMousePointer.esriPointerDefault;
    }
    else
    {
        MessageBox.Show(filePath + "不是有效的地图文档");
    }
}
private void button1_Click(object sender, EventArgs e)
{
    //点选
    flag = 1;
}
private void button2_Click(object sender, EventArgs e)
{
    //圆选
    flag = 2;
}
private void button3_Click(object sender, EventArgs e)
{
    //矩形框选
    flag = 3;
}
private void button4_Click(object sender, EventArgs e)
{
    //多边形选择
    flag = 4;
}
private void button5_Click(object sender, EventArgs e)
{
    //清除选择
    //在这里需要特别注意，在清除选择集前必须做一个刷新，否则在视图上看不到清除的效果
    IActiveView pActiveView = (IActiveView)( axMapControl1.Map );
```

```
        pActiveView.PartialRefresh(esriViewDrawPhase.esriViewGeoSelection,axMapControl1.
get_Layer(0), null);
        axMapControl1.Map.ClearSelection();
        pActiveView.PartialRefresh(esriViewDrawPhase.esriViewGeoSelection, axMapControl1.
get_Layer(0), null);
    }
    //名称查询
    private void button6_Click(object sender, EventArgs e)
    {
        string searchName = this.textBox1.Text.Trim();
        ILayer layer = axMapControl1.Map.get_Layer(0);
        IFeatureLayer featureLayer = layer as IFeatureLayer;
        IFeatureClass featureClass = featureLayer.FeatureClass;
        IQueryFilter queryFilter =new   QueryFilterClass ();
        IFeatureCursor featureCursor;
        IFeature feature = null;
        queryFilter.WhereClause ="continent ='" +searchName +"'";
        featureCursor = featureClass.Search(queryFilter, true);
        feature = featureCursor.NextFeature();
        if (feature !=null)
        {
            axMapControl1.Map.SelectFeature(axMapControl1.get_Layer(0), feature);
            axMapControl1.Refresh(esriViewDrawPhase.esriViewGeoSelection, null, null);
        }
    }
    private  void  axMapControl1_OnMouseDown(object  sender,  ESRI.ArcGIS.Controls.
IMapControl Events2_OnMouseDownEvent e)
    {
        axMapControl1.MousePointer =esriControlsMousePointer.esriPointerCrosshair;
        IGeometry geometry = null;
        ISelection selection;
        switch (flag)
        {
            case 1:
                ESRI.ArcGIS.Geometry.Point point = new ESRI.ArcGIS.Geometry.PointClass();
                point.X = e.mapX;
                point.Y = e.mapY;
                geometry = point as IGeometry;
                break;
            case 2:
                geometry = axMapControl1.TrackCircle();
                break;
            case 3:
                geometry = axMapControl1.TrackRectangle();
                break;
            case 4:
                geometry = axMapControl1.TrackPolygon();
                break;
        }
        axMapControl1.Map.SelectByShape(geometry, null, false);
        axMapControl1.Refresh(esriViewDrawPhase.esriViewGeoSelection, null, null);
    }
```

2.4.6　布局控件与地图控件关联

创建一个新 Windows 应用程序项目，命名为“lesson8”，在左边工具箱中拖放“MapControl”和“PageLayoutControl”控件到窗体上，如图 2-21 所示。

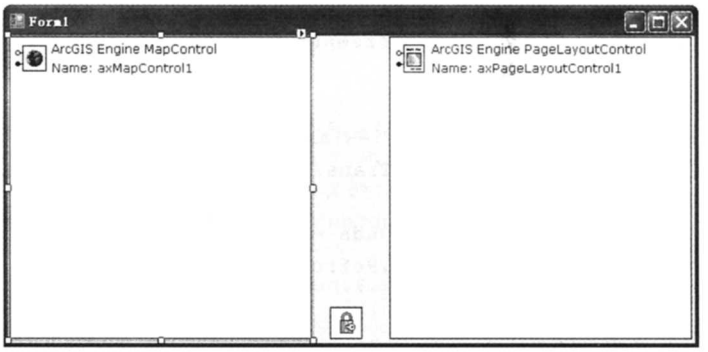


图 2-21　应用程序项目 lesson 8

在代码窗口中添加如下代码。

```
private void Form1_Load(object sender, EventArgs e)
{
    loadMapDocument();
}
//加载地图文档
private void loadMapDocument()
{
    System.Windows.Forms.OpenFileDialog openFileDialog;
    openFileDialog = new OpenFileDialog();
    openFileDialog.Title = "打开地图文档";
    openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
    openFileDialog.ShowDialog();
    string filePath = openFileDialog.FileName;
    if (axMapControl1.CheckMxFile(filePath))
    {
       axMapControl1.MousePointer = esriControlsMousePointer.esriPointerHourglass;
       axMapControl1.LoadMxFile(filePath, 0, Type.Missing);
       axMapControl1.MousePointer = esriControlsMousePointer.esriPointerDefault;
    }
    else
    {
       MessageBox.Show(filePath + "不是有效的地图文档");
    }
}
private void copyToPageLayout()
{
    IObjectCopy objectCopy = new ObjectCopyClass();
    object copyFromMap = axMapControl1.Map;
    object copyMap = objectCopy.Copy(copyFromMap);
    object copyToMap = axPageLayoutControl1.ActiveView.FocusMap;
```

```
        objectCopy.Overwrite(copyMap, ref copyToMap);
    }
    private void axMapControl1_OnMapReplaced(object sender, IMapControlEvents2_OnMap
ReplacedEvent e)
    {
        copyToPageLayout();
    }
    private void axMapControl1_OnAfterScreenDraw(object sender, IMapControlEvents2_
OnAfter ScreenDrawEvent e)
    {
        IActiveView activeView = (IActiveView)axPageLayoutControl1.ActiveView.FocusMap;
        IDisplayTransformation displayTransformation = activeView.ScreenDisplay.
DisplayTrans formation;
        displayTransformation.VisibleBounds = axMapControl1.Extent;
        axPageLayoutControl1.ActiveView.Refresh();
        copyToPageLayout();
    }
```

2.4.7 布局控件中属性设置与绘制元素

创建一个新 Windows 应用程序项目，命名为“lesson9”，在左边工具箱中拖放“PageLayoutControl”和“SymbologyControl”控件到窗体上，再添加 3 个“GroupBox”控件和 11 个“Button”控件，修改控件文本如图 2-22 所示。

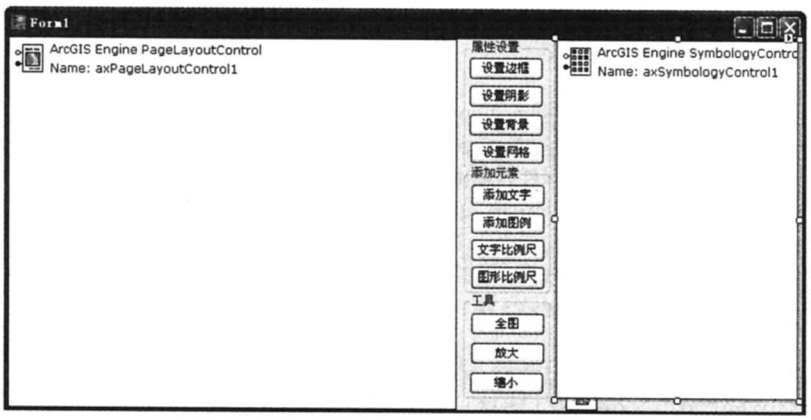


图 2-22 应用程序项目 lesson 9

在代码窗口中添加如下代码。

```
    int flag = 0;
    MapDocumentClass mapDocumentClass = new MapDocumentClass();
    private void Form1_Load(object sender, EventArgs e)
    {
        loadMapDocument();
        Microsoft.Win32.RegistryKey regKey = Microsoft.Win32.Registry.LocalMachine.
OpenSubKey("SOFT- WARE\\ESRI\\CoreRuntime", true);
        axSymbologyControl1.LoadStyleFile(regKey.GetValue("InstallDir")          +
"\\Styles\\ESRI.ServerStyle");
```

```
        axSymbologyControl1.GetStyleClass(esriSymbologyStyleClass.
esriStyleClassBackgrounds).Update();
        axSymbologyControl1.GetStyleClass(esriSymbologyStyleClass.
esriStyleClassBorders).Update();
        axSymbologyControl1.GetStyleClass(esriSymbologyStyleClass.
esriStyleClassShadows).Update();
    }
    //加载地图文档
    private void loadMapDocument()
    {
        System.Windows.Forms.OpenFileDialog openFileDialog;
        openFileDialog = new OpenFileDialog();
        openFileDialog.Title = "打开地图文档";
        openFileDialog.Filter = "map documents(*.mxd)|*.mxd";
        openFileDialog.ShowDialog();
        string filePath = openFileDialog.FileName;
        try
        {
            mapDocumentClass.Open(filePath, "");
            axPageLayoutControl1.PageLayout = mapDocumentClass.PageLayout;
            axPageLayoutControl1.Refresh();
        }
        catch (Exception ex )
        {
            MessageBox.Show(ex.ToString ());
        }
    }
    //设置边框
    private void button1_Click(object sender, EventArgs e)
    {
        axSymbologyControl1.StyleClass = esriSymbologyStyleClass.esriStyleClassBorders;
    }
    //设置阴影
    private void button2_Click(object sender, EventArgs e)
    {
        axSymbologyControl1.StyleClass = esriSymbologyStyleClass.esriStyleClassShadows;
    }
    //设置背景
    private void button3_Click(object sender, EventArgs e)
    {
        axSymbologyControl1.StyleClass = esriSymbologyStyleClass.esriStyleClassBackgrounds;
    }
    //设置网格
    private void button4_Click(object sender, EventArgs e)
    {
        IMap map;
        IActiveView activeView;
        activeView  = axPageLayoutControl1.PageLayout as IActiveView;
        map = activeView.FocusMap;
        IMapGrid mapGrid;
        IMeasuredGrid measuredGrid;
```

```
        measuredGrid = new MeasuredGridClass();
        mapGrid = measuredGrid as IMapGrid;
        measuredGrid.FixedOrigin = true;
        measuredGrid.Units = map.MapUnits;
        measuredGrid.XIntervalSize = 10;
        measuredGrid.YIntervalSize = 10;
        measuredGrid.XOrigin = -180;
        measuredGrid.YOrigin = -90;

        IProjectedGrid projectedGrid;
        projectedGrid = measuredGrid as IProjectedGrid;
        projectedGrid.SpatialReference = map.SpatialReference;
        mapGrid.Name = "Measured Grid";
        IMapFrame mapFrame;
        IGraphicsContainer graphicsContainer;
        graphicsContainer = activeView  as IGraphicsContainer;
        mapFrame  = graphicsContainer.FindFrame(map) as IMapFrame;
        IMapGrids mapGrids = mapFrame  as IMapGrids;
        mapGrids.AddMapGrid(mapGrid);
        activeView.PartialRefresh(esriViewDrawPhase.esriViewBackground, null, null);
    }
    private void axSymbologyControl1_OnItemSelected(object sender, ISymbologyControlEvents_
OnItemSelected Event e)
    {
        IStyleGalleryItem styleGalleryItem = (IStyleGalleryItem)e.styleGalleryItem;
        IFrameProperties frameProperties = (IFrameProperties)axPageLayoutControl1.
GraphicsContainer.Find Frame(axPageLayoutControl1.ActiveView.FocusMap);
        if (styleGalleryItem.Item is IBackground)
            frameProperties.Background = (IBackground)styleGalleryItem.Item;
        else if (styleGalleryItem.Item is IBorder)
            frameProperties.Border = (IBorder)styleGalleryItem.Item;
        else if (styleGalleryItem.Item is IShadow)
            frameProperties.Shadow = (IShadow)styleGalleryItem.Item;
        axPageLayoutControl1.Refresh(esriViewDrawPhase.esriViewBackground, null, null);
    }
    private void axPageLayoutControl1_OnMouseDown(object sender, IPageLayoutControlEvents_
OnMouse DownEvent e)
    {
        axPageLayoutControl1.MousePointer = esriControlsMousePointer. esriPointer-
Default;
        if (e.button == 1)
        {
            IEnvelope ipEnv;
            if (flag == 2)
            {
                axPageLayoutControl1.MousePointer = esriControlsMousePointer.esri-
PointerZoomIn;
                axPageLayoutControl1.Extent = axPageLayoutControl1. TrackRectan-
gle();
            }
            else if (flag == 1)
```

```
            {
                axPageLayoutControl1.MousePointer = esriControlsMousePointer. esri-
iPointerZoomOut;
                ipEnv = axPageLayoutControl1.Extent;
                ipEnv.Expand(2, 2, true);
                axPageLayoutControl1.Extent = ipEnv;
            }
        }
        else if (e.button == 2)
        {
            axPageLayoutControl1.MousePointer = esriControlsMousePointer. esri-
PointerPan;
            axPageLayoutControl1.Pan();
        }
    }
    private void button9_Click(object sender, EventArgs e)
    {
        axPageLayoutControl1.Extent = axPageLayoutControl1.FullExtent;
        axPageLayoutControl1.Refresh();
    }
    private void button10_Click(object sender, EventArgs e)
    {
        axPageLayoutControl1.MousePointer = esriControlsMousePointer.esriPointerZoomIn;
        flag = 2;
    }
    private void button11_Click(object sender, EventArgs e)
    {
        axPageLayoutControl1.MousePointer = esriControlsMousePointer.esriPointerZoomOut;
        flag = 1;
    }


    '添加文本，图例，图形比例尺，文字比例尺
    //添加文字
    private void button5_Click(object sender, EventArgs e)
    {
        IActiveView activeView ;
        IGraphicsContainer graphicsContainer;
        ITextElement  textElement;
        ITextSymbol textSymbol;
        IRgbColor  color;
        IElement element;
        IEnvelope envelope;

        activeView =axPageLayoutControl1.PageLayout as IActiveView  ;
        envelope =new  EnvelopeClass ();
        envelope .PutCoords (0,0,5,5);
        textElement =new TextElementClass ();
        element =textElement as IElement  ;
        element.Geometry =envelope ;
        textElement.Text ="我的地图";
```

```
        textSymbol  =new TextSymbolClass ();
        color =new RgbColorClass ();
        color.Green = 255;
        color.Blue = 255;
        color.Red = 0;
        textSymbol.Color =color as IColor  ;
        textSymbol.Size =30;
        textElement.Symbol =textSymbol ;
        graphicsContainer = activeView as IGraphicsContainer;
        graphicsContainer.AddElement (element ,0);
        axPageLayoutControl1.Refresh();
    }
    //添加图例
    private void button6_Click(object sender, EventArgs e)
    {
        UID uid;
        IEnvelope envelope;
        IMapSurround mapSurround;
        IGraphicsContainer graphicsContainer;
        IMapFrame mapFrame;
        IMapSurroundFrame mapSurroundFrame;
        IElement element;
        ITrackCancel trackCancel;

        uid = new UIDClass();
        uid.Value = "esriCarto.legend";
        envelope = new EnvelopeClass();
        envelope.PutCoords(1,1,2,2);
        graphicsContainer = axPageLayoutControl1.PageLayout as IGraphicsContainer ;
        mapFrame = graphicsContainer.FindFrame(axPageLayoutControl1.ActiveView. Focus
Map) as IMapFrame ;

        mapSurroundFrame = mapFrame.CreateSurroundFrame(uid, null );
        mapSurroundFrame.MapSurround.Name = "图例";
        element = mapSurroundFrame as IElement;
        element.Geometry = envelope;
        element.Activate(axPageLayoutControl1.ActiveView.ScreenDisplay);
        trackCancel = new CancelTrackerClass();
        element.Draw(axPageLayoutControl1.ActiveView.ScreenDisplay, trackCancel);
        graphicsContainer.AddElement(element, 0);
        axPageLayoutControl1.Refresh();
    }
    //文字比例尺 ScaleText
    private void button7_Click(object sender, EventArgs e)
    {
        UID uid;
        IEnvelope envelope;
        IMapSurround mapSurround;
        IGraphicsContainer graphicsContainer;
        IMapFrame mapFrame;
        IMapSurroundFrame mapSurroundFrame;
```

```
        IElement element;
        ITrackCancel trackCancel;
        uid = new UIDClass();
        uid.Value = "esriCarto.ScaleText";
        envelope = new EnvelopeClass();
        envelope.PutCoords(1, 1, 2, 2);
        graphicsContainer = axPageLayoutControl1.PageLayout as IGraphicsContainer;
        mapFrame = graphicsContainer.FindFrame(axPageLayoutControl1.ActiveView.
FocusMap) as IMapFrame;
        mapSurroundFrame = mapFrame.CreateSurroundFrame(uid, null);
        element = mapSurroundFrame as IElement;
        element.Geometry = envelope;
        element.Activate(axPageLayoutControl1.ActiveView.ScreenDisplay);
        trackCancel = new CancelTrackerClass();
        element.Draw(axPageLayoutControl1.ActiveView.ScreenDisplay, trackCancel);
        graphicsContainer.AddElement(element, 0);
        axPageLayoutControl1.Refresh();
    }
    //图形比例尺
    private void button8_Click(object sender, EventArgs e)
    {
        UID uid;
        IEnvelope envelope;
        IMapSurround mapSurround;
        IGraphicsContainer graphicsContainer;
        IMapFrame mapFrame;
        IMapSurroundFrame mapSurroundFrame;
        IElement element;
        ITrackCancel trackCancel;
        uid = new UIDClass();
        uid.Value = "esriCarto.ScaleLine";
        envelope = new EnvelopeClass();
        envelope.PutCoords(1, 1, 10, 2);
        graphicsContainer = axPageLayoutControl1.PageLayout as IGraphicsContainer;
        mapFrame = graphicsContainer.FindFrame(axPageLayoutControl1.ActiveView.
FocusMap) as IMapFrame;
        mapSurroundFrame = mapFrame.CreateSurroundFrame(uid, null);
        element = mapSurroundFrame as IElement;
        element.Geometry = envelope;
        element.Activate(axPageLayoutControl1.ActiveView.ScreenDisplay);
        trackCancel = new CancelTrackerClass();
        element.Draw(axPageLayoutControl1.ActiveView.ScreenDisplay, trackCancel);
        graphicsContainer.AddElement(element, 0);
        axPageLayoutControl1.Refresh();
    }
```

2.5 本章小结

本章主要介绍了 ArcEngine 提供的常用几个控件以及控件实现的接口，并提供了 9 个示例子演示这些控件的使用，本章例子的功能在以后的章节的实例程序中都将用到，初学者应理解这些程序，后面的章节中还会有更多的开发实例。