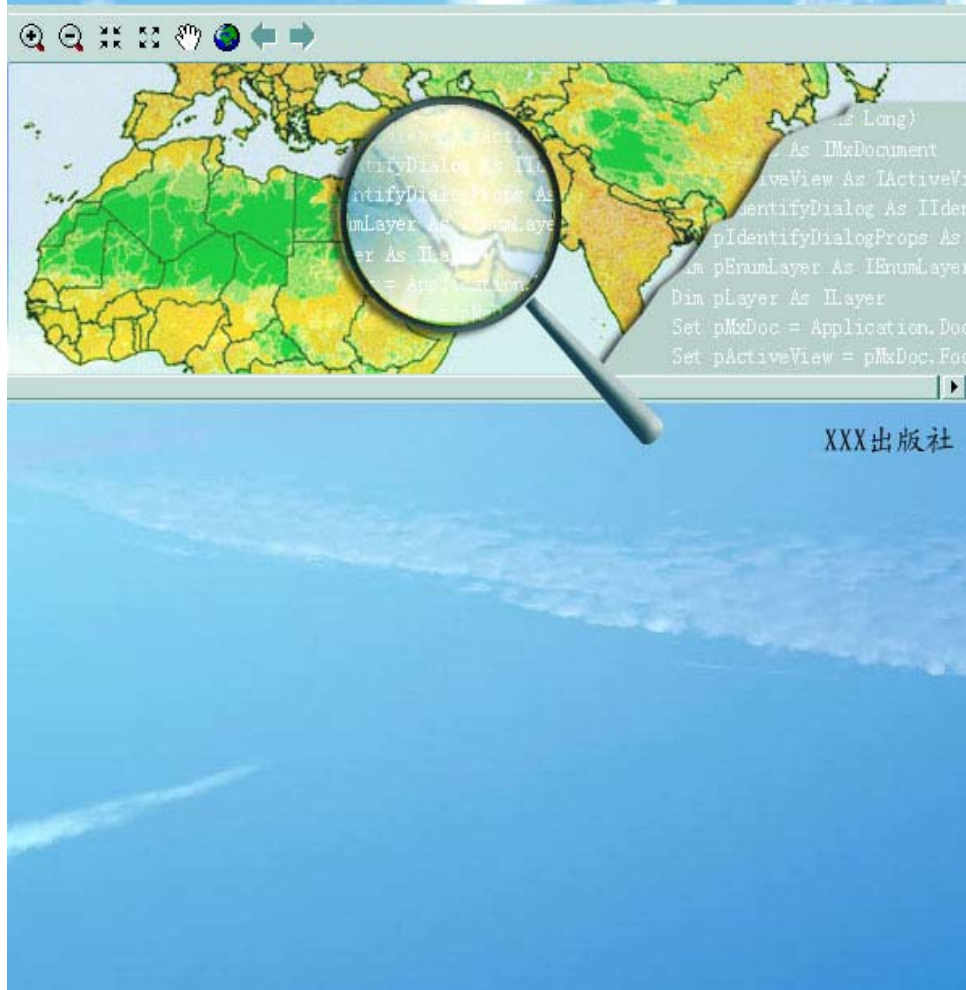


ArcGis

二次开发编程实例

超维信息技术有限公司 编著



ArcGIS

二次开发编程实例

超维空间信息技术有限公司 编著

X X X 出版社

内 容 提 要

本书通过大量的实例，从专业开发者的角度系统而详细地讲解了如何进行 ArcGIS 二次开发的编程，选材具有极强的针对性和实用性，内容翔实、基础、实用，旨在帮助开发人员能尽快掌握 ArcGIS 的二次开发。

全书分两部分：基础篇和提高篇。基础篇通过 100 多个具体的实例详细地讲解了 ArcGIS 二次开发过程中涉及到的各个主要的知识点；提高篇则以实际项目开发为例，综合运用基础篇的各个知识点，详细地展示了 ArcGIS 二次开发的流程、方法和各种开发技巧。

本书适合从事 ArcGIS 8.2/8.3 二次开发的工程技术人员阅读。

本书附带光盘一片，内容包括了书中全部实例的原码及测试数据。

前 言

目 录

前言

1. 基础篇	1
1.1. 开发环境	1
1.1.1. 如何在 ArcMap 的 VBA 环境中编程.....	1
1.1.2. 如何在 VB 环境中利用 ArcObjects 组件开发 ActiveX DLL ...	5
1.1.3. 如何在 ArcMap 中加载利用 ArcObjects 组件开发的 ActiveX DLL	7
1.1.4. 如何在 VB 环境中利用 ArcObjects 控件开发 EXE	8
1.2. 用户界面	10
1.2.1. 如何创建定制的按钮(Button).....	10
1.2.2. 如何创建定制的 Tool	12
1.2.3. 如何创建定制的工具条(Tool Bar).....	14
1.2.4. 如何创建定制的 MultiItem	16
1.2.5. 如何创建定制的菜单(Menu).....	18
1.2.6. 如何创建定制的 ToolControl	19
1.2.7. 如何创建定、使用制的可停靠窗口(Dockable Window).....	22
1.2.8. 如何创建、使用定制的 Extension	24
1.2.9. 如何使用状态条(StatusBar)与进度条(ProgressBar).....	25
1.2.10. 如何使用 ArcGIS 的对话框.....	27
1.2.11. 如何调用 ArcMap 中现有的功能.....	27
1.2.12. 如何创建放大镜(虫眼).....	28
1.3. GeoDataBase	29
1.3.1. 如何加载 Shape 文件.....	29
1.3.2. 如何在 ArcMap 中加入 Text 和 dBASE 文件.....	30
1.3.3. 如何连接 GeoDataBase 文件.....	32
1.3.4. 如何连接 Coverage 文件.....	34

1.3.5.	如何连接栅格文件.....	36
1.3.6.	如何创建 Shape 文件.....	37
1.3.7.	如何创建 DBF 文件.....	40
1.3.8.	如何创建 GeoDataBase 文件.....	42
1.3.9.	如何创建 Coverage 文件.....	43
1.3.10.	如何建立文件连接(Join / Link).....	45
1.3.11.	如何浏览纪录(属性查询).....	47
1.3.12.	如何编辑记录.....	48
1.3.13.	如何增加记录.....	49
1.3.14.	如何删除记录.....	51
1.3.15.	如何纪录排序(ITableSort).....	53
1.3.16.	如何添加字段.....	54
1.3.17.	如何删除字段.....	56
1.3.18.	如何进行空间查询.....	57
1.3.19.	如何进行高级空间查询(两个层之间的空间查询).....	59
1.3.20.	如何进行层与层之间的逻辑运算.....	60
1.3.21.	如何将 shape 文件转化成 GeoDataBase(各种文件格式的转换)	62
1.3.22.	如何将 Map 中显示的图形转化成栅格文件.....	65
1.3.23.	如何打开选中的层或独立表的属性窗口.....	66
1.3.24.	如何拷贝属性表中的一行.....	68
1.3.25.	如何为当前层或独立表创建一个 Summary 表.....	70
1.3.26.	如何利用用户定义的规则创建定制的排序.....	73
1.3.27.	如何实现在 ArcMap 上进行属性查询(Identify).....	79
1.3.28.	如何设置和修改层的数据源.....	82
1.4.	Display	83
1.4.1.	如何实现在 ArcMap 中放大缩小地图.....	83
1.4.2.	如何实现在 ArcMap 中移动地图.....	85
1.4.3.	如何实现在 ArcMap 上画 Polygon	87
1.4.4.	如何实现在 ArcMap 上进行测量.....	89

1.4.5.	如何在 ArcMap 上选取中记录.....	95
1.4.6.	如何在 ArcMap 中进行动作的撤销和重做.....	96
1.4.7.	如何画 Polygon Buffers	97
1.5.	图元编辑	99
1.5.1.	如何得到图形的基本属性.....	99
1.5.2.	如何将选中的点集转换成 Polygon	100
1.5.3.	如何将 Multipoint 转换成 Points	104
1.5.4.	如何通过 Polygon 中的多个 Ring 创建多个 Polygon	106
1.5.5.	如何从 Polyline 创建 Polygon	108
1.5.6.	如何从 Polygon 创建 Polyline	110
1.5.7.	如何将 Polygon/PolyCurve 一般化 (Generalize).....	112
1.5.8.	如何获得 Polygon 的中点.....	114
1.5.9.	如何判断图形间的逻辑运算.....	116
1.5.10.	如何进行图形间的逻辑运算.....	119
1.5.11.	如何创建 Envelope 的 Boundary	122
1.5.12.	如何通过鼠标移动图形.....	125
1.5.13.	如何为一个图形添加一个顶点.....	128
1.5.14.	如何删除一个图形上的一个顶点.....	131
1.5.15.	如何移动一个图形上的一个顶点.....	133
1.6.	Element	136
1.6.1.	如何创建 MarkerElement	136
1.6.2.	如何创建 TextElement	137
1.6.3.	如何创建 Balloon Callout	139
1.6.4.	如何创建 PolygonElement	140
1.6.5.	如何选中一个 Element	141
1.6.6.	如何移动 Element	142
1.6.7.	如何排列 Element	146
1.6.8.	如何通过名字查询 Element	148
1.6.9.	如何拷贝 Element	150
1.6.10.	如何沿着折线路径显示 Text	153

1.7.	Symbol 和 Renderer	154
1.7.1.	如何为一个层设置 Simple Renderer	154
1.7.2.	如何为一个层设置 UniqueValue Renderer	156
1.7.3.	如何为一个层设置 ClassBreaks Renderer	160
1.7.4.	如何为一个层设置 ProportionalSymbol Renderer	163
1.7.5.	如何为一个层设置 Chart Renderer	165
1.7.6.	如何为一个层设置 DotDensity Renderer	168
1.8.	Layout 和打印	170
1.8.1.	如何在 Page Layout 上添加 Text	170
1.8.2.	如何在 Page Layout 上添加 Legend	171
1.8.3.	如何在 Page Layout 上添加 North Arrow	174
1.8.4.	如何在 Page Layout 上添加 Scale bar	175
1.8.5.	如何在 Page Layout 上添加 Scale Text	177
1.8.6.	如何在 Page Layout 上添加 Picture	179
1.8.7.	如何创建、删除地图网格(Map Grid).....	180
1.8.8.	如何设置 Layout 中 MapFrame 的外观风格属性.....	182
1.8.9.	何设置 Layout 中 Page 的边框 (Border) 和背景 (Background)	184
1.8.10.	如何设置打印纸张的大小和方向.....	187
1.9.	坐标系统	188
1.9.1.	如何在 ArcMap 中设置地理坐标系和投影坐标系.....	188
1.9.2.	如何修改层的坐标系统.....	189
1.9.3.	如何把 Polygon 的顶点从经纬度坐标转换到平面直角坐标.	191
1.10.	ArcGis 相关文件	193
1.10.1.	如何夹载 grf 文件.....	193
1.10.2.	如何新建指向 Shape 文件的 lyr 文件.....	194
1.10.3.	如何新建指向 GeoDataBase 文件的 lyr 文件.....	195
1.10.4.	如何加载 mxd 文件.....	197
1.10.5.	如何加载 Apr 文件(ArcView32).....	198
1.10.6.	如何加载 lyr 文件.....	199

1.10.7.	lyr 文件的属性的设置	200
1.11.	其他	203
1.11.1.	如何创建简单的 Column Chart	203
1.11.2.	如何将数据输出到 Excel	204
1.11.3.	如何把 Labels 转换为 Annotation	206
1.11.4.	如何把 Annotation 转换为 Polygon Features	210
1.11.5.	如何设置 Featurelayer 的 Label	213
1.11.6.	如何设置图层显示的透明度.....	215
1.11.7.	如何过滤层中要显示的 Features	215
1.11.8.	如何在 MapControl 中新建一个 Document 并且保存.....	216
2.	提高篇	219
2.1.	缩略图的实现	219
2.2.	FeatureLayer 显示 Symbol 的定制	219
2.3.	空间查询的综合应用	219
2.4.	图形编辑的综合应用	219
2.5.	グラフの重ね合わせ表示と印刷	219
2.6.	バッファ処理	228
2.7.	Voronio 作成	234
2.8.	数据处理加速—地图分块处理	234
2.9.	MapControl 的使用	235
2.10.	运用 PageLayout 控件打印图形	235
附录	ArcGIS 的 GUID 一览表	235

1. 基础篇

1.1. 开发环境

1.1.1. 如何在 ArcMap 的 VBA 环境中编程

ArcMap 是 ArcGIS 家族的一员，它内置了一种集成编程环境—VBA (Visual Basic for Applications)。通过 VBA 编程，用户不但可以扩展 ArcMap 的菜单、工具条等，而且可以完成大多数用户的特定需求。

ArcMap 中 VBA 编程的方法有两种，一种是写 VBA 宏，另一种是创建 UIControl 并在其事件中写入实现用户需求的代码。下面列出两种方法的一般步骤。

方法一：写 VBA 宏（直接在 VBA 编辑器中编辑函数和过程）

1、如图 1，单击菜单栏中的<Tools>命令，选择<Macros>的<Visual Basic Editor>项，直接启动 ArcMap 的 VBA 编辑器；或者选择<Macros>的<Macros>项，进入如图 2 所示 Macro 对话框，在“Macro Name”文本框中输入要创建的宏的名称，并点<Create>按钮，启动 VBA 编辑器。

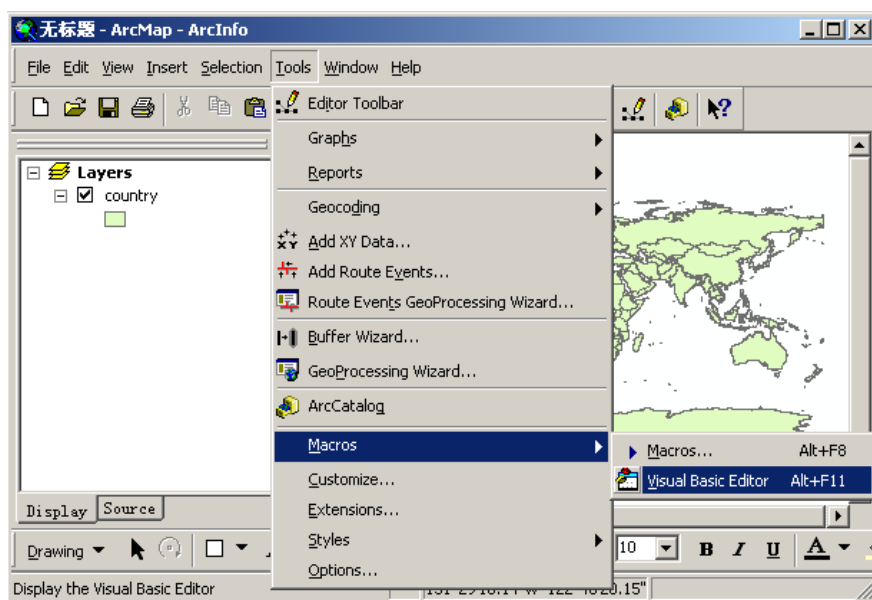


图 1 启动 Macro 对话框/启动 VBA 编辑器

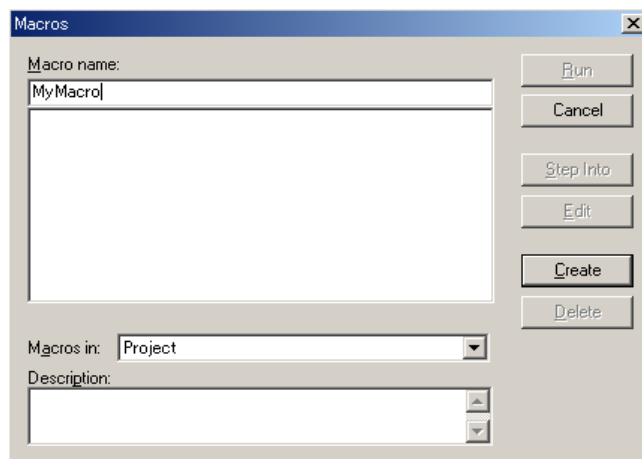


图 2 Macro 对话框

2、在图 3 所示的窗口中，用户可以根据实际选择在 Normal 节点或者 Project 节点的 ThisDocument、Forms、Modules 中编写宏（函数或过程），Normal 节点下所写的宏系统自动保存，除非用户删除，否则它将始终存在并在任何工程中都有效；而在 Project 节点下所写得宏随工程保存（如不保存工程，则宏也将不被保存），并只在工程中有效。

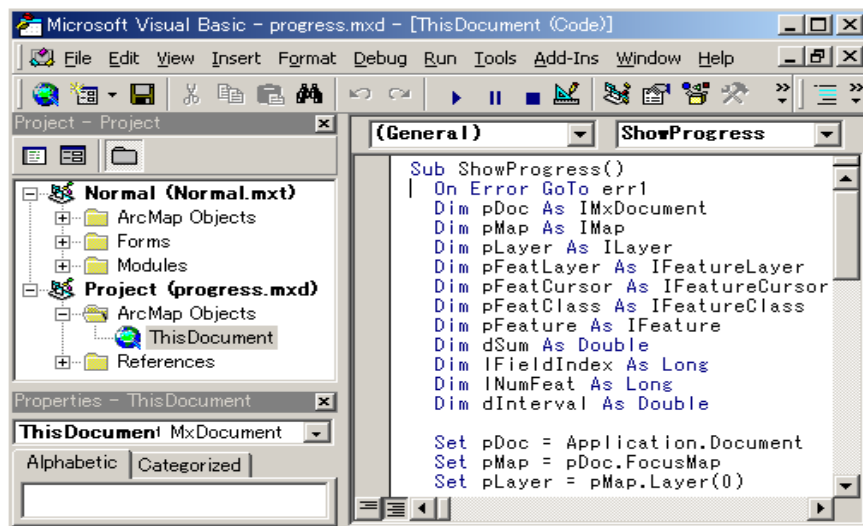



图 3 VBA 编辑器（VBE）

3、运行 VBA 宏

在 VBA 编辑器中写好 VBA 代码后，有两种方式运行：第一，点击 VBA 编辑器工具条中的 （运行）按钮，可立即运行写好的代码；第二，退出 VBA 编辑器，重新启动 Macro 对话框，如图 2，选择要运行的 VBA 宏名称，点击<Run>按钮即可运行相应的 VBA 宏。

方法二：创建 UIControl（交互式 VBA 编程）

1、用鼠标右击任何工具栏（条），在弹出的上托式菜单中选择<Customize>菜单项,如图 4，进入图 5 所示的 Customize 对话框。

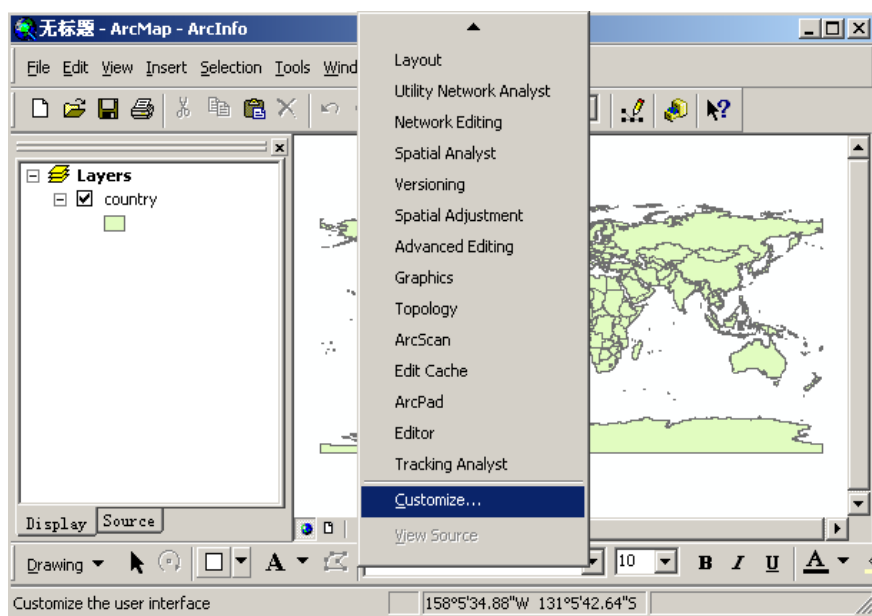


图 4 启动“Customize”对话框

2、切换到“Customize”对话框的“Commands”页，选中“UIControls”后点击<New UIControl>按钮，进入图 6 所示的“New UIControl”对话框。

3、在“New UIControl”对话框中，用户可根据需要选择 UIControl 类型：

UIButtonControl: 创建 Button;

UIToolControl: 创建与 Map 交互的 Tool;

UIMultiPageControl: 创建 MultiPage;

UIComboBoxControl: 创建 ComboBox。

最后点击<Create>按钮只创建 UIControl 或者点击<Create and Edit>按钮创建 UIControl 并进入 VBA 编辑器。与方法一不同, 此时应在 UIControl 的事件中进行 VBA 编程。

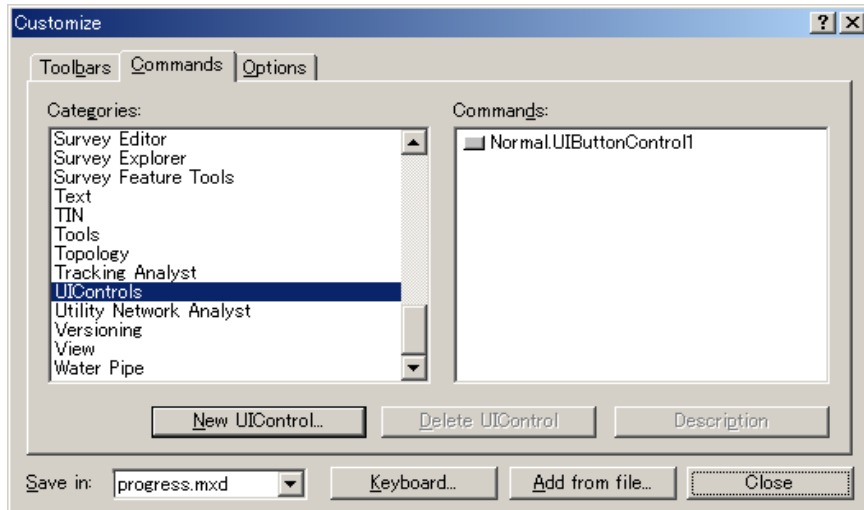


图 5 Customize 对话框

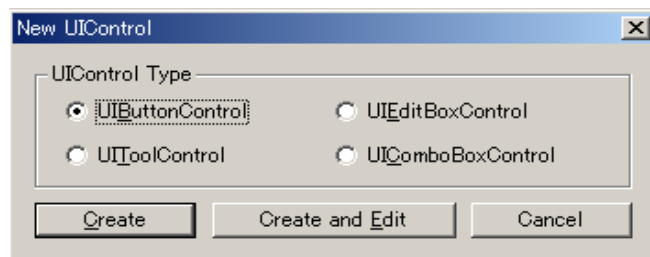


图 6 New UIControl 对话框

4、UIControl 创建后, 在图 5 所示的“Customize”对话框选中 UIControl 并将其拖置到任意工具条上, 用户便可象使用系统已有的 Control 一样使用所创建的 UIControl。

1.1.2. 如何在 VB 环境中利用 ArcObjects 组件开发 ActiveX DLL

1.1.1 节讨论了如何在 ArcGIS 的 VBA 环境中编程，虽然通过这种方式可以完成大多数用户的定制需求，但是，在某些情况下，对于特殊的应用，用户需要脱离 ArcGIS 环境而在 VB 开发环境中开发外部独立的应用程序，这种外部独立的应用程序有两种形式：ActiveX DLL 和 Standard EXE。Standard EXE 的开发将在 1.1.4 中讨论，本节将讨论 ActiveX DLL 的开发，其关键是引用 ArcObjects 对象库和实现 ArcObjects 接口（例如 ICommand, ITool, IToolBar 等）。

下面介绍在 VB 环境利用 ArcObjects 组件开发 ActiveX DLL 的一般步骤。

1、启动 VB 开发环境，在图 7 所示的“New Project”对话框中选择“ActiveX DLL”项，并点击<打开>按钮，进入 VBE 环境。

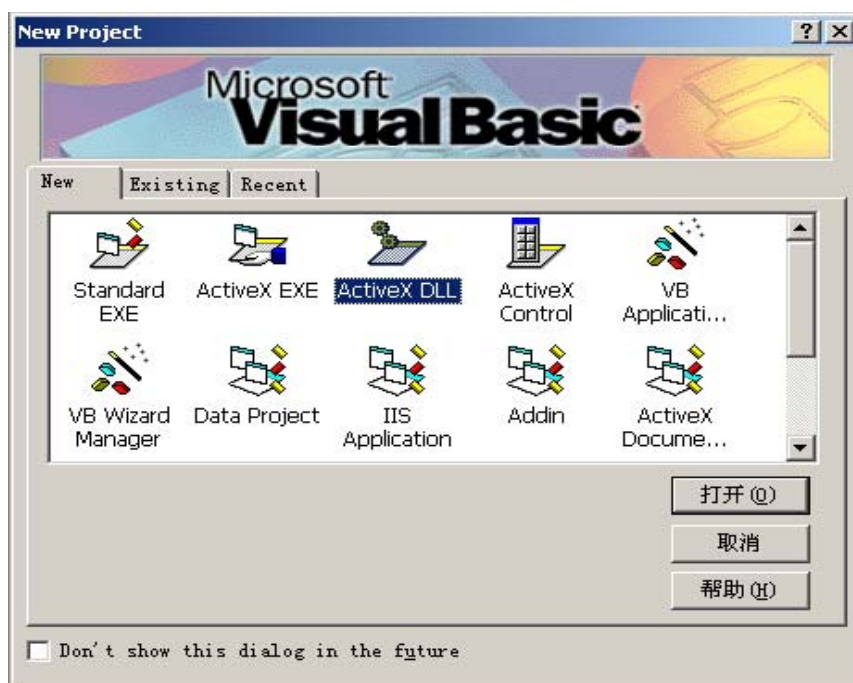


图 7 New Project 对话框

2、引用 ArcObjects 对象库：首先点击<Project>菜单中的<References>项，如图 8，进入对象库引用对话框，如图 9。

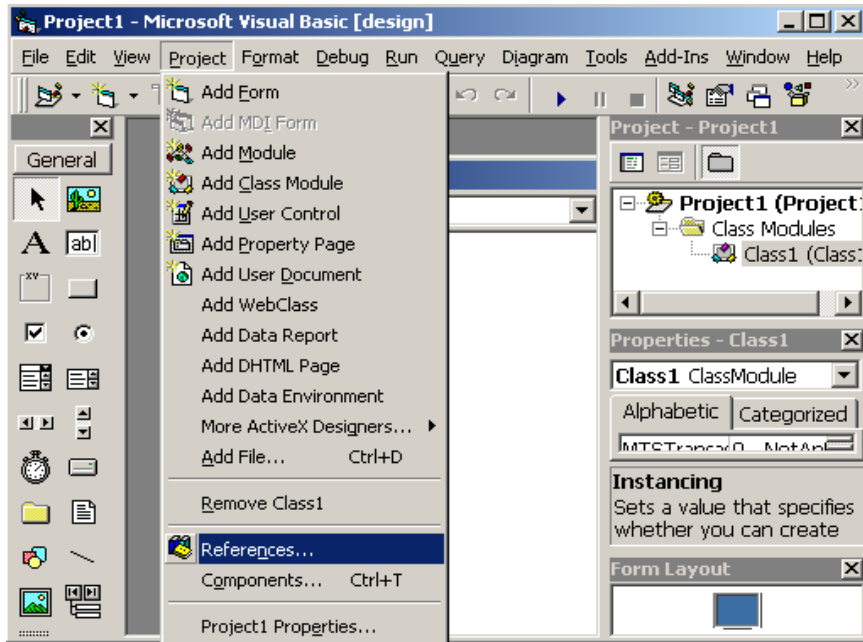


图 8 启动对象库引用对话框

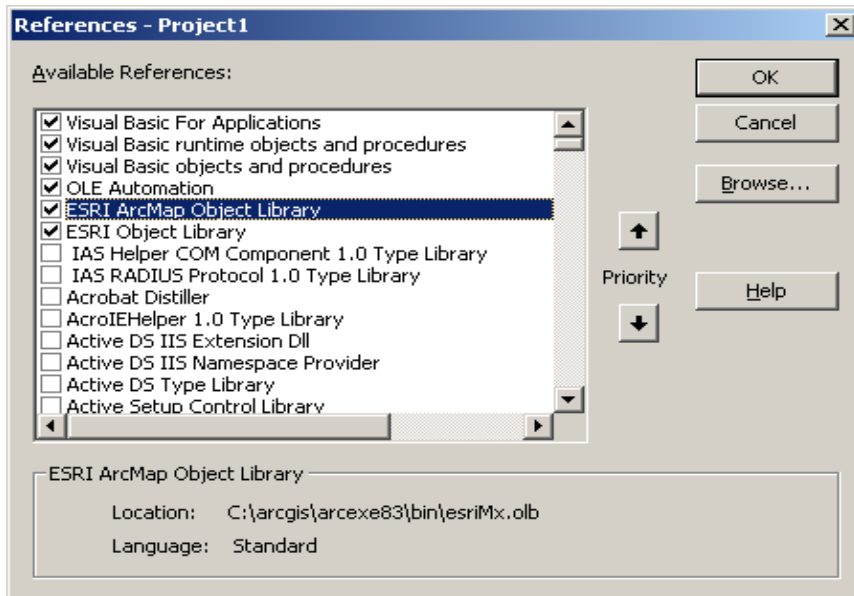


图 9 对象库引用对话框

3、对象库引用对话框（图 9）中选中“Esri ArcMap Object Library ”和“Esri Object Library” 两项，并点击<OK>按钮，返回 VBE 环境。

4、一般在类模块中写入实现特定 ArcObjects 接口的代码，如图 10，然后运行<File>菜单中的<Make project1.dll>项，生成 DLL 文件，如图 11。（project1.dll 随项目名改变）。

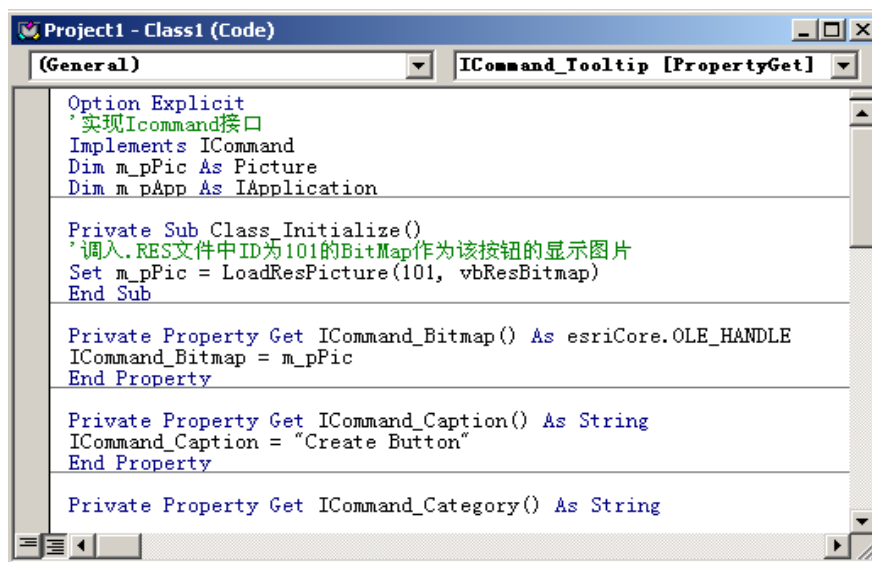


图 10 类模块编辑窗口

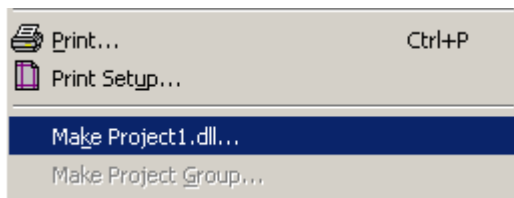


图 11 生成 DLL 文件

1.1.3. 如何在 ArcMap 中加载利用 ArcObjects 组件开发的 ActiveX DLL

用户通过 1.1.2 中介绍的方法开发好一个 ActiveX DLL 程序后，便可根据实际需要，在 ArcMap 环境下加载这个 ActiveX DLL 程序。其一般步骤如下：

1、用鼠标右击任何工具栏（条），点击弹出的上托式菜单中的<Customize>菜单项（参见图 4）。

2、在 Customize 对话框中，根据被加载 DLL 的类型切换到“Toolbars”或者“Commands”页（参见图 5），然后点击<Add From File>按钮。

3、在“打开文件”对话框中（Windows 通用“打开文件”对话框，图略），选择被加载的 Dll 文件，并点击<打开>按钮。

4、如果加载是“Commands”，则在图 5 所示的对话框中显示加载的 Command，并可以将其拖置于任何工具条上；如果加载是“ToolBars”，则在图 12 所示的对话框中显示加载的 ToolBar，选中后即可在 ArcMap 中显示。

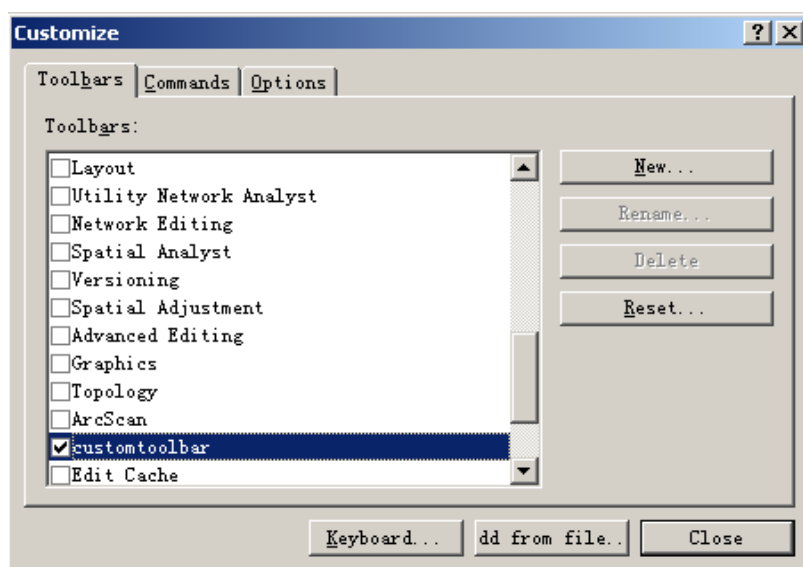


图 12 加载 ToolBar

1.1.4. 如何在 VB 环境中利用 ArcObjects 控件开发 EXE

利用 ArcObjects 控件开发 EXE 的前三步类似于 1.1.2 中开发“Active Dll”的前三步，唯一不同的是在“New Project”对话框中选择“Standard EXE”。

4、点击<Project>菜单项中的<Components>项，打开“Components”对话框，如图 13。

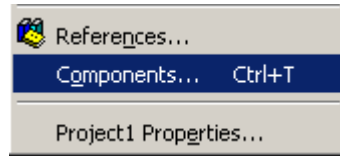


图 13 打开 Components 对话框

5、在“Components”对话框中，切换到 Controls 页，并选中“ESRI MapControl”项，点击<应用>或<确定>按钮，如图 14。

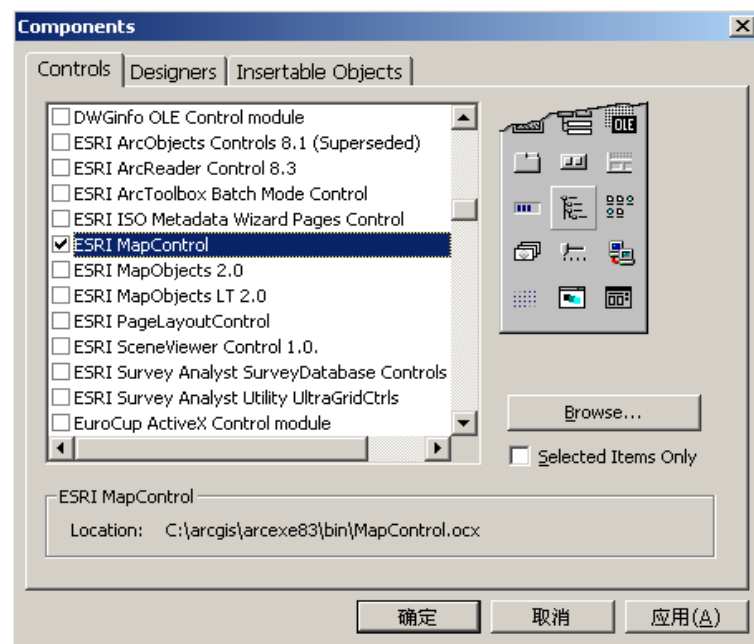


图 14 Components 对话框

6、如图 15 所示，加载 MapControl 控件之后，在 VBE 的控件面板中出现了 MapControl 控件图标，用户便可以象在 Form 中添加 Button 一样在 Form 中添加 MapControl 控件，并利用它开发 EXE。

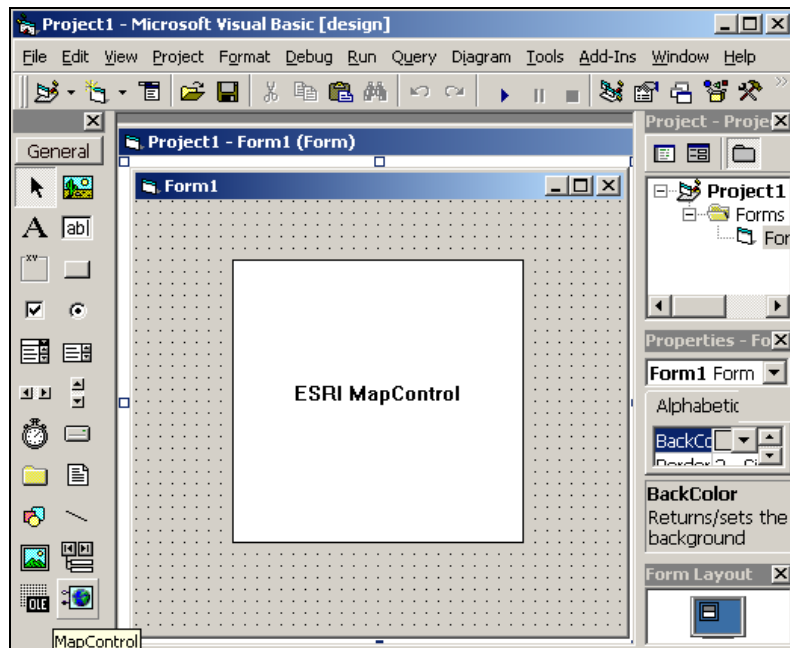


图 15 添加 MapControl 控件

1.2. 用户界面

1.2.1. 如何创建定制的按钮 (Button)

本例要实现的是如何创建定制的按钮 (Button)。

● 要点

用户通过在类模块中实现 ICommand 接口来创建定制的按钮 (COM command)。ICommand 接口包括 caption、name、category、bitmap、message(StatusBarr 的提示信息)、tooltip (微帮助)、help context id、help file、enabled 以及 checked 等十个属性和 OnCreate、OnClick 两个事件。从 Icommand 接口的 OnCreate 事件中获取的 ArcMap 的 Application 实例必须用一个公共变量保存, 以便在其它事件中(或者其它接口的事件中甚至整个工程中)使用。

- OnCreate 事件的参数 hook 传入的是一个 Object, 也就是 ArcMAP 的 Application 实例, 可把它赋给一个 IApplication 接口的变量, 便获得了 ArcMAP 的实例。

- 在 OnClick 事件中写入相关代码, 表示按下按钮时要实现的功能。

- 程序说明

程序在类模块中实现 ICommand 接口来创建自己的按钮(Button)

- 代码

```
Option Explicit
' 实现 Icommand 接口
Implements ICommand
Dim m_pPicture as Picture
Dim m_pApplication As IApplication

Private Sub Class_Initialize()
    ' 调入.RES 文件中 ID 为 101 的 BitMap 作为该按钮的显示图片
    Set m_pPicture = LoadResPicture(101, vbResBitmap)
End Sub

Private Property Get ICommand_Bitmap() As esriCore.OLE_HANDLE
    ICommand_Bitmap = m_pPicture
End Property

Private Property Get ICommand_Caption() As String
    ICommand_Caption = "Create Button"
End Property

Private Property Get ICommand_Category() As String
    ICommand_Category = " Create Button "
End Property

Private Property Get ICommand_Checked() As Boolean
End Property

Private Property Get ICommand_Enabled() As Boolean
    ICommand_Enabled = True
End Property

Private Property Get ICommand_HelpContextID() As Long
End Property

Private Property Get ICommand_HelpFile() As String
End Property

Private Property Get ICommand_Message() As String
End Property

Private Property Get ICommand_Name() As String
    ICommand_Name = " CreateButton "
End Property

Private Sub ICommand_OnClick()
    ' 加入按下按钮时实现的功能代码。在这里,
    ' 按钮按下时显示 ArcMap 的 Document 的 Tittle
    Dim pDocument As IDocument
```

```

        Set pDocument = m_pApplication.Document
        MsgBox pDocument.Title
    End Sub

    Private Sub ICommand_OnCreate(ByVal hook As Object)
        ' 获取 ArcMap 的 Application 实例
        Set m_pApplication = hook
    End Sub

    Private Property Get ICommand_Tooltip() As String
        ICommand_Tooltip = " Create Button "
    End Property

```

1.2.2. 如何创建定制的 Tool

本例要实现的是如何创建定制的 Tool

● 要点

用户在类模块中实现 ICommand（参见 1.2.1）和 ITool 接口。ITool 接口包括 mouse move, mouse button press/release, keyboard key press/release, double-click 以及 right click 等事件、Cursor 属性和 Refresh 方法。

Tool 既具有 Button 的功能，又具有与 ArcMAP 界面交互的功能，Button 的功能代码必须写在 ICommand 的 OnClick 事件中，而所有实现交互功能的代码必须写在 Itool 接口的各个事件中。Itool 接口的各个事件，用户可以在其中写入相关代码，表示用户与 ArcMAP 界面交互时一旦触发某事件要实现的功能。

● 程序说明

程序在类模块中实现 ICommand 和 Itool 接口来创建自己的 Tool。

● 代码

```

Option Explicit
' 实现 ICommand 和 Itool 接口
Implements ICommand
Implements ITool
Dim m_pApplication As IApplication
Dim m_pBitmap As IPictureDisp
Dim m_pCursor As IPictureDisp

Private Sub Class_Initialize()
    Set m_pBitmap = LoadResPicture(101, 0)
    ' 从 .RES 文件中调入 ID 为 102 的图片作为按下 Tool 后的 MouseCursor
    Set m_pCursor = LoadResPicture(102, 2)
End Sub

Private Property Get ICommand_Bitmap() As esriCore.OLE_HANDLE
    ICommand_Bitmap = m_pBitmap

```

```

End Property

Private Property Get ICommand_Caption() As String
    ICommand_Caption = "MyTool"
End Property

Private Property Get ICommand_Category() As String
    ICommand_Category = "MyCustomTools"
End Property

Private Property Get ICommand_Checked() As Boolean
End Property

Private Property Get ICommand_Enabled() As Boolean
    ICommand_Enabled = True
End Property

Private Property Get ICommand_HelpContextID() As Long
End Property

Private Property Get ICommand_HelpFile() As String
End Property

Private Property Get ICommand_Message() As String
    ICommand_Message = "This is my custom tool"
End Property

Private Property Get ICommand_Name() As String
    ICommand_Name = "MyCustomTool_MyTool"
End Property

Private Sub ICommand_OnClick()
    ' 加入按下按钮时实现的功能代码
    MsgBox "Clicked on my command"
End Sub

Private Sub ICommand_OnCreate(ByVal hook As Object)
    ' 获取 ArcMAP 的 Application 实例
    Set m_pApplication = hook
End Sub

Private Property Get ICommand_Tooltip() As String
    ICommand_Tooltip = "MyTool"
End Property

Private Property Get ITool_Cursor() As esriCore.OLE_HANDLE
    ITool_Cursor = m_pCursor
End Property

Private Function ITool_Deactivate() As Boolean
    ' 如果 ITool_Deactivate 设为 False, 则 Tool 不可用
    ITool_Deactivate = True
End Function

```

```

Private Function ITool_OnContextMenu(ByVal X As Long, ByVal Y As Long) As Bool an
    ' 在这里可以加入用户代码，点击 Mouse 右键时显示一个定制的 context menu
End Function

Private Sub ITool_OnDbClick()
    ' 在这里加入 Mouse 双击时的功能代码
End Sub

Private Sub ITool_OnKeyDown(ByVal keyCode As Long, ByVal Shift As Long)
End Sub

Private Sub ITool_OnKeyUp(ByVal keyCode As Long, ByVal Shift As Long)
End Sub

Private Sub ITool_OnMouseDown(ByVal Button As Long, ByVal Shift As Long, _
    ByVal X As Long, ByVal Y As Long)
    ' 加入 Mouse 单击时的功能代码
    If Button = 1 Then
        Dim pPoint As IPoint
        Dim pMxApplication As IMxApplication
        Set pMxApplication = m_pApp
        Set pPoint=pMxApplication.Display.DisplayTransformation.ToMapPoint(X, Y
        m_pApplication.StatusBar.Message(0) = Str(pPoint.X) & "," & Str(pPoint.Y)
    End If
End Sub

Private Sub ITool_OnMouseMove(ByVal Button As Long, ByVal Shift As Long, _
    ByVal X As Long, ByVal Y As Long)
    ' 加入 Mouse 移动时的功能代码
    m_pApplication.StatusBar.Message(0) = "ITool_OnMouseMove"
End Sub

Private Sub ITool_OnMouseUp(ByVal Button As Long, ByVal Shift As Long, _
    ByVal X As Long, ByVal Y As Long)
    ' 加入释放 Mouse 时的功能代码
    m_pApplication.StatusBar.Message(0) = "ITool_OnMouseUp"
End Sub

Private Sub ITool_Refresh(ByVal hDC As esriCore.OLE_HANDLE)
End Sub

```

1.2.3. 如何创建定制的工具条 (Tool Bar)

本例要实现的是如何创建定制的工具条 (Tool Bar)。就必须在类模块中实现 IToolBarDef 接口。IToolBarDef 接口包括 Caption、ItemCount 及 Name 三个属性和 GetItemInfo 方法。

● 要点

通过在类模块中实现 IToolBarDef 接口。IToolBarDef 接口包括 Caption、

ItemCount 及 Name 三个属性和 GetItemInfo 方法。

- ItemCount 属性表示 ToolBar 显示的条目 (Button、Tool 或其它控件) 数。
- GetItemInfo 方法定义工具条上各条目的 CLSID，其中，参数 *pos* 表示条目在 ToolBar 中的位置，*itemDef* 是定义相应位置的条目的 IItemDef 对象。
- 工具条条目的 CLSID 分为两种：

1、系统 CLSID，代表 ArcGIS 的一个功能，其引用方式为“esriCore. 命令名称”，如“esriCore.AddDataCommand”、“esriCore.FileSaveCommand”等。

2、用户定制 CLSID，表示用户自己定义的功能。其引用方式为“工程名称. 定制功能类名称”，如“ToolBarDef.ClsBar”。必须注意，这里“定制功能类名称”是工程中实现的一个功能类名称，“工程名称”即为当前工程的名称（不是 DLL 文件名，也不是工具条的名称），每次新建一个工程时，系统默认的工程名在某些情况下无法使用（在中文版的 VB 中是一个乱字符），必须改名后方能用。

● 程序说明

程序在类模块中实现 IToolBarDef 接口来创建自己的工具条 (ToolBar)。

● 代码

```
Option Explicit
Implements IToolBarDef

Private Property Get IToolBarDef_Caption() As String
    IToolBarDef_Caption = "CustomToolBar"
End Property

Private Sub IToolBarDef_GetItemInfo(ByVal pos As Long, ByVal itemDef As _
    esriCore.IItemDef)
    ' 这里假设在当前工程(工程名称为 ToolBarDef)中定义了一个类模块(名为 ClsBar),
    ' 它实现了 ICommand 接口(可参照 1.2.1)
    Select Case pos
    Case 0
        ' 用户自定义条目
        itemDef.ID = "ToolBarDef.ClsBar"
        itemDef.Group = False
    Case 1
        ' 系统条目
        itemDef.ID = "esriCore.AddDataCommand"
        itemDef.Group = False
    End Select
End Sub

Private Property Get IToolBarDef_ItemCount() As Long
    IToolBarDef_ItemCount = 2
End Property
```



```
Private Property Get IToolBarDef_Name() As String
    IToolBarDef_Name = "CustomToolBar"
End Property
```

1.2.4. 如何创建定制的 MultiItem

本例要实现的是如何创建定制的 MultiItem。

● 要点

需要实现 IMultiItem 接口，但不需要同时实现 ICommand 接口。IMultiItem 接口包括 Caption, itemCaption, ItemBitmap, ItemEnabled, ItemChecked, Message 及 Name 等属性和 OnItemClick, OnPopup 事件。

- itemCaption, ItemBitmap, ItemEnabled, ItemChecked 等属性的参数 index 表示当前 Item 的下标索引。

- OnPopup 事件的参数 hook 同 ICommand 接口的 OnCreate 事件的参数 hook 一样，传入 ArcGIS 的 Application 实例，同时，该事件返回将要显示的 Item 数目。

- OnItemClick 事件的参数 Index 表示用户当前点击的 Item 的索引，用户根据该索引分别定义点击各个 Item 时实现的功能。

● 程序说明

程序在类模块中实现 IMultiItem 接口来创建定制自己的 MultiItem。

● 代码

```
Option Explicit
Implements IMultiItem
Private m_pApp As IApplication
' ArcMap 的 Document
Private m_pMxDoc As IMxDocument
' 当前 Focus Map
Private m_pMap As IMap
' Map 中的层数
Private m_pLayerCnt As Long

Private Property Get IMultiItem_Caption() As String
    IMultiItem_Caption = "ZoomToLayers"
End Property

Private Property Get IMultiItem_HelpContextID() As Long
End Property

Private Property Get IMultiItem_HelpFile() As String
End Property
```

```

Private Property Get IMultiItem_ItemBitmap(ByVal Index As Long) As esriCore.OL_ _HANDLE
End Property

Private Property Get IMultiItem_ItemCaption(ByVal Index As Long) As String
    Dim i As Integer
    ' 遍历每一个层
    For i = 0 To m_pLayerCnt - 1
        ' 如果层号与当前 Item 的 Index 相同, 就设置该 Item 的 Caption
        If Index = i Then
            IMultiItem_ItemCaption = "Zoom to " & m_pMap.Layer(i).Name
        End If
    Next
End Property

Private Property Get IMultiItem_ItemChecked(ByVal Index As Long) As Boolean
End Property

Private Property Get IMultiItem_ItemEnabled(ByVal Index As Long) As Boolean
    Dim i As Integer
    ' 遍历每一个层
    For i = 0 To m_pLayerCnt - 1
        ' 如果层号与当前 Item 的 Index 相同, 则当前 Item 的 Enable 根据该层的 Visible 设置。
        If Index = i Then
            If m_pMap.Layer(i).Visible Then
                IMultiItem_ItemEnabled = True
            End If
        End If
    Next
End Property

Private Property Get IMultiItem_Message() As String
    IMultiItem_Message = "Zooms to the layer."
End Property

Private Property Get IMultiItem_Name() As String
    IMultiItem_Name = "ZoomMulti"
End Property

Private Sub IMultiItem_OnItemClick(ByVal Index As Long)
    Dim i As Integer
    Dim pEnv As IEnvelope
    Dim m_BookMark As IAOTBookmark
    ' 遍历每一个层
    For i = 0 To m_pLayerCnt - 1
        ' 如果层号与当前 Item 的 Index 相同, 则以该层的 AreaOfInterest 为范围执行 Zoom
        If Index = i Then
            Set pEnv = m_pMap.Layer(i).AreaOfInterest
            Set m_BookMark = New AOTBookmark
            Set m_BookMark.Location = pEnv
            m_BookMark.ZoomTo m_pMap
            m_pMxDoc.ActiveView.Refresh
        End If
    Next
End Sub

```

```

    Next
End Sub

Private Function IMultiItem_OnPopup(ByVal hook As Object) As Long
    Set m_pApp = hook
    ' 获取 Map 中的层数
    Set m_pMxDoc = m_pApp.Document
    Set m_pMap = m_pMxDoc.FocusMap
    m_pLayerCnt = m_pMap.LayerCount
    ' 显示的 Item 数等于层数
    IMultiItem_OnPopup = m_pLayerCnt
End Function

```

1.2.5. 如何创建定制的菜单(Menu)

本例要实现的是如何创建定制的菜单(Menu)。

● 要点

用户通过在类模块中实现 IMenuDef 接口来创建定制的菜单(Menu), 如果要使菜单出现在 Customize Dialog 的 Menus 类型中, 必须同时实现 IRootLevelMenu 接口, 它表明菜单为 root menu。IMenuDef 接口包括 Caption、ItemCount 及 Name 三个属性和 GetItemInfo 方法。类似 IToolBarDef(参照 1.2.3)

● 程序说明

程序在类模块中实现 IMenuDef 接口来创建定制的菜单(Menu)。

● 代码

```

Option Explicit

' Implement the IMenuDef interface and IRootLevelMenu interface
Implements IMenuDef
Implements IRootLevelMenu

Private Property Get IMenuDef_Caption() As String
    ' Set the string that appears as the menu's title
    IMenuDef_Caption = "MyMenu"
End Property

Private Sub IMenuDef_GetItemInfo(ByVal pos As Long, _
    ByVal itemDef As esriCore.IItemDef)
    ' Define the commands that will be on the menu. The built-in ArcMap
    ' Full Extent command, and Fixed Zoom In command are added to this custom menu.
    ' ID is the ClassID of the command. Group determines whether the command
    ' begins a new group on the menu
    Select Case pos
    Case 0
        itemDef.ID = "promenu.clsmultitem"
        itemDef.Group = False
    Case 1

```

```

        itemDef.ID = "esriCore.FullExtentCommand"
        itemDef.Group = True
    Case 2
        itemDef.ID = "esriCore.ZoomInFixedCommand"
        itemDef.Group = False
    End Select
End Sub

Private Property Get IMenuDef_ItemCount() As Long
    ' Set how many commands will be on the menu
    IMenuDef_ItemCount = 3
End Property

Private Property Get IMenuDef_Name() As String
    ' Set the internal name of the menu.
    IMenuDef_Name = "MyMenu"
End Property

```

1.2.6. 如何创建定制的 ToolControl

本例要实现的是如何创建定制的 ToolControl。ToolControl 是指具有 ComboBox 的下拉列表 或 EditBox 的编辑功能的一类控件。要创建定制的 ToolControl，必须在类模块中实现 ICommand 和 IToolControl 接口。IToolControl 接口包括 hWnd 属性和 OnDrop, OnFocus 事件。

●要点

- IToolControl 接口的 hWnd 属性，接受一个 Window Handle。
- IToolControl 接口的 OnDrop 事件，支持 ToolControl 的拖放，传入参数 *barType* 表示 Bar 类型。
- IToolControl 接口的 OnFocus 事件，传入 IcompletionNotify 类型的参数 *complete*，可以通过执行 IcompletionNotify 接口的 SetComplete 方法告之 ArcMAP，ToolControl 可以失去 Focus。

●程序说明

本例中涉及三个模块，详细描述如下，其中，在类模块中实现了 IToolBarDef 接口来创建自己的 ToolControl。

●代码

```

'1、frmImageCombo.frm 模块，定义选中 Combox 某一项之后实现的功能。要求在 Form 上放置一个
' ImageComb 控件（名为 ImageComb1）和一个 ImageList 控件（名为 ImageList1），并在 ImageList1
' 中添加三张图片。
Private Sub Form_Load()
    ' 设置 ImageComb1 的选择 Item
    Me.ImageComb1.ImageList = Me.ImageList1

```

```

Me.ImageCombo1.ComboItems.Add 1, "Red", "Red"
Me.ImageCombo1.ComboItems.Add 2, "Blue", "Blue"
Me.ImageCombo1.ComboItems.Add 3, "Green", "Green"
Me.ImageCombo1.ComboItems(1).Image = 1
Me.ImageCombo1.ComboItems(2).Image = 2
Me.ImageCombo1.ComboItems(3).Image = 3
End Sub

Private Sub ImageCombo1_Click()
    ' 选择颜色
    Dim sel As Variant
    sel = Me.ImageCombo1.SelectedItem
    Dim color As Variant
    Select Case sel
        Case "Blue"
            color = vbBlue
        Case "Red"
            color = vbRed
        Case "Green"
            color = vbGreen
    End Select
    Dim pDocument As IMxDocument
    Set pDocument = g_pApplication.Document
    ' 设置颜色
    Dim pRgbColor As IrgbColor
    Set pRgbColor = New RgbColor
    pRgbColor.RGB = color
    ' 改变选中部分的颜色
    Dim pSelectionEnvironment As ISelectionEnvironment
    Set pSelectionEnvironment = New SelectionEnvironment
    Set pSelectionEnvironment.DefaultColor = pRgbColor
    ' 刷新视图
    pDocument.ActivatedView.Refresh
    ' 通知 ArcMap, ToolControl 现在可以失去 Focus
    g_pCompletionNotify.SetComplete
End Sub

' 2、modPublicVars.bas 模块，定义工程中用到的全局变量。
Option Explicit
Public g_pApplication As IApplication
Public g_pCompletionNotify As IcompletionNotify

' 3、CustImageCombo.cls 模块，实现接口 ICommand 和 IToolControl。
Option Explicit
Implements ICommand
Implements IToolControl

Private Property Get ICommand_Bitmap() As esriCore.OLE_HANDLE
End Property

Private Property Get ICommand_Caption() As String
    ICommand_Caption = "Custom ImageCombo"
End Property

```

```

Private Property Get ICommand_Category() As String
    ICommand_Category = "Developer Samples"
End Property

Private Property Get ICommand_Checked() As Boolean
End Property

Private Property Get ICommand_Enabled() As Boolean
    ICommand_Enabled = True
End Property

Private Property Get ICommand_HelpContextID() As Long
End Property

Private Property Get ICommand_HelpFile() As String
End Property

Private Property Get ICommand_Message() As String
    ICommand_Message = "Change feature selection color"
End Property

Private Property Get ICommand_Name() As String
    ICommand_Name = "DeveloperSamples_CustomImageCombo"
End Property

Private Sub ICommand_OnClick()
End Sub

Private Sub ICommand_OnCreate(ByVal hook As Object)
    Set g_pApp = hook
End Sub

Private Property Get ICommand_Tooltip() As String
    ICommand_Tooltip = "Change Selection Color"
End Property

Private Property Get IToolControl_hWnd() As esriCore.OLE_HANDLE
    ' 将 frmImageCombo.ImageCombo1 的 Window Handle 赋给 IToolControl_hWnd
    IToolControl_hWnd = frmImageCombo.ImageCombo1.hWnd
End Property

Private Function IToolControl_OnDrop(ByVal barType As esriCore.esriCmdBarType) As Boolean
    ' 仅能将 ToolControl 拖放到 Toolbar 上
    If barType = esriCmdBarTypeToolbar Then
        IToolControl_OnDrop = True
    End If
End Function

Private Sub IToolControl_OnFocus(ByVal complete As esriCore.ICompletionNotify)
    Set g_pCompletionNotify = complete
End Sub

```

1.2.7. 如何创建定、使用制的可停靠窗口(Dockable Window)

本例要实现的是如何创建定制的可停靠窗口(Dockable Window)

●要点

用户通过在类模块中实现 IDockableWindowDef 接口来创建定制的可停靠窗口(Dockable Window)。IDockableWindowDef 接口包括 Caption、ChildHWND, UserData 及 Name 等属性和 OnCreate、OnDestroy 事件。

- ChildHWND 属性表示可停靠窗口包含的 Window 的 Handle。
- OnCreate 事件的参数 hook 传入 ArcGIS 的 Application 实例。
- 创建并注册可停靠窗口的步骤:

1、实现 IdockableWindowDef 接口(参见实例);

2、编译成 DLL;

3、调用 windows 目录下 system32 子目录下的 regsvr32.exe 用下面的形式注册编译好的 DLL:

```
win 目录\system32\regsvr32.exe <路径>\<文件名>.dll
```

4、运行<arcmap 目录>\arcexe81\Bin\categories.exe, 在打开的 Component Category Manager 中找到 ESRI Mx Dockable Window, 点击 Add Object...按钮将上面注册的DLL 文件加入, 并选中实现 IdockableWindowDef 接口的类名即可。

●程序说明

类模块 ClsDockableWindow 只是创建与注册可停靠窗口, 但还不能用, 还必须定义一个 IdockableWindow 接口的变量引用注册的类(必须用 IdockableWindowsManager 接口的 GetDockableWindow 获取, 其 ID 号用“实现 IdockableWindowDef 接口的工程名 project1. 实现 IdockableWindowDef 接口的类名 class1”)。

●代码

```
' 类模块 ClsDockableWindow
Option Explicit
Implements IDockableWindowDef
Dim m_Application As IApplication

Private Property Get IDockableWindowDef_Caption() As String
    IDockableWindowDef_Caption = "Dockable Window"
End Property
```

```

Private Property Get IDockableWindowDef_ChildHWND() As esriCore.OLE_HANDLE
    ' 将 FrmDWin 窗口的 Handle 赋给 IDockableWindowDef_ChildHWND
    IDockableWindowDef_ChildHWND = FrmDWin.hWnd
End Property

Private Property Get IDockableWindowDef_Name() As String
    IDockableWindowDef_Name = "docwin"
End Property

Private Sub IDockableWindowDef_OnCreate(ByVal hook As Object)
    Set m_pApplication = hook
End Sub

Private Sub IDockableWindowDef_OnDestroy()
    Set m_pApplication = Nothing
End Sub

Private Property Get IDockableWindowDef_UserData() As Variant
End Property

' 类模块 class1
Option Explicit
Implements ICommand
Dim m_pApp As IApplication
Dim m_pDWMgr As IDockableWindowManager
Dim m_pDWin As IDockableWindow

Private Property Get ICommand_Bitmap() As esriCore.OLE_HANDLE
End Property

Private Property Get ICommand_Caption() As String
    ICommand_Caption = "Dockable Window"
End Property

Private Property Get ICommand_Category() As String
    ICommand_Category = "Dockable Window"
End Property

Private Property Get ICommand_Checked() As Boolean
End Property

Private Property Get ICommand_Enabled() As Boolean
    ICommand_Enabled = True
End Property

Private Property Get ICommand_HelpContextID() As Long
End Property

Private Property Get ICommand_HelpFile() As String
End Property

Private Property Get ICommand_Message() As String
End Property

```



```

Private Property Get ICommand_Name() As String
    ICommand_Name = "DocWin"
End Property

Private Sub ICommand_OnClick()
    m_pDWin.Show Not m_pDWin.IsVisible
End Sub

Private Sub ICommand_OnCreate(ByVal hook As Object)
    Set m_pApp = hook
    ' QI(Dockable Window)
    Set m_pDWMgr = hook
    Dim pid As New UID
    pid.Value = "Prodockablewindow.Clsdockablewindow"
    Set m_pDWin = m_pDWMgr.GetDockableWindow(pid)
End Sub

Private Property Get ICommand_Tooltip() As String
    ICommand_Tooltip = "Dockable Window"
End Property

```

1.2.8. 如何创建、使用定制的 Extension

本例要实现的是如何创建、使用定制的 Extension。

●要点

用户要实现 IExtension 接口来创建定制的 Extension。 IExtension 接口包括 Name 属性和 startup 和 shutdown 事件。

- 创建并注册 Extension 的步骤:

1. 实现 IExtension 接口;
2. 编译成 DLL;
3. 调用 windows 目录下 system32 子目录下的 regsvr32.exe 用下面的形式注册编译好的 DLL

win 目录\system32\regsvr32.exe <路径>\<文件名>.dll

4. 运行<arcmap 目录>\arcexe81\Bin\categories.exe, 在打开的 Component Catregory Manager 中找到 ESRI Mx Extensions, 点击 Add Object...按钮将上面注册的 DLL 文件加入, 并选中实现 IExtension 接口的类名即可。

●程序说明

用户通过在类模块中实现 IExtension 接口来创建定制的 Extension。 Extension 将在 ArcMap 打开时自动加载, 在 ArcMap 关闭时自动卸载。

● 代码

```
Option Explicit
Implements IExtension
Dim m_pApplication As IApplication
' Need to listen for the MxDocument events
Dim WithEvents m_pDocument As MxDocument

Private Property Get IExtension_Name() As String
    IExtension_Name = "My Extension"
End Property

Private Sub IExtension_Shutdown()
    ' Clear the reference to the Application and MxDocument
    Set m_pApplication = Nothing
    Set m_pDocument = Nothing
End Sub

Private Sub IExtension_Startup(initializationData As Variant)
    ' This extension is an ArcMap Extension. When this extension is loaded on
    ' ArcMap startup, initializationData is passed in as a reference to the
    ' Application object
    Set m_pApplication = initializationData

    'Start listening for the MxDocument events.
    Set m_pDocument = m_pApp.Document
End Sub

Private Function m_pDocument_NewDocument() As Boolean
    ' Do something when a new document is created
    MsgBox "Creating a new document."
End Function

Private Function m_pDocument_OpenDocument() As Boolean
    ' So something when a document is opened.
    MsgBox "Opening a document"
End Function
```

1.2.9. 如何使用状态条(StatusBar)与进度条(ProgressBar)

本例要演示的是如何使用状态条(StatusBar)与进度条(ProgressBar)。实现后的结果为在 ArcMap 中，状态条位于其底部，它显示 ArcMAP 当前状态的信息，包含进度条。

● 要点

一般情况下，通过 ArcMAP 的 Application 实例获取 IstatusBar 的实例，然后再通过 StatusBar 获取 Iprogressbar 的实例，并将 Iprogressbar 的实例赋给 IstepProgressor 类型的变量。

● 程序说明

运行函数 ShowProgress 将在 ArcMap 的下方添加一个状态条(StatusBar)和进度条(ProgressBar)。

● 代码

```
Sub ShowProgress()  
    On Error GoTo err1  
    Dim pDocument As IMxDocument  
    Dim mMap As IMap  
    Dim pLayer As ILayer  
    Dim pFeatureLayer As IFeatureLayer  
    Dim pFeatureCursor As IFeatureCursor  
    Dim pFeatureClass As IFeatureClass  
    Dim pFeature As IFeature  
    Dim dSum As Double  
    Dim lFieldIndex As Long  
    Dim lNumFeat As Long  
    Dim dInterval As Double  
  
    Set pDocument = Application.Document  
    Set mMap = pDocument.FocusMap  
    Set pLayer = mMap.Layer(0)  
    Set pFeatureLayer = pLayer  
    Set pFeatureClass = pFeatureLayer.FeatureClass  
    Set pFeatureCursor = pFeatureLayer.Search(Nothing, True)  
    Dim pStatusBar As IStatusBar  
    Set pStatusBar = Application.StatusBar  
    Dim pStepProgressor As IStepProgressor  
    Set pStepProgressor = pStatusBar.ProgressBar  
    lNumFeat = pFeatureClass.FeatureCount(Nothing)  
    dInterval = lNumFeat / 100  
    Set pFeature = pFeatureCursor.NextFeature  
    ' 字段名"FID"用户根据实际情况而改变  
    lFieldIndex = pFeature.Fields.FindField("FID")  
    Dim PauseTime, Start, Finish, TotalTime, i  
    PauseTime = 0.5  
    pStepProgressor.MinRange = 1  
    pStepProgressor.MaxRange = lNumFeat  
    pStepProgressor.StepValue = dInterval  
    For i = 1 To lNumFeat  
        dSum = dSum + pFeature.Value(lFieldIndex)  
        Set pFeature = Nothing  
        Set pFeature = pFeatureCursor.NextFeature  
        pStepProgressor.Position = i  
        pStepProgressor.Message = "Reading record " & Str(i) & ". Sum =" & Str(dSum)  
        pStepProgressor.Step  
        pStepProgressor.Show  
        Start = Timer  
        Do While Timer < Start + PauseTime  
            DoEvents  
        Loop  
    Next i  
    TotalTime = Timer - Start  
    MsgBox "Total Time: " & Str(TotalTime)
```

```

Next
pStepProgressor.Hide
Exit Sub
err1:
MsgBox Err.Description
End Sub

```

1. 2. 10. 如何使用 ArcGIS 的对话框

添加对话框可以通过相应的接口实现。比如“添加数据对话框”使用 IaddDataDialog 接口，“生成点坐标对话框”使用 ICoordinateDialog 接口，“生成字符串对话框”使用 IGetStringDialog 接口，“生成数值对话框”使用 INumberDialog 接口等等。本例以添加数据对话框 (Add Data Dialog) 为例，讲述对话框是如何通过接口实现添加的。

● 要点

用户通过实现 IaddDataDialog 接口来创建定制的添加数据对话框，IaddDataDialog 接口包括 Document 和 Map 属性和 Show 事件。

● 程序说明

在程序中除了必须生成 IaddDataDialog 接口的实例外，还必须指定对话框的 Document 和 Map。当为 AddDataDialog 指定 Document 和 Map 之后，系统会自动将用户选择的数据加入到指定 Document 和 Map 中。最后实现在 ArcMap 中添加数据的对话框。

● 代码

```

Sub ShowProgress()
Dim mDocument As IMxDocument
Dim mAddDataDialog As IAddDataDialog
Set mAddDataDialog = New AddDataDialog
Set mDocument = ThisDocument
mAddDataDialog.Document = mDocument
mAddDataDialog.Map = mDocument.FocusMap
mAddDataDialog.Show Application.hWnd, True
End Sub

```

1. 2. 11. 如何调用 ArcMap 中现有的功能

如何调用 ArcMap 中现有的功能，比如菜单栏、工具栏中的某些功能。这些都可以通过 UID 来实现。本例是通过 UID 调用“另存为”功能。

可以通过两种方法得到 UID：

方法一：运用 ArcID 模块

- 要点

通过 ArcID 获得 UID, ArcID 是 ArcMap 的 VBA 中的模块。只需要知道要调用功能的名称运用代码就可以实现。

- 程序说明

程序通过运用 ArcID 模块和命令名称来实现调用“另存为”的功能。

- 代码

```
Sub ExecuteCmd()  
    Dim pCommandItem As ICommandItem  
    ' Use ArcID module and the Name of the SaveAs command  
    Set pCommandItem = Application.Document.CommandBars.Find(arcid.File_SaveAs)  
    pCommandItem.Execute  
End Sub
```

方法二：直接写代码

- 要点

通过直接写代码获得 UID 实现调用功能。

- 程序说明

写入文件菜单项的 GUID (CLSID 或 ProgID) 来调用文件菜单项, 同时还需要通过设置 Subtype 的值来调用文件菜单项的“另存为”功能。

- 代码

```
Sub ExecuteCmd2()  
    Dim pUID As New UID  
    Dim pCommandItem As ICommandItem  
    ' Use the GUID of the Save command  
    pUID.Value = "{119591DB-0255-11D2-8D20-080009EE4E51}"  
    ' or you can use the ProgID  
    ' pUID.Value = "esriCore.MxFileMenuItem"  
    pUID.SubType = 3  
    Set pCommandItem = Application.Document.CommandBars.Find(pUID)  
    pCommandItem.Execute  
End Sub
```

1. 2. 12. 如何创建放大镜(虫眼)

本例要实现的是如何创建放大镜(虫眼), 将所选区域放大一定的倍数。

- 要点

用户通过定义 IMapInset、IMapInsetWindow、IDataWindowFactory 三个接口, 运用它们的方法、属性来创建放大镜(虫眼)。

- 程序说明

运用这个子程序生成了一个新的放大镜窗口, 在本例中将放大率设定为 200%

代替原来的 400%。

- 代码

```
Public Sub CreateMagnifierWindow()  
    Dim pMapInset As IMapInset  
    Dim pMapInsetWindow As IMapInsetWindow  
    Dim pDataWindowFactory As IDataWindowFactory  
  
    Set pDataWindowFactory = New MapInsetWindowFactory  
    If pDataWindowFactory.CanCreate(Application) Then  
        Set pMapInsetWindow = pDataWindowFactory.Create(Application)  
        Set pMapInset = pMapInsetWindow.MapInset  
        'Set the zoom percent to 200%  
        pMapInset.ZoomPercent = 200  
        pMapInsetWindow.Show True  
    End If  
End Sub
```

1.3. GeoDataBase

1.3.1. 如何加载 Shape 文件

本例实现的是在 ArcMap 中连接指定的 Shape 文件，并将其加载到当前激活的 Map 中。

- 要点

通过 FeatureLayer 类实现 IFeatureLayer 接口对象，设置 IFeatureLayer.FeatureClass 属性和 Name 属性，使用 IMap.AddLayer 方法将新层添加到当前地图。利用 IWorkspaceFactory 接口、IFeatureWorkspace 接口和 IFeatureLayer 接口实现连接 Shape 文件

- 程序说明

函数 OpenShapeFile 根据输入的 Shape 文件路径 sFilePath，将文件名为 sFileName 的 Shape 文件连接到当前激活的 Map 中去。

- 代码

```
Private Sub OpenShapeFile(ByVal sFilePath As String, ByVal sFileName As String)  
    Dim pWorkspaceFactory As IWorkspaceFactory  
    Dim pFeatureWorkspace As IFeatureWorkspace  
    Dim pFeatureLayer As IFeatureLayer  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim sDir As String  
  
    On Error GoTo ErrorHandler:  
    sDir = Dir(sFilePath & "\" & sFileName & ".shp")  
    If (sDir = "") Then  
        sDir = Dir(sFilePath & "\" & sFileName)  
        If (sDir = "") Then  
            MsgBox ("文件不存在")  
        End If  
    End If  
End Sub
```

```

        Exit Sub
    End If
End If

'Create a new ShapefileWorkspaceFactory object and open a shapefile folder
Set pWorkspaceFactory = New ShapefileWorkspaceFactory
Set pFeatureWorkspace = pWorkspaceFactory.OpenFromFile(sFilePath, 0)

'Create a new FeatureLayer and assign a shapefile to it
Set pFeatureLayer = New FeatureLayer
Set pFeatureLayer.FeatureClass = pFeatureWorkspace.OpenFeatureClass(sFileName)
pFeatureLayer.Name = pFeatureLayer.FeatureClass.AliasName

'Add the FeatureLayer to the focus map
Set pMxdDocument = Application.Document
Set pMap = pMxdDocument.FocusMap
pMap.AddLayer pFeatureLayer
Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

Private Sub UIButtonControl1_Click()
    Dim pVBProject As VBProject
On Error GoTo ErrorHandler:
    Set pVBProject = ThisDocument.VBProject
    OpenShapeFile pVBProject.FileName & "..\..\..\\" & "\data\", "Continents
    Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

```

1.3.2. 如何在 ArcMap 中加入 Text 和 dBASE 文件

本例实现的是如何在当前的 ArcMap 中加入 Text 文件和 dBASE 文件。

● 要点

首先为 Text 文件或 dBASE 文件创建一个与之对应的 ITable 接口对象，然后通过 IMap 实例获得 IStandaloneTable 接口对象和 IStandaloneTableCollection 接口对象，并设置其属性，最后使用 IStandaloneTableCollection.AddStandaloneTable 方法将 Text 文件或 dBASE 文件加入到当前的 ArcMap 中。加入 Text 文件或 dBASE 文件的区别仅在于创建 ITable 对象时 IWorkspaceFactory 的类型不同，加入 Text 文件时是 TextFileWorkspaceFactory 类型，加入 dBASE 文件时是 ShapefileWorkspaceFactory 类型。

主要用到了 IWorkspaceFactory 接口，IWorkspace 接口，IFeatureWorkspace

接口, ITable 接口, IStandaloneTable 接口和 IStandaloneTableCollection 接口。

- 程序说明

函数 AddTextFile 通过文件路径 sFilePath 和文件名 sFileName 找到 Text 文件并为其创建 ITable 对象

函数 AddDBASEFile 通过文件路径 sFilePath 和文件名 sFileName 找到 dBASE 文件并为其创建 ITable 对象

函数 Add_Table_TOC 将 ITable 对象 pTable 加入到当前的 ArcMap 中。

- 代码

```
Private Sub AddTextFile(ByVal sFilePath As String, ByVal sFileName As String)
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pWorkspace As IWorkspace
    Dim pFeatureWorkspace As IFeatureWorkspace
    Dim pTable As ITable
    Dim sDir As String

    On Error GoTo ErrorHandler:
        sDir = Dir(sFilePath & sFileName & ".txt")
        If (sDir = "") Then
            MsgBox (sFileName & ".txt" & " 文件不存在")
            Exit Sub
        End If

        'Get the ITable from the geodatabase
        Set pWorkspaceFactory = New TextFileWorkspaceFactory
        Set pWorkspace = pWorkspaceFactory.OpenFromFile(sFilePath, 0)
        Set pFeatureWorkspace = pWorkspace
        Set pTable = pFeatureWorkspace.OpenTable(sFileName & ".txt")

        'Add the table
        Add_Table_TOC pTable
        Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub AddDBASEFile(ByVal sFilePath As String, ByVal sFileName As String)
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pWorkspace As IWorkspace
    Dim pFeatureWorkspace As IFeatureWorkspace
    Dim pTable As ITable

    On Error GoTo ErrorHandler:
        'Get the ITable from the geodatabase
        Set pWorkspaceFactory = New ShapefileWorkspaceFactory
        Set pWorkspace = pWorkspaceFactory.OpenFromFile(sFilePath, 0)
        Set pFeatureWorkspace = pWorkspace
```



```

        Set pTable = pFeatureWorkspace.OpenTable(sFileName)
        'Add the table
        Add_Table_TOC pTable
        Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub Add_Table_TOC(pTable As ITable)
    Dim pDoc As IMxDocument
    Dim pMap As IMap
    Dim pStandaloneTable As IStandaloneTable
    Dim pStandaloneTableC As IStandaloneTableCollection

    On Error GoTo ErrorHandler:
        Set pDoc = ThisDocument
        Set pMap = pDoc.FocusMap

        'Create a new standalone table and add it
        'to the collection of the focus map
        Set pStandaloneTable = New StandaloneTable
        Set pStandaloneTable.Table = pTable
        Set pStandaloneTableC = pMap
        pStandaloneTableC.AddStandaloneTable pStandaloneTable

        'Refresh the TOC
        pDoc.UpdateContents
        Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControll1_Click()
    Dim pVBProject As VBProject

    On Error GoTo ErrorHandler:
        Set pVBProject = ThisDocument.VBProject

        'Add text file to ArcMap. Dont include .txt extension
        AddTextFile pVBProject.FileName & "..\..\..\.." & "\data\", "Continents"
        'Add dBASE file to ArcMap
        AddDBASEFile pVBProject.FileName & "..\..\..\.." & "\data\", "Continents"
        Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.3. 如何连接 GeoDataBase 文件

本例实现的是连接一个 GeoDataBase 文件，并在 ArcMap 中加载该 GeoDataBase 文件的一个表。

- 要点

定义 IWorkspaceFactory 接口对象, 使用 AccessWorkspaceFactory 类实现之。再创建 IFeatureLayer 接口对象, 用 IFeatureWorkspace.OpenFeatureClass 方法加载 GeoDataBase 文件的一个表到 IFeatureLayer.FeatureClass 对象中。最后用 IMap.AddLayer 方法将新层添加到当前地图。

使用接口有: IWorkspaceFactory 接口、IFeatureWorkspace 接口、IFeatureLayer 接口和 IMap 接口。

- 程序说明

函数 OpenGeoDataBaseFile 根据输入的 GeoDataBase 文件的路径(带文件名及后缀) sAllFileName 连接 GeoDataBase 文件, 再根据输入的 GeoDataBase 文件中的某表表名 sTableName 加载该表到激活的 Map 中去。

- 代码

```
Private Sub OpenGeoDataBaseFile(ByVal sAllFileName As String, ByVal sTableName As String)
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pFeatureWorkspace As IFeatureWorkspace
    Dim pFeatureLayer As IFeatureLayer
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim sDir As String

    On Error GoTo ErrorHandler:
        sDir = Dir(sAllFileName)
        If (sDir = "") Then
            MsgBox ("文件不存在")
            Exit Sub
        End If

        'Create a new AccessWorkspaceFactory object and open a GeoDataBaseFile
        Set pWorkspaceFactory = New AccessWorkspaceFactory
        Set pFeatureWorkspace = pWorkspaceFactory.OpenFromFile(sAllFileName, 0)

        'Create a new FeatureLayer and assign a Table to it
        Set pFeatureLayer = New FeatureLayer
        Set pFeatureLayer.FeatureClass = pFeatureWorkspace.OpenFeatureClass(sTableName)
        pFeatureLayer.Name = pFeatureLayer.FeatureClass.AliasName

        'Add the FeatureLayer to the focus map
        Set pMxDocument = Application.Document
        Set pMap = pMxDocument.FocusMap
        pMap.AddLayer pFeatureLayer
        Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControl1_Click()
    Dim pVBProject As VBProject
    On Error GoTo ErrorHandler:
```

```

Set pVBProject = ThisDocument.VBProject
OpenGeoDataBaseFile pVBProject.FileName & "..\..\..\.." & "\data\airport.mdb",
"arterials"
Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

```

1.3.4. 如何连接 Coverage 文件

本例实现的是如何在当前激活的 Map 中连接一个 Coverage 文件。

● 要点

使用 ArcInfoWorkspaceFactory 类实现 IWorkspaceFactory 接口对象，用 IWorkspaceFactory.Open 方法打开一个 Workspace，并获得 Dataset 对象。由于此时的 Dataset 对象可能有多个 Coverage 文件，所以要获得 IEnumDataset 接口对象，通过 IEnumDataset.Next 方法获得一个 Coverage 文件，并将其所有的 FeatureClass 放在 IFeatureClassContainer 对象中。最后通过 IFeatureClassContainer.Class 方法获得 IFeatureClass 接口实例，用 IMap.AddLayer 方法将要连接的 Coverage 文件的所有 FeatureClass 加载到当前激活的 Map 中。

主要用到 IWorkspaceFactory 接口，IWorkspace 接口，IPropertySet 接口，IDataset 接口，IEnumDataset 接口，IFeatureClassContainer 接口。

● 程序说明

函数 ConnectCoverageFile 将 sFilePath 指定的 ArcInfo Workspace 中的名称和 sFileName 相同的 Coverage 文件加载到当前激活的 Map 中。

● 代码

```

Private Sub ConnectCoverageFile(ByVal sFilePath As String, ByVal sFileName As String)
    Dim pWorkspace As IWorkspace
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pPropertySet As IPropertySet
    Dim pDataset As IDataset
    Dim pEnumDataset As IEnumDataset
    Dim pFeatureClassC As IFeatureClassContainer
    Dim pFeatureLayer As IFeatureLayer
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim nNumber As Integer
    Dim sWorkspace As String

    On Error GoTo ErrorHandler:

```

```

sWorkspace = Dir(sFilePath, vbDirectory)
If (sWorkspace = "") Then
    MsgBox ("文件不存在")
    Exit Sub
End If
Set pWorkspaceFactory = New ArcInfoWorkspaceFactory
Set pPropertySet = New PropertySet
'canada is an arcinfo workspace
pPropertySet.SetProperty "DATABASE", sFilePath
'pWorkSp is a pointer to the IArcInfoWorkspace
Set pWorkspace = pWorkspaceFactory.Open(pPropertySet, 0)
'now get to dataset objects using Idataset
Set pDataset = pWorkspace

' use enum to get datasets
Set pEnumDataset = pDataset.Subsets
pEnumDataset.Reset
' use FeatureClassContainer to get datasets
Set pFeatureClassC = pEnumDataset.Next
Do While Not pFeatureClassC Is Nothing
    Set pDataset = pFeatureClassC
    If (pDataset.Name <> sFileName) Then
        Set pFeatureClassC = pEnumDataset.Next
    Else
        Exit Do
    End If
Loop
'add FeatureClassContainer to map
If (pFeatureClassC Is Nothing) Then
    MsgBox ("文件不存在")
Else
    nNumber = 0
    Set pMxdDocument = ThisDocument
    Set pMap = pMxdDocument.FocusMap

    Do While nNumber < pFeatureClassC.ClassCount
        Set pFeatureLayer = New FeatureLayer
        Set pFeatureLayer.FeatureClass = pFeatureClassC.Class(nNumber)
        pFeatureLayer.Name = pFeatureLayer.FeatureClass.AliasName
        nNumber = nNumber + 1
        pMap.AddLayer pFeatureLayer
    Loop
End If
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControll_Click()
    Dim pVBProject As VBProject
On Error GoTo ErrorHandler:
    Set pVBProject = ThisDocument.VBProject
    ConnectCoverageFile pVBProject.FileName & "..\..\..\.." & "\data\canada", "canada"
    Exit Sub
ErrorHandler:

```

```
MsgBox Err.Description
End Sub
```

1.3.5. 如何连接栅格文件

本例实现的是如何在当前激活的 Map 中添加一个栅格文件。

- 要点

创建一个 IRasterLayer 接口对象, 使用 IRasterLayer.CreateFromFilePath 方法加载一个 Raster 文件, 最后用 IMap.AddLayer 方法将 IRasterLayer 添加到当前激活的 Map 中。

主要用到 IRasterLayer 接口。

- 程序说明

函数 AddRasterFile 将路径 sFilePath 下的栅格文件 sFileName 添加到当前激活的 Map 中。

- 代码

```
Private Sub AddRasterFile(sFilePath As String, sFileName As String)
    'sFileName: the filename of the raster dataset
    'sPath: the directory where the raster dataset resides
    Dim pRasterLayer As IRasterLayer
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim sRasterFile As String

    On Error GoTo ErrorHandler:
        sRasterFile = Dir(sFilePath & sFileName)
        If (sRasterFile = "") Then
            MsgBox ("文件不存在")
            Exit Sub
        End If
        'Create a raster layer
        Set pRasterLayer = New RasterLayer
        'This is only one of the three ways to create a RasterLayer object.
        'If there is already a Raster or RasterDataset object, then
        'method CreateFromDataset or CreateFromRaster can be used.
        pRasterLayer.CreateFromFilePath sFilePath & sFileName
        'Add the raster layer to ArcMap
        Set pMxDocument = ThisDocument
        Set pMap = pMxDocument.FocusMap
        pMap.AddLayer pRasterLayer
        pMxDocument.ActiveView.Refresh
        Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControll_Click()
```

```

        Dim pVBProject          As VBProject
On Error GoTo ErrorHandler:
    Set pVBProject = ThisDocument.VBProject
    AddRasterFile pVBProject.FileName & "..\..\..\..\data\" & "photo.tif"
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.6. 如何创建 Shape 文件

本例实现的是如何创建一个 Shape 文件。

● 要点

首先创建新 IField 接口实例，生成新字段，并获得该实例的 IFieldEdit 接口对象，用 FieldsEdit 的 AddField 方法将新字段加入到 IFields 接口对象中，最后用 IFeatureWorkspace 的 CreateFeatureClass 方法生成新的 Shape 文件

主要用到 IFeatureWorkspace 接口，IWorkspaceFactory 接口，IFieldsEdit 接口，IFieldEdit 接口，IFeatureClass 接口。

● 程序说明

函数 CreatShapeFile 根据输入的文件路径和文件名，创建 Shape 文件。

● 代码

```

Private Sub CreatShapeFile(ByVal sFilePath As String, ByVal sFileName As String)
    Dim pFeatureWorkspace          As IFeatureWorkspace
    Dim pWorkspaceFactory          As IWorkspaceFactory
    Dim pFields                    As IFields
    Dim pFieldsEdit                As IFieldsEdit
    Dim pField                    As IField
    Dim pFieldEdit                As IFieldEdit
    Dim pGeometryDef              As IGeometryDef
    Dim pGeometryDefEdit          As IGeometryDefEdit
    Dim pFeatClass                As IFeatureClass
    Dim sShapeFieldName           As String
    Dim sNewShapeFileName         As String

On Error GoTo ErrorHandler:
    sNewShapeFileName = Dir(sFilePath & sFileName & ".shp")
    If (sNewShapeFileName <> "") Then
        MsgBox ("文件已经存在")
        Exit Sub
    End If

    sShapeFieldName = "Shape"

    'Open the folder to contain the shapefile as a workspace
    Set pWorkspaceFactory = New ShapefileWorkspaceFactory
    Set pFeatureWorkspace = pWorkspaceFactory.OpenFromFile(sFilePath, 0)

```

```

' Set up a simple fields collection
Set pFields = New esriCore.Fields
Set pFieldsEdit = pFields

' Make the shape field
' it will need a geometry definition, with a spatial reference
Set pField = New esriCore.Field
Set pFieldEdit = pField
pFieldEdit.Name = sShapeFieldName
pFieldEdit.Type = esriFieldTypeGeometry

Set pGeometryDef = New GeometryDef
Set pGeometryDefEdit = pGeometryDef
With pGeometryDefEdit
    .GeometryType = esriGeometryPolygon
    Set .SpatialReference = New UnknownCoordinateSystem
End With
Set pFieldEdit.GeometryDef = pGeometryDef
pFieldsEdit.AddField pField
' Add others miscellaneous text field
Set pField = New esriCore.Field
Set pFieldEdit = pField
With pFieldEdit
    .Name = "SmallInteger"
    .Type = esriFieldTypeSmallInteger
End With
pFieldsEdit.AddField pField

Set pField = New esriCore.Field
Set pFieldEdit = pField
With pFieldEdit
    .Name = "Integer"
    .Type = esriFieldTypeInteger
End With
pFieldsEdit.AddField pField

Set pField = New esriCore.Field
Set pFieldEdit = pField
With pFieldEdit
    .Name = "Single"
    .Type = esriFieldTypeSingle
End With
pFieldsEdit.AddField pField

Set pField = New esriCore.Field
Set pFieldEdit = pField
With pFieldEdit
    .Precision = 5
    .Scale = 5
    .Name = "Double"
    .Type = esriFieldTypeDouble
End With
pFieldsEdit.AddField pField

```

```

Set pField = New esriCore.Field
Set pFieldEdit = pField
With pFieldEdit
    .Length = 30
    .Name = "String"
    .Type = esriFieldTypeString
End With
pFieldsEdit.AddField pField

Set pField = New esriCore.Field
Set pFieldEdit = pField
With pFieldEdit
    .Name = "Date"
    .Type = esriFieldTypeDate
End With
pFieldsEdit.AddField pField

'Create the shapefile
'(some parameters apply to geodatabase options and can be defaulted as Not ing)
Set pFeatClass = pFeatureWorkspace.CreateFeatureClass _
    (sFileName, pFields, Nothing, Nothing, _
    esriFTSimple, sShapeFieldName, "")

sNewShapeFileName = Dir(sFilePath & "\MyShapeFile.shp")
If (sNewShapeFileName = "") Then
    MsgBox ("Build Success")
Else
    MsgBox ("Build Fail")
End If
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControl1_Click()
    Dim pVBProject As VBProject
On Error GoTo ErrorHandler:
    Set pVBProject = ThisDocument.VBProject

    'Dont include .shp extension
    CreatShapeFile pVBProject.FileName & "\..\..\..\.." & "\data\", "MyShapeFile"
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControl1_Click()
    Dim pVBProject As VBProject
On Error GoTo ErrorHandler:
    Set pVBProject = ThisDocument.VBProject

    'Dont include .shp extension
    CreatShapeFile pVBProject.FileName & "\..\..\..\.." & "\data\", "MyShapeFile"
Exit Sub
ErrorHandler:

```



```
MsgBox Err.Description
End Sub
```

1.3.7. 如何创建 DBF 文件

本例要实现的是如何创建一个单独的 DBF 文件。

- 要点

首先设定 DBF 文件的字段个数，再创建新的 IField 对象，生成新字段，设置其属性，再加入到 IFields 对象中，最后用 IFeatureWorkspace.CreateTable 方法创建一个新的 DBF 文件并返回 ITable 对象。

主要用到 IField 接口，IFieldEdit 接口，IFields 接口，IFieldsEdit 接口。

- 程序说明

函数 CreateDBF 根据输入的路径和文件名创建一个 DBF 文件并返回一个 ITable 对象。

- 代码

```
Private Function CreateDBF (sFilePath As String, sFileName As String) As ITable
'createDBF: simple function to create a DBASE file.
'note: the name of the DBASE file should not contain the .dbf extension

On Error GoTo ErrorHandler:

    Dim pFeatureWorkspace          As IFeatureWorkspace
    Dim pWorkspaceFactory          As IWorkspaceFactory
    Dim FileFolder                 As New Scripting.FileSystemObject
    Dim pFieldsEdit                As esriCore.IFieldsEdit
    Dim pFieldEdit                 As esriCore.IFieldEdit
    Dim pFields                    As IFields
    Dim pField                     As IField
    Dim sDir                       As String

    'Open the Workspace
    Set pWorkspaceFactory = New ShapefileWorkspaceFactory
    If Not FileFolder.FolderExists(sFilePath) Then
        MsgBox "路径不存在" & vbCrLf & sFilePath
        Exit Function
    End If

    sDir = Dir(sFilePath & sFileName & ".dbf")
    If (sDir <> "") Then
        MsgBox ("文件已存在")
        Exit Function
    End If

    Set pFeatureWorkspace = pWorkspaceFactory.OpenFromFile(sFilePath, 0)

    'if a fields collection is not passed in then create one
```

```

'create the fields used by our object
Set pFields = New esriCore.Fields
Set pFieldsEdit = pFields
pFieldsEdit.FieldCount = 6

'Create text Fields
Set pField = New Field
Set pFieldEdit = pField
With pFieldEdit
    .Name = "SmallInteger"
    .Type = esriFieldTypeSmallInteger
End With
Set pFieldsEdit.Field(0) = pField

Set pField = New Field
Set pFieldEdit = pField
With pFieldEdit
    .Name = "Integer"
    .Type = esriFieldTypeInteger
End With
Set pFieldsEdit.Field(1) = pField
Set pField = New Field
Set pFieldEdit = pField
With pFieldEdit
    .Name = "Single"
    .Type = esriFieldTypeSingle
End With
Set pFieldsEdit.Field(2) = pField

Set pField = New Field
Set pFieldEdit = pField
With pFieldEdit
    .Precision = 5
    .Scale = 5
    .Name = "Double"
    .Type = esriFieldTypeDouble
End With
Set pFieldsEdit.Field(3) = pField

Set pField = New Field
Set pFieldEdit = pField
With pFieldEdit
    .Length = 30
    .Name = "String"
    .Type = esriFieldTypeString
End With
Set pFieldsEdit.Field(4) = pField

Set pField = New Field
Set pFieldEdit = pField
With pFieldEdit
    .Name = "Date"
    .Type = esriFieldTypeDate
End With
Set pFieldsEdit.Field(5) = pField

```

```

        Set createDBF = pFeatureWorkspace.CreateTable(sFileName, pFields, Nothing, Nothing,
        "")
        sDir = Dir(sFilePath & sFileName & ".dbf")
        If (sDir <> "") Then
            MsgBox ("Build Success")
        Else
            MsgBox ("Build Fail")
        End If

        Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIButtonControl1_Click()
    Dim pVBProject As VBProject
    Dim pTable As ITable

    On Error GoTo ErrorHandler:
        Set pVBProject = ThisDocument.VBProject

        ' Dont include .dbf extension
        Set pTable = CreateDBF (pVBProject.FileName & "..\..\..\.." & "\data\","MyDI File")
        Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.8. 如何创建 GeoDataBase 文件

本例要实现的是如何创建一个 GeoDataBase 文件。

● 要点

定义 IWorkspaceFactory 接口对象，并用 esriCore.AccessWorkspaceFactory 类来实现，再调用 IWorkspaceFactory.Create 方法创建一个 GeoDataBase 文件。

主要用到了 IWorkspaceFactory 接口。

● 程序说明

函数 CreateAccessWorkspace 根据要创建的 GeoDataBase 文件所在路径 sFilePath 和文件名 sFileName 创建 GeoDataBase 文件。

● 代码

```

Private Function CreateAccessWorkspace(sFilePath As String, sFileName As String)
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim sDir As String

```

```

On Error GoTo ErrorHandler:

    sDir = Dir(sFilePath & sFileName & ".mdb")
    If (sDir <> "") Then
        MsgBox ("文件已存在")
        Exit Function
    End If

    'create the Access Workspace factory
    Set pWorkspaceFactory = New esriCore.AccessWorkspaceFactory
    pWorkspaceFactory.Create sFilePath, sFileName, Nothing, 0

    sDir = Dir(sFilePath & sFileName & ".mdb")
    If (sDir <> "") Then
        MsgBox ("Build Success")
    Else
        MsgBox ("Build Fail")
    End If

    Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIButtonControll1_Click()
    Dim pVBProject As VBProject

    On Error GoTo ErrorHandler:
        Set pVBProject = ThisDocument.VBProject

        'Dont include .mdb extension
        CreateAccessWorkspace pVBProject.FileName & "\..\..\..\.." & " data\","
        "MyGEODataFile"
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.9. 如何创建 Coverage 文件

本例要实现的是如何创建一个 Coverage 文件。

● 要点

首先为 IWorkspaceFactory 接口创建一个 ArcInfoWorkspaceFactory 的实例，然后根据路径 sWorkspacePath 使用 IWorkspaceFactory.Create 方法和 IWorkspaceFactory.Open 方法，获得一个名为 sWorkspaceName 的 ArcInfo Workspace，最后使用 IArcInfoWorkspace.CreateCoverage 方法创建一个名为 sFileName 的 Coverage 文件。

主要用到 IWorkspaceFactory 接口，IArcInfoWorkspace 接口和 IPropertySet 接口。

- 程序说明

函数 CreateCoverageFile 根据路径 sWorkspacePath 和名称 sWorkspaceName 创建一个 ArcInfo Workspace，再在其中创建名为 sFileName 的 Coverage 文件。

- 代码

```
Private Sub CreateCoverageFile(ByVal sWorkspacePath As String, _
    ByVal sWorkspaceName As String, ByVal sFileName As String)

    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pArcInfoWorkspace As IArcInfoWorkspace
    Dim pPropertySet As IPropertySet
    Dim pFeatureDataset As IFeatureDataset
    Dim sTemplateCoverage As String
    Dim sCoverageFile As String

    On Error GoTo ErrorHandler:

    sCoverageFile = Dir(sWorkspacePath & "\" & sWorkspaceName & "\" & sFileName,
vbDirectory)
    If (sCoverageFile <> "") Then
        MsgBox ("文件已经存在")
        Exit Sub
    End If

    Set pFeatureDataset = Nothing
    Set pPropertySet = New PropertySet
    pPropertySet.SetProperty "SERVER", sWorkspaceName

    Set pWorkspaceFactory = New ArcInfoWorkspaceFactory

    'create an arcinfoworkspace
    pWorkspaceFactory.Create sWorkspacePath, sWorkspaceName, pPropertySet, 0

    pPropertySet.SetProperty "DATABASE", sWorkspacePath & "\" & sWorkspaceName

    'pArcInfoWorkspace is a pointer to the IArcInfoWorkspace
    Set pArcInfoWorkspace = pWorkspaceFactory.Open(pPropertySet, 0)

    'create a coverage without a template
    Set pFeatureDataset = pArcInfoWorkspace.CreateCoverage(sFileName, "", _
        esriCoveragePrecisionDouble)

    ' or use the methods on iarinfoworkspace
    sTemplateCoverage = "C:\arcgis\arcexe83\arcobjects\developer
kit\samples\data\canada\canada"
    Set pFeatureDataset = pArcInfoWorkspace.CreateCoverage(sF leName,
sTemplateCoverage, _
        esriCoveragePrecisionDouble)
```

```

    If (pFeatureDataset Is Nothing) Then
        MsgBox ("Build Success")
    Else
        MsgBox ("Build Fail")
    End If
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControll1_Click()
    Dim pVBProject As VBProject
On Error GoTo ErrorHandler:
    Set pVBProject = ThisDocument.VBProject

    CreateCoverageFile pVBProject.FileName & "..\..\..\.." & "\data", _
        "MyArcInfoWorkspace", "MyCoverFile"

    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.10. 如何建立文件连接 (Join / Link)

本例实现的是如何将地图中的一个 FeatureLayer 的属性表与另一个数据文件建立连接。

● 要点

首先需要定义两个 ITable 接口对象，分别用来获得地图中的属性表和需要连接的数据文件，再通过 IMemoryRelationshipClassFactory.Open 方法将两个 ITable 接口对象根据某个关键字段建立连接，

最后使用 IDisplayRelationshipClass.DisplayRelationshipClass 方法将显示该连接

主要用到 IMemoryRelationshipClassFactory 接口，IRelationshipClass 接口和 IDisplayRelationshipClass 接口。

● 程序说明

函数 Join 是将当前激活的地图中名称为 sLayerName 的图层和路径为 sFilePath、文件名为 sFileName 的文件按字段名为 sFieldName 的字段进行连接。

● 代码

```

Private Function Join(ByVal sLayerName As String, ByVal sFilePath As String, _
    ByVal sFileName As String, ByVal sFieldName As String) As Boolean

    Dim pMxDocument As IMxDocument

```

```

Dim pMap As IMap
Dim pWorkspaceFactory As IWorkspaceFactory
Dim pWorkspace As IWorkspace
Dim pFeatureWorkspace As IFeatureWorkspace
Dim pFeatureLayer As IFeatureLayer
Dim pFeatureClass As IFeatureClass
Dim pPrimaryTable As ITable
Dim pForeignTable As ITable
Dim pDisplayTable As IDisplayTable
Dim pMemoryRelationshipCF As IMemoryRelationshipClassFactory
Dim pRelationshipClass As IRelationshipClass
Dim pDisplayRelationshipC As IDisplayRelationshipClass
Dim nNumber As Integer
Dim sForeignFile As String

On Error GoTo ErrorHandler:
Join = False

sForeignFile = Dir(sFilePath & "\" & sFileName)
If (sForeignFile = "") Then
MsgBox "The ForeignFile is not exist."
Exit Function
End If

Set pWorkspaceFactory = New ShapefileWorkspaceFactory
Set pWorkspace = pWorkspaceFactory.OpenFromFile(sFilePath, 0)
Set pFeatureWorkspace = pWorkspace
Set pForeignTable = pFeatureWorkspace.OpenTable(sFileName)

Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
For nNumber = 0 To pMap.LayerCount - 1
If pMap.Layer(nNumber).Name = sLayerName Then
Set pFeatureLayer = pMap.Layer(nNumber)
Exit For
End If
Next
If pFeatureLayer Is Nothing Then
MsgBox "No Layer's Name is " & sLayerName
Exit Function
End If
Set pDisplayTable = pFeatureLayer
Set pFeatureClass = pDisplayTable.DisplayTable
Set pPrimaryTable = pFeatureClass
Set pMemoryRelationshipCF = New MemoryRelationshipClassFactory
Set pRelationshipClass = pMemoryRelationshipCF.Open("TabletoLayer", pPrimaryTable,
sFieldName, _
pForeignTable, sFieldName, "forward", "backward",
esriRelCardinalityOneToOne)
Set pDisplayRelationshipC = pFeatureLayer
pDisplayRelationshipC.DisplayRelationshipClass pRelationshipClass,
esriLeftOuterJoin

Join = True
Exit Function

```



```

Set pQueryFilter = New QueryFilter
pQueryFilter.WhereClause = "FID < 2"
Set pFeatureCursor = pFeatureClass.Search(pQueryFilter, False)
Set pFeature = pFeatureCursor.NextFeature
Do While Not pFeature Is Nothing
    'More Operations
    Set pFeature = pFeatureCursor.NextFeature
Loop

Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

Private Sub UIButtonControl1_Click()
On Error GoTo ErrorHandler:
SelectFeatures
Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

```

1.3.12. 如何编辑记录

本例实现的是如何修改 FeatureClass 中某条记录（Feature）的值。

● 要点

通过 IFeatureClass.Update 方法获得可修改记录的 IFeatureCursor 接口对象，使用 IFeatureCursor.NextFeature 方法获得 IFeature 接口对象，修改其属性值，通过 IFeatureCursor.UpdateFeature 方法提交 IFeature 修改内容。

主要用到 IFeatureCursor 接口

● 程序说明

函数 OpenFeatureClass 获得当前激活的 Map 中第一层的 IFeatureClass 接口对象。

函数 EditFeature 修改 pFeatureClass 中第一条记录的第七个字段的值。

● 代码

```

Private Function EditFeature(pFeatureClass As IFeatureClass) As Boolean
Dim pFeature As IFeature
Dim pFeatureCursor As IFeatureCursor

On Error GoTo ErrorHandler:
EditFeature = False
If (pFeatureClass Is Nothing) Then
Exit Function
End If

```

```

Set pFeatureCursor = pFeatureClass.Update(Nothing, False)
Set pFeature = pFeatureCursor.NextFeature

If (Not pFeature Is Nothing) Then
    pFeature.Value(6) = "New Place"
    pFeatureCursor.UpdateFeature pFeature
    MsgBox ("修改成功")
    EditFeature = True
Else
    MsgBox ("修改失败")
End If

Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Function OpenFeatureClass() As IFeatureClass
    Dim pMxDocument          As IMxDocument
    Dim pMap                  As IMap
    Dim pFeatureLayer         As IFeatureLayer
    Dim pFeatureClass         As IFeatureClass

On Error GoTo ErrorHandler:

    Set OpenFeatureClass = Nothing
    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
    If (pMap.LayerCount = 0) Then
        MsgBox ("缺少数据")
        Exit Function
    End If

    Set pFeatureLayer = pMap.Layer(0)
    Set pFeatureClass = pFeatureLayer.FeatureClass
    Set OpenFeatureClass = pFeatureClass
    Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIButtonControll1_Click()
On Error GoTo ErrorHandler:
    Dim pFeatureClass          As IFeatureClass
    Set pFeatureClass = OpenFeatureClass()
    EditFeature pFeatureClass
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1. 3. 13. 如何增加记录

本例要实现的是如何在 FeatureClass 中新增一条记录 (Feature)。

- 要点

通过 IFeatureClass.Insert 方法获得可插入记录的游标 IFeatureCursor, 然后使用 IFeatureClass.CreateFeatureBuff 方法获得 IFeatureBuffer 接口实例, 使用 IFeatureCursor.InsertFeature 方法插入记录。

主要用到 IFeatureCursor 接口。

- 程序说明

函数 OpenFeatureClass 获得当前激活的 Map 中第一层的 IFeatureClass 接口对象。

函数 InsertFeature 在 pFeatureClass 中添加一条记录。

- 代码

```
Private Function InsertFeature(pFeatureClass As IFeatureClass) As Boolean
    Dim pFeatureCursor As IFeatureCursor
    Dim pFeatureBuffer As IFeatureBuffer
    Dim nFeatureNumber As Integer

    On Error GoTo ErrorHandler:
        InsertFeature = False
        If (pFeatureClass Is Nothing) Then
            Exit Function
        End If

        Set pFeatureCursor = pFeatureClass.Insert(True)
        Set pFeatureBuffer = pFeatureClass.CreateFeatureBuffer

        nFeatureNumber = -1
        pFeatureBuffer.Value(6) = "Insert Land"
        nFeatureNumber = pFeatureCursor.InsertFeature(pFeatureBuffer)

        If (nFeatureNumber <> -1) Then
            MsgBox ("添加了第" & nFeatureNumber & "条记录")
            InsertFeature = True
        Else
            MsgBox ("添加失败")
            InsertFeature = False
        End If

        Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Function OpenFeatureClass() As IFeatureClass
    Dim pMxdDocument As IMxdDocument
    Dim pMap As IMap
    Dim pFeatureLayer As IFeatureLayer
    Dim pFeatureClass As IFeatureClass
```

```

On Error GoTo ErrorHandler:
    Set OpenFeatureClass = Nothing

    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
    If (pMap.LayerCount = 0) Then
        MsgBox ("缺少数据")
        Exit Function
    End If
    Set pFeatureLayer = pMap.Layer(0)
    Set pFeatureClass = pFeatureLayer.FeatureClass
    Set OpenFeatureClass = pFeatureClass
    Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIButtonControll1_Click()
On Error GoTo ErrorHandler:
    Dim pFeatureClass As IFeatureClass
    Set pFeatureClass = OpenFeatureClass()
    InsertFeature pFeatureClass
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.14. 如何删除记录

本例要实现的是如何在 FeatureClass 中删除一条记录 (Feature)。

● 要点

获得游标 IFeatureCursor，然后定义 IFeature 接口对象，并获得要删除的记录，最后使用 IFeature.Delete 方法删除记录。

主要用到 IFeature 接口和 IFeatureCursor 接口。

● 程序说明

函数 OpenFeatureClass 获得当前激活的 Map 中第一层的 IFeatureClass 接口对象。

函数 DeleteFeature 删除 PLACENAME 字段值为 "Insert Land" 的所有记录。

● 代码

```

Private Sub DeleteFeature(pFeatureClass As IFeatureClass)
    Dim pFeature As IFeature
    Dim pFeatureCursor As IFeatureCursor
    Dim pQueryFilter As IQueryFilter
    Dim nFeatureNumber As Integer

On Error GoTo ErrorHandler:

```

```

    If (pFeatureClass Is Nothing) Then
        Exit Sub
    End If

    Set pQueryFilter = New QueryFilter
    pQueryFilter.WhereClause = "PLACENAME = 'Insert Land'"
    Set pFeatureCursor = pFeatureClass.Search(pQueryFilter, False)
    Set pFeature = pFeatureCursor.NextFeature
    nFeatureNumber = 0

    Do While Not pFeature Is Nothing
        pFeature.Delete
        nFeatureNumber = nFeatureNumber + 1
        Set pFeature = pFeatureCursor.NextFeature
    Loop
    MsgBox ("Delete " & nFeatureNumber & " Features")

    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Function OpenFeatureClass() As IFeatureClass
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pFeatureLayer As IFeatureLayer
    Dim pFeatureClass As IFeatureClass

    On Error GoTo ErrorHandler:
        Set OpenFeatureClass = Nothing

        Set pMxDocument = ThisDocument
        Set pMap = pMxDocument.FocusMap
        If (pMap.LayerCount = 0) Then
            MsgBox ("缺少数据")
            Exit Function
        End If

        Set pFeatureLayer = pMap.Layer(0)
        Set pFeatureClass = pFeatureLayer.FeatureClass
        Set OpenFeatureClass = pFeatureClass
        Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIButtonControl1_Click()
    On Error GoTo ErrorHandler:
        Dim pFeatureClass As IFeatureClass
        Set pFeatureClass = OpenFeatureClass()
        DeleteFeature pFeatureClass
        Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.15. 如何纪录排序(ITableSort)

本例要实现的是如何将一个FeatureClass中的数据按某字段的值进行排序。

- 要点

定义 ITableSort 接口对象，并用 TableSort 类实现之，设置排序所用到的字段、排序方式（升序或降序）以及排序的数据源，然后使用 ITableSort.Sort 方法进行排序。

主要用到 ITableSort 接口。

- 程序说明

函数 OpenFeatureClass 获得当前激活的 Map 中第一层的 IFeatureClass 接口对象。

函数 SortFeatures 按照 pFeatureClass 的第五个字段值对 pFeatureClass 的数据进行从小到大排序，并返回一个排好序的 ICursor 接口对象。

- 代码

```
Private Function SortFeatures(pFeatureClass As IFeatureClass) As ICursor
    Dim pTableSort As ITableSort
    Dim pFields As IFields
    Dim pField As IField
    Dim pQueryFilter As IQueryFilter
    Dim pCursor As ICursor

    On Error GoTo ErrorHandler:
        Set SortFeatures = Nothing
        Set pFields = pFeatureClass.Fields
        Set pField = pFields.Field(5)
        Set pTableSort = New esriCore.TableSort
        Set pQueryFilter = New QueryFilter
        Set pCursor = Nothing

        With pTableSort
            .Fields = pField.Name
            .Ascending(pField.Name) = True
            .CaseSensitive(pField.Name) = True
            Set .QueryFilter = pQueryFilter
            Set .Table = pFeatureClass
        End With

        pTableSort.Sort Nothing
        Set pCursor = pTableSort.Rows
        Set SortFeatures = pCursor
        If (pCursor Is Nothing) Then
            MsgBox ("未排序")
        Else
            MsgBox ("排序完成")
        End If
    End Function
```

```

        End If
        Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Function OpenFeatureClass() As IFeatureClass
    Dim pMxDocument          As IMxDocument
    Dim pMap                  As IMap
    Dim pFeatureLayer         As IFeatureLayer
    Dim pFeatureClass         As IFeatureClass

On Error GoTo ErrorHandler:
    Set OpenFeatureClass = Nothing

    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
    If (pMap.LayerCount = 0) Then
        MsgBox ("缺少数据")
        Exit Function
    End If

    Set pFeatureLayer = pMap.Layer(0)
    Set pFeatureClass = pFeatureLayer.FeatureClass
    Set OpenFeatureClass = pFeatureClass
    Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIButtonControll1_Click()
On Error GoTo ErrorHandler:
    Dim pFeatureClass          As IFeatureClass
    Set pFeatureClass = OpenFeatureClass()
    SortFeatures pFeatureClass
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControll1_Click()
On Error GoTo ErrorHandler:
    Dim pFeatureClass          As IFeatureClass
    Set pFeatureClass = OpenFeatureClass()
    SortFeatures pFeatureClass
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.16. 如何添加字段

本例实现的是如何在一个 FeatureClass 中新增一个字段(Field)。

- 要点

定义 IField 接口对象, 并用 Field 类实现, 通过 IFieldEdit 接口对象设置 IField 接口对象的属性, 最后通过 IFeatureClass.AddField 方法添加一个字段。

主要用到 IField 接口、IFieldEdit 接口和 IFeatureClass 接口。

- 程序说明

函数 OpenFeatureClass 获得当前激活的 Map 中第一层的 IFeatureClass 接口对象。

函数 AddField 生成一个新的字段(Field)并添加到 pFeatureClass 中。

- 代码

```
Private Function AddField(pFeatureClass As IFeatureClass) As Boolean
    Dim pField As IField
    Dim pFieldEdit As IFieldEdit

    On Error GoTo ErrorHandler:
        AddField = False
        If (pFeatureClass Is Nothing) Then
            Exit Function
        End If

        Set pField = New esriCore.Field
        Set pFieldEdit = pField
        With pFieldEdit
            .Length = 10
            .Name = "NewField"
            .Type = esriFieldTypeString
        End With

        pFeatureClass.AddField pField
        MsgBox ("已添加新字段:" & " " & pField.Name)
        AddField = True
        Exit Function
    ErrorHandler:
        MsgBox Err.Description
End Function

Private Function OpenFeatureClass() As IFeatureClass
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pFeatureLayer As IFeatureLayer
    Dim pFeatureClass As IFeatureClass

    On Error GoTo ErrorHandler:
        Set OpenFeatureClass = Nothing

        Set pMxDocument = ThisDocument
        Set pMap = pMxDocument.FocusMap
        If (pMap.LayerCount = 0) Then
            MsgBox ("缺少数据")
```



```

        Exit Function
    End If

    Set pFeatureLayer = pMap.Layer(0)
    Set pFeatureClass = pFeatureLayer.FeatureClass
    Set OpenFeatureClass = pFeatureClass
    Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIButtonControll1_Click()
On Error GoTo ErrorHandler:
    Dim pFeatureClass As IFeatureClass
    Set pFeatureClass = OpenFeatureClass()
    AddField pFeatureClass
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.17. 如何删除字段

本例实现的是如何在一个 FeatureClass 中删除一个字段(Field)。

● 要点

定义 IField 接口实例，并使用 Field 类实现，使用 IFields.FindField 方法和 IFields.Field 方法获得 IFeatureClass 中要删除的字段，最后用 IFeatureClass.DeleteField 方法删除字段。

主要用到 IFields 接口，IField 接口和 IFeatureClass 接口。

● 程序说明

函数 OpenFeatureClass 获得当前激活的 Map 中第一层的 IFeatureClass 接口对象。

函数 DeleteField 删除 pFeatureClass 中字段名为 NewField 的字段。

● 代码

```

Private Function DeleteField(pFeatureClass As IFeatureClass) As Boolean
    Dim pFields As IFields
    Dim pField As IField
    Dim lFieldNumber As Long

On Error GoTo ErrorHandler:
    DeleteField = False
    If (pFeatureClass Is Nothing) Then
        Exit Function
    End If

```

```

Set pFields = pFeatureClass.Fields
lFieldNumber = pFields.FindField("NewField")
If (lFieldNumber = -1) Then
    MsgBox ("无此字段")
    Exit Function
End If

Set pField = pFields.Field(lFieldNumber)
pFeatureClass.DeleteField pField
MsgBox ("已删除字段:" & "NewField")
DeleteField = True

Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Function OpenFeatureClass() As IFeatureClass
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pFeatureLayer As IFeatureLayer
    Dim pFeatureClass As IFeatureClass

On Error GoTo ErrorHandler:
    Set OpenFeatureClass = Nothing

    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
    If (pMap.LayerCount = 0) Then
        MsgBox ("缺少数据")
        Exit Function
    End If

    Set pFeatureLayer = pMap.Layer(0)
    Set pFeatureClass = pFeatureLayer.FeatureClass
    Set OpenFeatureClass = pFeatureClass
    Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIButtonControl1_Click()
On Error GoTo ErrorHandler:
    Dim pFeatureClass As IFeatureClass
    Set pFeatureClass = OpenFeatureClass()
    DeleteField pFeatureClass
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1. 3. 18. 如何进行空间查询

本例实现的是在一个图层上画一个 polygon，根据该 polygon 查询出图层上

与之相交的 polygon 并高亮显示出来。

- 要点

通过 RubberPolygon 类来实现接口 IRubberBand 接口对象，用 IRubberBand.TrackNew 方法在图层上画出 polygon, 然后定义 IGeometry 获得该 polygon, 创建 ISpatialFilter 接口对象实现过滤功能, 通过 ILayer 接口实例获得 IFeatureSelection 接口，调用。

IFeatureSelection.SelectFeatures 方法将结果高亮显示。

- 程序说明

过程 UIToolControl1_MouseDown 是实现模块。

- 代码

```
Option Explicit

Private Function UIToolControl1_Deactivate() As Boolean
    UIToolControl1_Deactivate = True
End Function

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, _
    ByVal x As Long, ByVal y As Long)

    Dim pMxDoc As IMxDocument
    Dim pActiveView As IActiveView
    Dim pScreenDisplay As IScreenDisplay
    Dim pRubberPolygon As IRubberBand
    Dim pFillSymbol As ISimpleFillSymbol
    Dim pRgbColor As IRgbColor
    Dim pPolygon As IPolygon
    Dim pGeometry As IGeometry
    Dim pFeatselect As IFeatureSelection
    Dim pSpatialFilter As ISpatialFilter

On Error GoTo ErrorHandler:
    Set pMxDoc = ThisDocument
    Set pActiveView = pMxDoc.FocusMap
    'Draw Polygon
    Set pScreenDisplay = pActiveView.ScreenDisplay
    Set pRubberPolygon = New RubberPolygon

    Set pFillSymbol = New SimpleFillSymbol
    Set pRgbColor = New RgbColor
    pRgbColor.NullColor = True
    pFillSymbol.Color = pRgbColor

    Set pPolygon = pRubberPolygon.TrackNew(pScreenDisplay, pFillSymbol)

    With pScreenDisplay
        .StartDrawing pScreenDisplay.hDC, esriNoScreenCache
        .SetSymbol pFillSymbol
        .DrawPolygon pPolygon
    End With
End Sub
```

```

        .FinishDrawing
    End With
    'set up pFilter
    Set pGeometry = pPolygon
    Set pSpatialFilter = New SpatialFilter

    With pSpatialFilter
        Set .Geometry = pGeometry
        .SpatialRel = esriSpatialRelIntersects
    End With
    'select
    Set pFeatselect = pMxDoc.FocusMap.Layer(0)
    pFeatselect.SelectFeatures pSpatialFilter, esriSelectionResultNew, False
    pFeatselect.SelectionSet.Refresh
    pMxDoc.ActiveView.Refresh
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.19. 如何进行高级空间查询(两个层之间的空间查询)

本例实现的是在 Map 的两个 Polygon 图层中，查询出第一个 Polygon 层中的 Polygon 被第二个 Polygon 层的 Polygon 包含的所有记录

● 要点

定义 IGeometryCollection 接口实例，并使用 GeometryBag 类实现，将查询图层所有记录的图形信息添加进去。创建 ISpatialFilter 接口实例来设置空间查询运算符，本例设为 esriSpatialRelContains。通过查询层 Featurelayer 获得 IFeatureSelection 接口实例，最后使用 IFeatureSelection.SelectFeatures 方法实现本例。

● 程序说明

本例使用的数据为 “WorldCountries.shp” 和 “USUrbanAreas.shp”。

过程 UIButtonControl1_Click 是实现模块。

● 代码

```

Option Explicit
Private Sub UIButtonControl1_Click()
    Dim pMxDoc As IMxDocument
    Dim pMap As IMap
    Dim pQueryFeatLayer As IFeatureLayer
    Dim pFeatLayer As IFeatureLayer
    Dim pFeatureClass As IFeatureClass
    Dim pInFeatureCursor As IFeatureCursor
    Dim pOutFeatureCursor As IFeatureCursor
    Dim pFeature As IFeature

```

```

Dim pFeatselect As IFeatureSelection
Dim pFilter As ISpatialFilter
Dim pGeoCollection As IGeometryCollection

On Error GoTo Err_Handle:
    Set pMxDoc = ThisDocument
    Set pMap = pMxDoc.FocusMap

    ' according to the name of layers to set up featurelayer
    If pMap.Layer(1).Name = "WorldCountries" Then
        Set pFeatLayer = pMap.Layer(1)
        Set pQueryFeatLayer = pMap.Layer(0)
    Else
        Set pFeatLayer = pMap.Layer(0)
        Set pQueryFeatLayer = pMap.Layer(1)
    End If

    Set pFeatureClass = pFeatLayer.FeatureClass
    Set pGeoCollection = New esriCore.GeometryBag
    Set pOutFeatureCursor = pFeatureClass.Search(Nothing, False)
    Set pFeature = pOutFeatureCursor.NextFeature

    ' add feature into pGeoCollection
    Do While Not pFeature Is Nothing
        pGeoCollection.AddGeometry pFeature.Shape
        Set pFeature = pOutFeatureCursor.NextFeature
    Loop

    Set pFilter = New SpatialFilter
    ' set up pFilter
    With pFilter
        Set .Geometry = pGeoCollection
        .GeometryField = "Shape"
        .SpatialRel = esriSpatialRelContains
    End With

    Set pFeatselect = pQueryFeatLayer

    ' filter the features and display the results in screen
    pFeatselect.SelectFeatures pFilter, esriSelectionResultNew, False
    pFeatselect.SelectionSet.Refresh
    pMxDoc.ActiveView.Refresh
    Exit Sub
Err_Handle:
    MsgBox Err.Description
End Sub

```

1. 3. 20. 如何进行层与层之间的逻辑运算

本例要实现的是将两个同一 GeometryType 图层联合成为一个图层，输出 Shape 文件，并且加载到 Map 中显示出来。

● 要点

定义 ITable 的两个接口变量，通过两个图层 FeatureClass 实例化。然后由接口 IFeatureClassName、IWorkspaceName 和 IDatasetName 实现创建一个新的 shape 文件。再创建 IBasicGeoprocessor 接口对象，使用 IBasicGeoprocessor.Union 方法实现两个图层的联合。

- 程序说明

过程 UIButtonControll_Click 是实现模块。

- 代码

```
Option Explicit
Private Sub UIButtonControll_Click()
    Dim pMxDoc As IMxDocument
    Dim pLayer As ILayer
    Dim pInputTable As ITable
    Dim pOverlayTable As ITable
    Dim pFeatClassName As IFeatureClassName
    Dim pNewWSName As IWorkspaceName
    Dim pDatasetName As IDatasetName
    Dim dtol As Double
    Dim pBasicGeop As IBasicGeoprocessor
    Dim pOutputFeatClass As IFeatureClass
    Dim pOutputFeatLayer As IFeatureLayer
    Dim App As VBProject

    On Error GoTo ErrorHandler:
        Set pMxDoc = ThisDocument
        Set pLayer = pMxDoc.FocusMap.Layer(0)
        Set App = ThisDocument.VBProject

        ' Get the input table
        ' Use the Itable interface from the Layer (not from the FeatureClass)
        Set pInputTable = pLayer

        ' Get the overlay layer and table
        ' Use the Itable interface from the Layer (not from the FeatureClass)
        Set pLayer = pMxDoc.FocusMap.Layer(1)
        Set pOverlayTable = pLayer

        ' Error checking
        If pInputTable Is Nothing Then
            MsgBox "Table QI failed"
            Exit Sub
        End If

        If pOverlayTable Is Nothing Then
            MsgBox "Table QI failed"
            Exit Sub
        End If

        ' Define the output feature class name
        Set pFeatClassName = New FeatureClassName
```

```

' Set output location and feature class name
Set pNewWSName = New WorkspaceName
pNewWSName.WorkspaceFactoryProgID = "esriCore.ShapeFileWorkspaceFactory.1"
pNewWSName.PathName = App.FileName & "\\.."

Set pDatasetName = pFeatClassName
pDatasetName.Name = "Union_result"

Set pDatasetName.WorkspaceName = pNewWSName

' Set the tolerance. Passing 0.0 causes the default tolerance to be used.
' The default tolerance is 1/10,000 of the extent of the data frame's spatial domain
dtol = 0#

' Perform the union
Set pBasicGeop = New BasicGeoprocessor

Set pOutputFeatClass = pBasicGeop.Union(pInputTable, False, pOverlayTable, False, _
dtol, pFeatClassName)

' Add the output layer to the map
Set pOutputFeatLayer = New FeatureLayer
Set pOutputFeatLayer.FeatureClass = pOutputFeatClass
pOutputFeatLayer.Name = pOutputFeatClass.AliasName
pMxDoc.FocusMap.AddLayer pOutputFeatLayer
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1.3.21. 如何将 shape 文件转化成 GeoDataBase(各种文件格式的转换)

本例演示的是如何将 shape 文件转化成 personal GeoDatabase 文件，其它格式间的与此转换类似。主要用到 IFeatureDataConverter 接口的 ConvertFeatureClass 方法。

● 要点

首先，创建新的 GeoDataBase 数据库，并创建 IFeatureDatasetName 对象。创建定义两个 IFeatureClassName 接口对象分别引用输入表（shape 文件）和输出表。

然后设置输出表的 Shape 字段的 GeometryDef 属性。这一步非常关键，因为其中包含了数据库和 shape 文件的空间参考信息。

最后调用 IFeatureDataConverter.ConvertFeatureClass 方法完成功能。

● 程序说明

过程 UIBConvert_Click 是实现模块, 调用过程 ConvertShapeToGeodatabase 实现功能。

sDataPath 定义了数据与工程文件的相对路径。SHAPE_NAME 描述了要转化的 shape 文件的文件名。MDB_NAME 和 F_DS_NAME 分别描述了 Access 数据库名和库的数据集的名称。

● 代码

```
Option Explicit

Private Sub UIBConvert_Click()
    Call ConvertShapeToGeodatabase
End Sub

Private Sub ConvertShapeToGeodatabase()
    Dim pOutWorkspaceFactory As IWorkspaceFactory
    Dim pOutWorkspaceName As IWorkspaceName
    Dim pInWorkspaceName As IWorkspaceName
    Dim pOutFeatureDSName As IFeatureDatasetName
    Dim pOutDSName As IDatasetName
    Dim pInFeatureClassName As IFeatureClassName
    Dim pInDatasetName As IDatasetName
    Dim pOutFeatureClassName As IFeatureClassName
    Dim pOutDatasetName As IDatasetName
    Dim iCounter As Long
    Dim pOutFields As IFields
    Dim pInFields As IFields
    Dim pFieldChecker As IFieldChecker
    Dim pGeoField As IField
    Dim pOutGeometryDef As IGeometryDef
    Dim pOutGeometryDefEdit As IGeometryDefEdit
    Dim pName As IName
    Dim pInFeatureClass As IFeatureClass
    Dim pShpToFeatClsConverter As IFeatureDataConverter
    Dim pVBProject As VBProject

    Dim sDataPath As String
    Const SHAPE_NAME As String = "country"
    Const MDB_NAME As String = "countryDB"
    Const F_DS_NAME As String = "World"

    On Error GoTo ErrorHandler

    Set pVBProject = ThisDocument.VBProject
    sDataPath = pVBProject.FileName & "\..\..\..\data\"

    If Not "" = Dir(sDataPath & MDB_NAME & ".mdb") Then
        MsgBox MDB_NAME & ".mdb already exist"
        Exit Sub
    Else
        ' Create a new Access database
        Set pOutWorkspaceFactory = New AccessWorkspaceFactory
    End If
End Sub
```



```

Set pOutWorkspaceName = pOutWorkspaceFactory.Create(sDataPath, MI3_NAME,
Nothing, 0)
' create a new feature dataset name object for the output Access feature dataset,
call
' it "World"
Set pOutFeatureDSName = New FeatureDatasetName
Set pOutDSName = pOutFeatureDSName
Set pOutDSName.WorkspaceName = pOutWorkspaceName
pOutDSName.Name = F_DS_NAME

' Get the name object for the input shapefile workspace
Set pInWorkspaceName = New WorkspaceName
pInWorkspaceName.PathName = sDataPath
pInWorkspaceName.WorkspaceFactoryProgID = _
"esriCore.ShapefileWorkspaceFactory.1"

Set pInFeatureClassName = New FeatureClassName
Set pInDatasetName = pInFeatureClassName
pInDatasetName.Name = SHAPE_NAME
Set pInDatasetName.WorkspaceName = pInWorkspaceName

' Create the new output FeatureClass name object that will be passed
' into the conversion function
Set pOutFeatureClassName = New FeatureClassName
Set pOutDatasetName = pOutFeatureClassName

' Set the new FeatureClass name to be the same as the input FeatureClass name
pOutDatasetName.Name = pInDatasetName.Name

' Open the input Shapefile FeatureClass object, so that we can get its fields
Set pName = pInFeatureClassName
Set pInFeatureClass = pName.Open

' Get the fields for the input feature class and run them through
' field checker to make sure there are no illegal or duplicate field names
Set pInFields = pInFeatureClass.Fields
Set pFieldChecker = New FieldChecker
pFieldChecker.Validate pInFields, Nothing, pOutFields

' Loop through the output fields to find the geometry field
For iCounter = 0 To pOutFields.FieldCount
    If pOutFields.Field(iCounter).Type = esriFieldTypeGeometry Then
        Set pGeoField = pOutFields.Field(iCounter)
        Exit For
    End If
Next iCounter

' Get the geometry field's geometry definition
Set pOutGeometryDef = pGeoField.GeometryDef

' Give the geometry definition a spatial index grid count and grid size
Set pOutGeometryDefEdit = pOutGeometryDef
pOutGeometryDefEdit.GridCount = 1
pOutGeometryDefEdit.GridSize(0) = 1500000

```

```

' Now use IFeatureDataConverter::Convert to create the output FeatureDat set and
' FeatureClass.
Set pShpToFeatClsConverter = New FeatureDataConverter
pShpToFeatClsConverter.ConvertFeatureClass pInFeatureClassName, Nothing, _
    pOutFeatureDSName, pOutFeatureClassName, Nothing, pOutFields, " ", 1000,
0

    MsgBox "Convert operation complete!", vbInformation
End If

Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.22. 如何将 Map 中显示的图形转化成栅格文件

本例要实现的是如何将当前激活的 Map 中显示的图形转化成栅格文件。

● 要点

通过 IMap 实例获得 IActiveView 接口对象，定义 IExporter 接口变量，使用 TiffExporter 实现该接口并对其中的属性进行赋值，使用 IActiveView.Output 方法将 Map 中显示的图形导出。

主要用到 IActiveView 接口，IExporter 接口和 IEnvelope 接口。

● 程序说明

函数 Output 将当前激活的 Map 中显示的图形转化成栅格文件，栅格文件路径及名称由参数 sFileAllName 确定。

● 代码

```

Private Sub Output(ByVal sFileAllName As String)
    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pExporter As IExporter
    Dim pEnvelope As IEnvelope
    Dim ptagRECT As tagRECT
    Dim pTrackCancel As ITrackCancel
    Dim lscreenResolution As Long

    On Error GoTo ErrorHandler:
        Set pMxDocument = ThisDocument
        Set pActiveView = pMxDocument.ActiveView

        lscreenResolution = pActiveView.ScreenDisplay.DisplayTransformation.Resolution

        ptagRECT.Top = 0
        ptagRECT.Left = 0
        ptagRECT.Right = pActiveView.Extent.Width

```

```

ptagRECT.bottom = pActiveView.Extent.Height

' We must calculate the size of the user specified Rectangle in Device unit
' Hence convert width and height
Set pEnvelope = New Envelope
pEnvelope.PutCoords ptagRECT.Left, ptagRECT.bottom, ptagRECT.Right, ptagRECT.Top
Set pExporter = New TiffExporter
pExporter.Resolution = lscreenResolution
pExporter.ExportFileName = sFileAllName
pExporter.PixelBounds = pEnvelope

Set pTrackCancel = New CancelTracker

pActiveView.Output pExporter.StartExporting, lscreenResolution, _
    ptagRECT, pActiveView.Extent, pTrackCancel

pExporter.FinishExporting
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControl1_Click()
    Dim pVBProject As VBProject
On Error GoTo ErrorHandler:
    Set pVBProject = ThisDocument.VBProject
    Output pVBProject.FileName & "..\..\..\.." & "\data\MyTifFile.tif"
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.23. 如何打开选中的层或独立表的属性窗口

本例实现的是如何打开选中的层或独立表的属性窗口 (Attribute Table)。主要用到 ITableWindow 和 ITableWindow2 接口。

● 要点

首先需要选中一个层或独立表。可在 UI Button Control 的 Enabled 事件中测试用户选定了有效的对象后，才使按钮有效。

然后判断属性窗口是否已经打开。如果尚未打开，则创建新的 ITableView2 对象。

● 程序说明

过程 UIBAttributeWindow_Click 调用过程 OpenAttribWnd 实现功能。

函数 UIBAttributeWindow_Enabled 用来测试用户是否已正确选中了层或独立表，如果是，则使按钮有效。

过程 OpenAttribWnd 是功能模块，实现了属性窗口的测试和创建，以及显示。

● 代码

```
Option Explicit

Private Sub UIBAttributeWindow_Click()
    Call OpenAttribWnd
End Sub

Private Function UIBAttributeWindow_Enabled() As Boolean
    Dim pMxDocument As IMxDocument
    Dim pSelectedItem As IUnknown
    Dim bEnabled As Boolean
    Set pMxDocument = ThisDocument
    Set pSelectedItem = pMxDocument.SelectedItem
    bEnabled = True

    ' Disable if the selected item is nothing or if
    ' it is not a layer or table
    If pSelectedItem Is Nothing Then
        bEnabled = False
    ElseIf (TypeOf pSelectedItem Is IFeatureLayer) Or (TypeOf pSelectedItem Is
    IStandaloneTable) Then
        bEnabled = True
    End If

    UIBAttributeWindow_Enabled = bEnabled
End Function

Private Sub OpenAttribWnd()
    Dim pMxDocument As IMxDocument
    Dim pLayer As ILayer
    Dim pStandaloneTable As IStandaloneTable
    Dim pSelectedItem As IUnknown
    Dim pTableWindowExist As ITableWindow
    Dim pTableWindow2 As ITableWindow2
    Dim bSetProperties As Boolean

    On Error GoTo ErrorHandler:

    Set pMxDocument = ThisDocument
    Set pSelectedItem = pMxDocument.SelectedItem
    Set pTableWindow2 = New TableWindow
    ' Determine the selected item's type
    ' Exit sub if item is not a feature layer or standalone table
    If TypeOf pSelectedItem Is IFeatureLayer Then
        Set pLayer = pSelectedItem
        Set pTableWindowExist = pTableWindow2.FindViaLayer(pLayer)
        ' Check if a table already exist; if not create one
        If pTableWindowExist Is Nothing Then
            Set pTableWindow2.Layer = pLayer
            bSetProperties = True
        End If
    ElseIf TypeOf pSelectedItem Is IStandaloneTable Then
```

```

        Set pStandaloneTable = pSelectedItem
        Set pTableWindowExist = pTableWindow2.FindViaStandaloneTable(pStandaloneTable)
        ' Check if a table already exists; if not, create one
        If pTableWindowExist Is Nothing Then
            Set pTableWindow2.StandaloneTable = pStandaloneTable
            bSetProperties = True
        End If
    End If

    If bSetProperties Then
        pTableWindow2.TableSelectionAction = esriSelectFeatures
        pTableWindow2.ShowSelected = False
        pTableWindow2.ShowAliasNamesInColumnHeadings = True
        Set pTableWindow2.Application = Application
    Else
        Set pTableWindow2 = pTableWindowExist
    End If

    ' Ensure Table Is Visible
    If Not pTableWindow2.IsVisible Then
        pTableWindow2.Show True
    End If

    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.3.24. 如何拷贝属性表中的一行

本例要实现的是如何将所有属性表（Attribute Table）中的行拷贝到 Windows 剪贴板，使用户能使用文本编辑器等软件对选中的数据做进一步编辑，从而满足特殊要求。行中的每个属性用半角字符的逗号“,”分隔，行间用换行符分隔。

● 要点

首先需要取得某属性表中的所有选中记录的全部属性，以一个字符串来存储。因为在属性表中选取中记录（Row）后，层中的相应记录（Feature）也将选中。两种途径都能获得所需属性值。

得到所需的字符串 sResult 后，就可以将其拷贝到剪贴板。在 VB 中剪贴板是全局对象。可像如下使用：

```

Clipboard.Clear
Clipboard.SetText sResult

```

本例将在 VBA 中实现相同的功能。用到了 IGraphicsContainer、IGraphicsContainerSelect、ITextElement、IElement、IClipboardFormat 接

口。

- 程序说明

过程 UIBCopyRow_Click 是实现模块，调用过程 CopyRow 实现功能。过程 CopyRow 将选中行的全部属性值（忽略 Shape 属性）连接成字符串，然后创建 TextElement 对象，并添加到 IGraphicsContainer 对象的选择集中，再调用 TextClipboardFormat 的 Copy 方法，把字符拷贝到 Windows 剪贴板。

- 代码

```
Option Explicit

Private Sub UIBCopyRow_Click()
    Call CopyRow
End Sub

Public Sub CopyRow()
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pActiveView As IActiveView
    Dim pGraphicsContainer As IGraphicsContainer
    Dim pGraphicsContainerS As IGraphicsContainerSelect
    Dim pFields As IFields
    Dim iCounter As Integer
    Dim iIndex As Integer
    Dim pTextElement As ITextElement
    Dim pElement As IElement
    Dim sResult As String
    Dim pEnumFeature As IEnumFeature
    Dim pEnumFeatureS As IEnumFeatureSetup
    Dim pFeature As IFeature
    Dim pClipboardFormat As IClipboardFormat

    On Error GoTo ErrorHandler
    ' Used for string operation on the clipboard
    Set pClipboardFormat = New TextClipboardFormat

    Set pMxDocument = ThisDocument
    Set pActiveView = pMxDocument.ActivatedView
    Set pMap = pMxDocument.FocusMap
    Set pGraphicsContainer = pMap

    ' Get selected features to retrieve their attribute values
    Set pEnumFeature = pMap.FeatureSelection
    Set pEnumFeatureS = pEnumFeature
    pEnumFeatureS.AllFields = True
    Set pFeature = pEnumFeature.Next
    If pFeature Is Nothing Then
        MsgBox "No row selected"
        Exit Sub
    End If
```

```

Set pFields = pFeature.Fields
iCounter = pFields.FieldCount

Do Until pFeature Is Nothing
    For iIndex = 0 To iCounter - 1
        If Not TypeOf pFeature.Value(iIndex) Is IGeometry Then
            sResult = sResult & pFeature.Value(iIndex) & ","
        End If
    Next iIndex
    ' Remove the trailing comma
    sResult = Left(sResult, Len(sResult) - 1)
    sResult = sResult & vbNewLine
    Set pFeature = pEnumFeature.Next
Loop

' If you're tending to build a dll to implement the same function and
' programming in VB enviroment, simply use the next to statement
' to copy the string into windows clipboard
' Clipboard.Clear
' Clipboard.SetText sResult
' Otherwise, programe as follows
' Copy the string into clipboard using objects included in esriCore

' To clear clipboard
pClipboardFormat.Paste pMxDocument
pGraphicsContainer.DeleteAllElements

' Construct a new TextElement with the string to copy into clipboard
Set pTextElement = New TextElement
pTextElement.Text = sResult
Set pElement = pTextElement
' Point(100, 100) is for temporary use
pElement.Geometry = pActiveView.ScreenDisplay.DisplayTransformation _
    .ToMapPoint(100, 100)
Set pGraphicsContainer = pMap
pGraphicsContainer.AddElement pElement, 0
Set pGraphicsContainerS = pGraphicsContainer
pGraphicsContainerS.UnselectAllElements
pGraphicsContainerS.SelectElement pElement

pClipboardFormat.Copy pMxDocument

pGraphicsContainerS.UnselectElement pElement
pGraphicsContainer.DeleteElement pElement
pActiveView.Refresh

Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

```

1. 3. 25. 如何为当前层或独立表创建一个 Summary 表

本例要实现的是如何按某一字段“分组”(dissolve), 统计其它字段的数据

信息摘要（创建 Summary 表）。可得到的主要信息包括该字段值相同的每组记录中的记录数量、最大值、最小值、和、平均值等。主要用到 IBasicGeoprocessor 接口的 Dissolve 方法。

- 要点

为当前层创建 Summary 表，要得到当前层的引用，并确定在其上执行 Dissolve 操作的字段。对独立表的操作方法与层的操作类似。

- 程序说明

过程 UIBCreateSummaryTable_Click 是实现模块，调用过程 CreateSummaryTable 实现功能。过程 CreateSummaryTable 中应先确认层（例中为 states）和要“Dissolve”的字段（例中为 SUB_REGION）存在，同时要定义摘要表的名字（本例为 SumStates）。

然后指定执行 Dissolve 方法的操作符（如 Minimum, Count, Average 等）和在其上施行操作的字段名（例中为 AREA）。操作结果作为独立表添加到当前 Map。

因为 Dissolve 方法参数表中的“输入表”和“输出数据集的名字”都是引用，为了避免多次调用过程使最终 SumStates 表中的结果不唯一，每次执行 Dissolve 前，将 SumStates 的已存内容删除。

- 代码

```
Private Sub UIBCreateSummaryTable_Click()  
    Call CreateSummaryTable  
End Sub  
  
Public Sub CreateSummaryTable()  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pLayer As ILayer  
    Dim pFeatLayer As IFeatureLayer  
    Dim iCount As Integer  
  
    Dim pFeatureClass As IFeatureClass  
    Dim pTable As ITable  
    Dim pDataSet As IDataset  
    Dim pWorkspace As IWorkspace  
    Dim pWorkspaceDataset As IDataset  
    Dim pWorkspaceName As IName  
  
    Dim pOutTableName As ITableName  
    Dim pOutDatasetName As IDatasetName  
    Dim pEnumDataset As IEnumDataset  
    Dim pBasicGeoprocessor As IBasicGeoprocessor  
    Dim pSumTable As ITable  
  
    Dim pStandaloneTable As IStandaloneTable
```



```

Dim pStandaloneTableColl As IStandaloneTableCollection

' Define current layer name and output table name
Const sLayerName As String = "states"
Const sSumTableName As String = "SumStates"

Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap

On Error GoTo ErrorHandler

Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap

On Error GoTo ErrorHandler

' Find the layer named states
For iCount = 0 To pMap.LayerCount - 1
    Set pLayer = pMap.Layer(iCount)
    If TypeOf pLayer Is IFeatureLayer Then
        If pLayer.Name = sLayerName Then
            Set pFeatLayer = pLayer
            Exit For
        End If
    End If
Next

If pFeatLayer Is Nothing Then
    MsgBox "The " & sLayerName & " layer was not found"
    Exit Sub
End If

' Get the workspace of the states layer
Set pFeatureClass = pFeatLayer.FeatureClass
Set pTable = pFeatureClass
Set pDataSet = pTable
Set pWorkspace = pDataSet.Workspace
Set pWorkspaceDataset = pWorkspace
Set pWorkspaceName = pWorkspaceDataset.FullName

' Set up the output table
Set pOutTableName = New TableName
Set pOutDatasetName = pOutTableName
pOutDatasetName.Name = sSumTableName
Set pOutDatasetName.WorkspaceName = pWorkspaceName

' Make sure there is a field called SUB_REGION in the layer
If pTable.FindField("SUB_REGION") = -1 Then
    MsgBox "There must be a field named SUB_REGION in states"
    Exit Sub
End If

' Check if SumStates.dbf file already exist: if yes, delete it
Set pEnumDataset = pWorkspace.Datasets(esriDTable)
Set pWorkspaceDataset = pEnumDataset.Next

```

```

Do Until pWorkspaceDataset Is Nothing
    If pWorkspaceDataset.Name = pOutDatasetName.Name Then
        pWorkspaceDataset.Delete
        Exit Do
    End If
    Set pWorkspaceDataset = pEnumDataset.Next
Loop

' Perform the summarize. Note the summary fields string (minimum.SUB_REGION ...)
' below. This is a comma-delimited string that lists the generated summary
' fields. Each field must start with a keyword, and be followed by .fieldN me,
' where fieldName is the name of a field in the original table.
'
' If you specify the Shape field, you must use the keyword 'Dissolve'. Thi
' is not used below since we are creating a non-spatial summary table.

Set pBasicGeoprocessor = New BasicGeoprocessor
Set pSumTable = pBasicGeoprocessor.Dissolve(pTable, False, "SUB_REGION", _
    "Minimum.SUB_REGION, Count.SUB_REGION, Sum.AREA, Average.AREA," & _
    "Minimum.AREA, Maximum.AREA, StdDev.AREA, Variance.AREA", _
    pOutDatasetName)

' add the table to map
Set pStandaloneTable = New StandaloneTable
Set pStandaloneTable.Table = pSumTable
Set pStandaloneTableColl = pMap
pStandaloneTableColl.AddStandaloneTable pStandaloneTable

' Refresh the TOC
pMxDocument.UpdateContents

Exit Sub
ErrorHandler:
    MsgBox Err.Number & " " & Err.Description
End Sub

```

1.3.26. 如何利用用户定义的规则创建定制的排序

利用 ITableSort 接口可以完成普通的对记录排序的功能。ITableSortCallback 机制允许用户通过执行自定义的排序算法来完成定制的排序。本例演示了如何创建这样的用户类，通过实现 ITableSortCallback 接口来完成该功能。

假设有如下原始数据：其中“Address”字段描述了道路（Street）的道路编号（Street Number）如“2805”，和道路名（Stree Name）如“Citrus Ave”。

Attributes of TableSort				
	OBJECTID*	Address	Town	State
	1	2805 Citrus Ave	Redlands	California
	2	301 Atwood Street	College	Alaska
	3	1089 Iowa Street	Bellingham	Washington
	4	4500 University Ave	Williston	North Dakota
	5	1609 Cherry Hill Road	Inkster	Michigan
	6	587 Tyler Street	Quincy	Massachusetts

Record: Show: Records 0 out of 6 Selected

现在要按道路名排序所有的记录。因为排序字段时必须忽略道路编号，故需要自定义排序规则。

- 要点

首先需要创建用户自定义的类，并生成其实例。该类实现了 ITableSortCallBack 接口。然后把它的引用赋给 ITableSort 的 Compare 属性。最后用 ITableSort 的 Sort 方法完成排序。

- 程序说明

过程 UIBCustomSort_Click 是实现模块，调用过程 CustomSort 实现功能。

类模块 clsTailSort 为自定义模块，实现 ITableSortCallBack 接口。包括两个函数：ITableSortCallBack_Compare（用于定义字符串比较的规则）和 Get_String（用于得到地址字段的道路名部分）。

过程 CustomSort 中创建 TableSort 和 clsTailSort 的实例，并对结果进行排序，然后调用过程 CreateTable，将排序后的结果存入 C:\temp 目录的 NewSortTable.dbf 文件，并作为独立表加入当前 Map。

- 代码

类模块 clsTailSort

```
Option Explicit
' Custom class for ITableSortCallBack
' ClassName: clsTailSort

Implements ITableSortCallBack

Private Function ITableSortCallBack_Compare(ByVal value1 As Variant, ByVal value2 As Variant, ByVal FieldIndex As Long, ByVal fieldSortIndex As Long) As Long
    ' Custom table sort
    ' Get_string function gets the first block of characters (e.g street numbers)
```

```

' in each value.
' Comparison is then made on the remaining characters (e.g. street names).
On Error GoTo ErrorHandler

value1 = Get_String(value1)
value2 = Get_String(value2)

If value1 > value2 Then
    ITableSortCallBack_Compare = 1
ElseIf value1 < value2 Then
    ITableSortCallBack_Compare = -1
Else: value1 = value2
    ITableSortCallBack_Compare = 0
End If

Exit Function
ErrorHandler:
    MsgBox Err.Description

End Function

Private Function Get_String(ByVal sMyStr As Variant) As Variant
' This function gets the tail of the string
' after the first block of characters
Dim sFindString As String
Dim nPosition As Integer
Dim nStringLen As Integer

On Error GoTo ErrorHandler
nStringLen = Len(sMyStr)
nPosition = 1
Do Until nPosition = nStringLen
    sFindString = Mid(sMyStr, nPosition, 1)
    If sFindString = " " Then
        Exit Do
    End If
    nPosition = nPosition + 1
Loop
Get_String = Mid(sMyStr, nPosition + 1)
Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

```

功能模块

```

Option Explicit

Private pMxDocument As IMxDocument
Private pMap As IMap
Private pApplication As IApplication
Public Sub CustomSort()

```

```

Dim pSelectedItem As IUnknown
Dim pStandaloneTable As IStandaloneTable
Dim pTable As ITable
Dim pTableSort As ITableSort
Dim pTableSortCallBack As ITableSortCallBack
Dim pCursor As ICursor
Dim pRow As IRow

On Error GoTo ErrorHandler

Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pApplication = Application
Set pSelectedItem = pMxDocument.SelectedItem

If pSelectedItem Is Nothing Then
    MsgBox "Nothing selectd.", vbExclamation
    Exit Sub
' If a table is selected
ElseIf Not TypeOf pSelectedItem Is IStandaloneTable Then
    MsgBox "No table selectd.", vbExclamation
    Exit Sub
Else
    Set pStandaloneTable = New esriCore.StandaloneTable
    Set pStandaloneTable = pSelectedItem
End If

Set pTable = pStandaloneTable.Table

' Create a new custom TableSortCallBack and TableSort object
' Class clsTailSort defined in Class Modules
Set pTableSortCallBack = New clsTailSort
Set pTableSort = New TableSort

' Set up the parameters for the sort and excute
With pTableSort
    .Fields = "Address"
    .Ascending("Address") = True
    .CaseSensitive("Address") = True
    Set .Table = pTable
    Set .Compare = pTableSortCallBack
End With
pTableSort.Sort Nothing

' Create a new cursor object to hold the sorted rows
Set pCursor = pTableSort.Rows

' Create a new sorted table
Call CreateTable(pTable, pCursor)

Set pTableSortCallBack = Nothing
Set pTableSort = Nothing

Exit Sub
ErrorHandler:

```

```

MsgBox Err.Description
End Sub

Public Sub CreateTable(pTab As ITable, pCur As ICursor)
    ' Create a new .dbf file of the sorted data
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pFeatureWorkspace As IFeatureWorkspace
    Dim pWorkspace As IWorkspace
    Dim pDatasetWkSp As IDataset
    Dim pWorkspaceName As IWorkspaceName
    Dim pDatasetNameOut As IDatasetName
    Dim pFields As IFields
    Dim pFields2 As esriCore.IFields
    Dim pDataset As IDataset
    Dim pDatasetName As IDatasetName
    Dim pDS As IDataset
    Dim pEnumDS As IEnumDataset

    Dim pStandaloneTable2 As IStandaloneTable
    Dim pTable2 As ITable
    Dim pTableNew As ITable
    Dim pCursor2 As ICursor
    Dim pRowBuffer As IRowBuffer
    Dim pRow As IRow
    Dim pName As IName
    Dim pStandaloneTable As IStandaloneTable
    Dim pStandaloneTableC As IStandaloneTableCollection

    Dim j As Integer
    Dim i As Integer

    On Error GoTo ErrorHandler

    ' Get the dataset name for the input table
    Set pDataset = pTab
    Set pDatasetName = pDataset.FullName

    ' Set the output dataset name.
    ' New .dbf file will be created in c:\temp
    Set pFields = pTab.Fields
    Set pWorkspaceFactory = New ShapefileWorkspaceFactory
    Set pWorkspace = pWorkspaceFactory.OpenFromFile("c:\temp", 0)
    Set pFeatureWorkspace = pWorkspace
    Set pDatasetWkSp = pWorkspace
    Set pWorkspaceName = pDatasetWkSp.FullName
    Set pDatasetNameOut = New TableName
    pDatasetNameOut.Name = "NewSortTable"
    Set pDatasetNameOut.WorkspaceName = pWorkspaceName

    ' Check if .dbf file already exist: if yes, delete it
    Set pEnumDS = pWorkspace.Datasets(esriDTable)
    Set pDS = pEnumDS.Next
    Do Until pDS Is Nothing
        If pDS.Name = pDatasetNameOut.Name Then
            pDS.Delete
        End If
    Loop

```

```

        Exit Do
    End If
    Set pDS = pEnumDS.Next
Loop

' Create a new .dbf table
pFeatureWorkspace.CreateTable pDatasetNameOut.Name, pFields, Nothing, Nothing, ""

' Create a new stand alone table object to represent the .dbf table
Set pStandaloneTable2 = New StandaloneTable
Set pStandaloneTable2.Table = pFeatureWorkspace.OpenTable(pDatasetNameOut.Name)
Set pTable2 = pStandaloneTable2.Table
Set pFields2 = pTable2.Fields

' Open an insert cursor on the new table
Set pCursor2 = pTable2.Insert(True)

' Create a row buffer for the row inserts
Set pRowBuffer = pTable2.CreateRowBuffer

' Loop through the sorted cursor and write to new table
For j = 0 To pTab.RowCount(Nothing) - 1
    Set pRow = pCur.NextRow
    If Not pRow Is Nothing Then
        i = 1
        Do Until i = pFields2.FieldCount
            If Not IsEmpty(pRow.Value(i)) Then
                If pFields.Field(i).Editable Then
                    pRowBuffer.Value(i) = pRow.Value(i)
                End If
            End If
            i = i + 1
        Loop
        pCursor2.InsertRow pRowBuffer
    End If
Next j

' Add the new sorted table to map document
Set pName = pDatasetNameOut
Set pTableNew = pName.Open
Set pStandaloneTable = New StandaloneTable
Set pStandaloneTable.Table = pTableNew
Set pStandaloneTableC = pMap
pStandaloneTableC.AddStandaloneTable pStandaloneTable

pMxDocument.UpdateContents

Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

```

1.3.27. 如何实现在 ArcMap 上进行属性查询(Identify)

本例要演示的是如何查询 Feature 的属性信息。实现后的结果为选择了 UI Tool Control 后，在要查询的 Feature 上单击鼠标，查询的结果将显示在弹出的窗体上。

● 要点

首先需要得到要查询的 Feature 对象。使用 IIdentify 接口的 Identify 方法可以对给定的位置进行查询，得到结果为 IIdentifyObj 对象的数组。然后通过 IIdentifyObj 对象设置 IFeatureIdentifyObj 查询接口，即可进一步得到 Feature 对象。因为 IFeatureIdentifyObj 接口的 Feature 属性具有只写(write only) 属性，故又用到另一个接口 IRowIdentifyObj。

得到 Feature 对象后即可操作其 Fields 属性和 Value 属性，得到其属性字段名和值。

● 程序说明

在窗体上使用了 MSFlexGrid Control 6.0 来显示查询结果。所以本例也演示了 MSFlexGrid 控件的使用方法。

窗体名: frmResult

MSFlexGrid 控件名: flxAttr

标签控件名: lblLocation (标签用来显示查询位置的地理坐标)

● 代码

```
Private Sub UIT_Identify_MouseDown(ByVal button As Long, ByVal shift As Long,
                                   ByVal x As Long, ByVal y As Long)
    Dim pMxApplication As IMxApplication
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pPoint As IPoint
    Dim pIDArray As IArray
    Dim pIdentify As IIdentify
    Dim pFeatureIdentifyObj As IFeatureIdentifyObj
    Dim pIdentifyObj As IIdentifyObj
    Dim pRowIdentifyObj As IRowIdentifyObject
    Dim pFeature As IFeature
    Dim pFields As IFields
    Dim pField As IField
    Dim iFieldIndex As Integer
    Dim iLayerIndex As Integer
    Dim sShape As String
    On Error GoTo ErrorHandler
```



```

Set pMxApplication = Application
Set pMxDocument = Application.Document
Set pMap = pMxDocument.FocusMap

'Identify from TOP layer to BOTTOM, exit loop since one Feature identified
For iLayerIndex = 0 To pMap.LayerCount - 1
    Set pIdentify = pMap.Layer(iLayerIndex)

    'Convert x and y to map units
    Set pPoint = pMxApplication.Display.DisplayTransformation.ToMapPoint(x y)

    'Set label on the form, coordinates would have 6 digits behind decimal point
    frmResult.lblLocation = "Location:(" & Format(pPoint.x, "##0.000000") & ", " _
        & Format(pPoint.y, "##0.000000") & ")"
    Set pIDArray = pIdentify.Identify(pPoint)

    'Get the FeatureIdentifyObject
    If Not pIDArray Is Nothing Then
        Set pFeatureIdentifyObj = pIDArray.Element(0)
        Set pIdentifyObj = pFeatureIdentifyObj
        pIdentifyObj.Flash pMxApplication.Display

        'Feature property of FeatureIdentifyObject has write only access
        Set pRowIdentifyObj = pFeatureIdentifyObj
        Set pFeature = pRowIdentifyObj.Row
        Set pFields = pFeature.Fields

        'Set the MSFlexGrid control on form to display identify result
        With frmResult.flxAttr
            .AllowUserResizing = flexResizeColumns
            .ColAlignment(1) = AlignmentSettings.flexAlignLeftCenter
            .ColWidth(0) = 1500
            .ColWidth(1) = 1800
            'Add header to MSFlexGrid control
            .Rows = pFields.FieldCount + 1
            .Cols = 2
            .FixedRows = 1
            .FixedCols = 0
            .TextMatrix(0, 0) = "Field"
            .TextMatrix(0, 1) = "Value"

            For iFieldIndex = 0 To pFields.FieldCount - 1
                Set pField = pFields.Field(iFieldIndex)

                'Set field "Field" of the MSFlex control
                .TextMatrix(iFieldIndex + 1, 0) = pField.Name

                'Set field "Value" of the MSFlex control
                Select Case pField.Type
                    Case esriFieldTypeOID
                        .TextMatrix(iFieldIndex + 1, 1) = pFeature.OID
                    Case esriFieldTypeGeometry
                        'The function QueryShapeType return a String that
                        ' correspond with the esriGeoemtryType const

```

```

        sShape = QueryShapeType(pField.GeometryDef.GeometryType)
        .TextMatrix(iFieldIndex + 1, 1) = sShape
    Case Else
        .TextMatrix(iFieldIndex + 1, 1) = pFeature.Value(iFieldIndex)
    End Select
Next iFieldIndex
End With

    frmResult.Show modal
Exit Sub
End If
Next iLayerIndex

' If code goes here, no Feature was identified, clear the MSFlex control's content
' and show a message
frmResult.flxAttr.Clear
MsgBox "No feature identified."

Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Public Function QueryShapeType(ByVal enuGeometryType As esriGeometryType) As String
    Dim sShapeType As String

    Select Case enuGeometryType
        Case esriGeometryPolyline
            sShapeType = "Polyline"
        Case esriGeometryPolygon
            sShapeType = "Polygon"
        Case esriGeometryPoint
            sShapeType = "Point"
        Case esriGeometryMultipoint
            sShapeType = "Multipoint"
        Case esriGeometryNull
            sShapeType = "Unknown"
        Case esriGeometryLine
            sShapeType = "Line"
        Case esriGeometryCircularArc
            sShapeType = "CircularArc"
        Case esriGeometryEllipticArc
            sShapeType = "EllipticArc"
        Case esriGeometryBezier3Curve
            sShapeType = "BezierCurve"
        Case esriGeometryPath
            sShapeType = "Path"
        Case esriGeometryRing
            sShapeType = "Ring"
        Case esriGeometryEnvelope
            sShapeType = "Envelope"
        Case esriGeometryAny
            sShapeType = "Any valid geometry"
        Case esriGeometryBag
            sShapeType = "GeometryBag"
    End Select
End Function

```

```

        Case esriGeometryMultiPatch
            sShapeType = "MultiPatch"
        Case esriGeometryTriangleStrip
            sShapeType = "TriangleStrip"
        Case esriGeometryTriangleFan
            sShapeType = "TriangleFan"
        Case esriGeometryRay
            sShapeType = "Ray"
        Case esriGeometrySphere
            sShapeType = "Sphere"
        Case Else
            sShapeType = "Unknown!"
    End Select
    QueryShapeType = sShapeType
End Function

```

1.3.28. 如何设置和修改层的数据源

本例要实现的是如何改变（或设置）一个层的数据源（Data Source）。主要用到 IMapAdmin2 接口。

● 要点

首先需要得到新数据源的 IFeatureClass 接口对象和当前要改变数据源的层的当前 IFeatureClass 接口对象，然后调用 IMapAdmin2 接口的 FireChangeFeatureClass 方法实现之。

● 程序说明

过程 UICMD_ChageDataSource_Click 是实现模块，调用过程 ChangeLayerDataSource 实现功能。

sNewFileName 是层的新数据源的 shape 文件的完整文件名（包含）。

● 代码

```

Private Sub UICMD_ChageDataSource_Click()
    Dim pVBProject As VBProject
    Dim sProjectName As String
    Dim sNewFileName As String

    On Error GoTo ErrorHandler:

    Set pVBProject = ThisDocument.VBProject
    'Get MXD File Path
    sProjectName = pVBProject.FileName
    'Get Data File Path
    sNewFileName = sProjectName & "..\..\..\data\country.shp"

    'Call Procedure
    ChangeLayerDataSource sNewFileName
    Exit Sub
ErrorHandler:

```

```

        MsgBox Err.Description
    End Sub

Private Sub ChangeLayerDataSource(ByVal sNewFileName As String)
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pWorkspace As IWorkspace
    Dim pFeatureWorkspace As IFeatureWorkspace
    Dim pNewFeatureCls As IFeatureClass
    Dim pOldFeatureCls As IFeatureClass

    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pActiveView As IActiveView
    Dim pMapAdmin2 As IMapAdmin2
    Dim pFeatureLayer As IFeatureLayer

    On Error GoTo ErrorHandler
    'Get Data FeatureClass
    Set pWorkspaceFactory = New ShapefileWorkspaceFactory
    Set pWorkspace = pWorkspaceFactory.OpenFromFile(sNewFileName & "\..\", 0)
    Set pFeatureWorkspace = pWorkspace
    Set pNewFeatureCls = pFeatureWorkspace.OpenFeatureClass("country")
    'Get Lay(0)' s FeatureClass
    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
    Set pMapAdmin2 = pMap
    Set pActiveView = pMap
    Set pFeatureLayer = pMap.Layer(0)
    Set pOldFeatureCls = pFeatureLayer.FeatureClass

    'Change Data Source
    Set pFeatureLayer.FeatureClass = pNewFeatureCls
    pMapAdmin2.FireChangeFeatureClass pOldFeatureCls, pNewFeatureCls
    pActiveView.Refresh

    'if want to change Display in Toc ,cancel these comment below
    'pFeatureLayer.Name = pNewFeatureCls.AliasName
    'pMxDocument.CurrentContentsView.Refresh 0 Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.4. Display

1.4.1. 如何实现在 ArcMap 中放大缩小地图

用户点击按钮后，可以在地图上进行点击或者拖放矩形框来放大缩小地图

● 要点

因为考虑到用户可以单击放大缩小，也可以拖放矩形框来放大缩小，所以不可以直接使用 IRubberBand 接口，而是采用 INewEnvelopeFeedback 接口

●程序说明

主要通过 InewEnvelopeFeedback.StartPoint 和 MoveTo 方法来绘制矩形框，然后赋值给 IActiveView.Extend 属性，达到地图的放大缩小

●代码

```
Private m_pFeedbackEnv As INewEnvelopeFeedback
Private m_pPoint As IPoint
Private m_bIsMouseDown As Boolean
Private m_pActiveView As IActiveView

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)
    Dim pMxDocument As IMxDocument
    On Error GoTo ErrorHandler:
    'Left Button Check
    If button <> 1 Then Exit Sub
    If m_pActiveView Is Nothing Then
        Set pMxDocument = ThisDocument
        Set m_pActiveView = pMxDocument.ActivatedView
    End If
    '得到起始点
    Set m_pPoint = m_pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoin(x, y)
    m_bIsMouseDown = True
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControl1_MouseMove(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)
    On Error GoTo ErrorHandler:
    If Not m_bIsMouseDown Then Exit Sub
    If m_pFeedbackEnv Is Nothing Then
        Set m_pFeedbackEnv = New NewEnvelopeFeedback
        Set m_pFeedbackEnv.Display = m_pActiveView.ScreenDisplay
        m_pFeedbackEnv.Start m_pPoint
    End If
    Set m_pPoint = m_pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoin(x, y)
    'Draw Envelope
    m_pFeedbackEnv.MoveTo m_pPoint
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControl1_MouseUp(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)
    Dim pEnv As IEnvelope
    On Error GoTo ErrorHandler:
    'Left Button Check
```

```

If button <> 1 Then Exit Sub
If (m_pFeedbackEnv Is Nothing) Then
    'User Only Click Map with left button
    Set pEnv = m_pActiveView.Extent
    '如果是缩小的话，将这里的两个 0.5 都改成 1.5
    pEnv.Expand 0.5, 0.5, True
Else
    'User Draw a Envelope
    Set pEnv = m_pFeedbackEnv.Stop
End If
m_pActiveView.Extent = pEnv
m_bIsMouseDown = False
Set m_pPoint = Nothing
Set m_pFeedbackEnv = Nothing
m_pActiveView.Refresh
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.4.2. 如何实现在 ArcMap 中移动地图

用户点击按钮后，可以拖动地图显示

● 要点

采用 IActiveView.ScreenDisplay.PanStart 和 PanStop 方法使地图移动。

● 程序说明

通过 IActiveView.ScreenDisplay 的 PanStart 和 PanStop 方法在 ITool 的MouseDown, MouseUp 和 MouseMove 事件的响应实现移动效果，将移动结果得到 IEnvelope 赋值给 IActiveView.Extent，实现地图的刷新

● 代码

```

Option Explicit
Private m_pMxApp As IMxApplication
Private m_pMxDocument As IMxDocument
Private m_pScreenDisplay As IScreenDisplay
Private m_pMapInsetWindow As IMapInsetWindow
Private m_bMouseDown As Boolean

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, _
    ByVal x As Long, ByVal y As Long)

    Dim pStartPoint As IPoint

    If Not button = 1 Then Exit Sub
    Set m_pScreenDisplay = GetFocusDisplay
    Set m_pMapInsetWindow = GetMapInset(m_pScreenDisplay)
    If Not m_pMapInsetWindow Is Nothing Then
        If m_pMapInsetWindow.IsLive Then Exit Sub
    End If

```

```

        m_bMouseDown = True
        Set pStartPoint = m_pScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
        '得到起始点, 开始移动
        m_pScreenDisplay.PanStart pStartPoint
    End Sub

Private Sub UIToolControl1_MouseMove(ByVal button As Long, ByVal shift As Long, _
    ByVal x As Long, ByVal y As Long)

    Dim pMoveToPoint As IPoint
    If Not m_bMouseDown Then Exit Sub
    Set pMoveToPoint = m_pScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
    '根据鼠标移动, 移动地图
    m_pScreenDisplay.PanMoveTo pMoveToPoint
End Sub

Private Sub UIToolControl1_MouseUp(ByVal button As Long, ByVal shift As Long,
    ByVal x As Long, ByVal y As Long)

    Dim pEnvelope As IEnvelope
    Dim pActiveView As IActiveView
    Dim pMapInset As IMapInset
    Dim pMapInsetWindow As IMapInsetWindow
    If Not m_bMouseDown Then Exit Sub
    m_bMouseDown = False
    Set pEnvelope = m_pScreenDisplay.PanStop
    If pEnvelope Is Nothing Then Exit Sub
    '窗口判断
    If Not m_pMapInsetWindow Is Nothing Then
        Set pMapInset = m_pMapInsetWindow.MapInset
        pMapInset.VisibleBounds = pEnvelope
        m_pMapInsetWindow.Refresh
        Exit Sub
    Else
        Set pActiveView = m_pMxDocument.ActiveView
        '地图刷新
        If TypeOf pActiveView Is IMap Then
            pActiveView.Extent = pEnvelope
            pActiveView.Refresh
        Else
            Set pActiveView = pActiveView.FocusMap
            pActiveView.Extent = pEnvelope
            pActiveView.Refresh
        End If
    End If
End Sub

Private Sub UIToolControl1_Select()
    '初始化接口
    m_bMouseDown = False
    Set m_pMxApp = Application
    Set m_pMxDocument = Application.Document
End Sub

Private Function GetFocusDisplay() As IScreenDisplay
    Dim pActiveView As IActiveView
    Dim pActiveMap As IMap

```

```

Set pActiveView = m_pMxDocument.ActiveView
If TypeOf pActiveView Is IMap Then
    Set GetFocusDisplay = m_pMxApp.Display.FocusScreen
Else
    Set pActiveView = pActiveView.FocusMap
    Set GetFocusDisplay = pActiveView.ScreenDisplay
End If
End Function

Private Function GetMapInset(pScreenDisplay As IScreenDisplay) As IMapInsetWindow
    Dim pAppWindows As IApplicationWindows
    Dim pWindowsSet As ISet
    Dim pDataWindow As IDataWindow
    Dim pLensWindow As ILensWindow
    Set pAppWindows = m_pMxApp 'QI
    Set pWindowsSet = pAppWindows.DataWindows
    pWindowsSet.Reset
    Set pDataWindow = pWindowsSet.Next
    Do While Not pDataWindow Is Nothing
        If TypeOf pDataWindow Is ILensWindow Then
            Set pLensWindow = pDataWindow
            If pLensWindow.ScreenDisplay Is m_pScreenDisplay Then
                If TypeOf pLensWindow Is IMapInsetWindow Then
                    Set GetMapInset = pLensWindow
                    Exit Function
                End If
            End If
        End If
        Set pDataWindow = pWindowsSet.Next
    Loop
    Set GetMapInset = Nothing
End Function

```

1.4.3. 如何实现在 ArcMap 上画 Polygon

用户点击按钮后，在地图上任意点击生成 Polygon，双击 Polygon 生成完成

●要点

IRubberBand.TrackNew 方法,

IActiveview.ScreenDisplay.StartDrawing, DrawPolygon 和 EndDrawing 方法

●程序说明

通过 IRubberBand.TrackNew 方法实现 Polygon 的作成,

通过 IActiveview.ScreenDisplay.StartDrawing, DrawPolygon 和 EndDrawing 方法来绘制 Polygon

通过 Map 事件的重载，使绘制的 Polygon 不会因为 Map 的刷新而消失

●代码


```

Private m_pPolygon          As IPolygon
Private m_pFillSymbol       As IFillSymbol
Private m_pScreenDisplay    As IScreenDisplay
' 事件重载
Private WithEvents ActiveViewEvents As Map

' 重新绘制 Polygon
Private Sub ActiveViewEvents_AfterDraw(ByVal Display As IDisplay, ByVal phase As esriViewDrawPhase)
    If Not phase = esriDPGeography Then Exit Sub
    If m_pPolygon Is Nothing Then Exit Sub
    With m_pScreenDisplay
        .SetSymbol m_pFillSymbol
        .DrawPolygon m_pPolygon
    End With
End Sub

' 事件初始化
Private Sub UIToolControl1_Select()
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    Set ActiveViewEvents = pMxDoc.FocusMap
End Sub

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, _
    ByVal x As Long, ByVal y As Long)

    Dim pMxDoc As IMxDocument
    Dim pActiveView As IActiveView
    Dim pRubberPolygon As IRubberBand
    Dim pFillSymbol As ISimpleFillSymbol
    Dim pRgbColor As IRgbColor
    Dim pPolygon As IPolygon

    Set pMxDoc = Application.Document
    Set pActiveView = pMxDoc.FocusMap
    Set m_pScreenDisplay = pActiveView.ScreenDisplay
    Set pRubberPolygon = New RubberPolygon

    Set m_pFillSymbol = New SimpleFillSymbol
    Set pRgbColor = New RgbColor
    pRgbColor.Red = 255
    m_pFillSymbol.Color = pRgbColor

' 获得 Polygon
Set m_pPolygon = pRubberPolygon.TrackNew(m_pScreenDisplay, pFillSymbol)

' 绘制 Polygon
With m_pScreenDisplay
    .StartDrawing m_pScreenDisplay.hDC, esriNoScreenCache
    .SetSymbol m_pFillSymbol
    .DrawPolygon m_pPolygon
    .FinishDrawing
End With

```

1.4.4. 如何实现在 ArcMap 上进行测量

距离的测量：要实现的是测量两个点之间的距离。用 ToolControl 实现，选中工具后，在测量的开始点按鼠标左键，鼠标拖动过程中实时画一条从开始点到鼠标当前位置的橡皮线，计算并显示开始点到目标点的距离，释放左键后擦除所画的线和文本。

● 要点

要实时显示结果，计算过程应该在 MouseMove 事件中处理。

要绘制橡皮线，必须设置当前绘图模式为 esriROPXorPen。即混合后的颜色取为当前背景色和画笔颜色的“异或”结果。这样在设定了画笔颜色后，在同一位置第二次画同一图形，就会将图形“擦除”，并恢复原来的背景色。

所有的对设备（包括显示器、打印机、内存位图）的绘图操作前后都应分别调用 IDisplay 的两个方法 StartDrawing 和 EndDrawing。StartDrawing 可以准备特定的设备环境，管理本例中要用到的各种 Symbols，FinishDrawing 完成收尾工作，以保证下一次对 StartDrawing 的调用不会出错。

● 程序说明

函数 UITMeasureDistance_Deactivate 是 Deactivate 属性的处理代码，当工具失去焦点时，清除已创建的对象。过程 UITMeasureDistance_MouseDown 是 MouseDown 事件的处理代码，当鼠标键按下时，记录起始点。过程 UITMeasureDistance_MouseMove 是 MouseMove 事件的处理代码，鼠标移动过程中测量距离，计算文本显示的角度，以及完成屏幕上橡皮线的绘制。过程 UITMeasureDistance_MouseUp 是 MouseUp 事件的处理代码，当释放鼠标键时，擦除刚绘制的图形。函数 GetSmashedLine 将获得一条 IPolyline 对象，这条 Polyline 在要显示文本的地方留下了空白，以防止出现所画线穿过文字的现象。

另外，本例未考虑坐标系的转换，在球面地理坐标系是测量结果为经度差或纬度差。

● 代码

```
Option Explicit
Private m_bInUse      As Boolean
Private m_pLineSymbol As ILineSymbol
Private m_pLinePolyline As IPolyline
Private m_pTextSymbol As ITextSymbol
Private m_pStartPoint As IPoint
Private m_pTextPoint As IPoint
```

```

Private Function UITMeasureDistance_Deactivate() As Boolean
    ' Stop doing operation
    Set m_pTextSymbol = Nothing
    Set m_pTextPoint = Nothing
    Set m_pLinePolyline = Nothing
    Set m_pLineSymbol = Nothing
    m_bInUse = False
    UITMeasureDistance_Deactivate = True
End Function

Private Sub UITMeasureDistance_MouseDown(ByVal Button As Long, ByVal Shift As Long, _
    ByVal X As Long, ByVal Y As Long)

    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView

    m_bInUse = True
    Set pMxDocument = ThisDocument
    Set pActiveView = pMxDocument.FocusMap

    ' Get point to measure distance from
    Set m_pStartPoint = pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(X, Y)
End Sub

Private Sub UITMeasureDistance_MouseMove(ByVal Button As Long, ByVal Shift As Long, _
    ByVal X As Long, ByVal Y As Long)

    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim bFirstTime As Boolean
    Dim pPoint As IPoint
    Dim pRGBColor As IRgbColor
    Dim pSymbol As ISymbol
    Dim pFont As IFontDisp
    Dim pLine As ILine
    Dim dAngle As Double
    Dim dDeltaX As Double
    Dim dDeltaY As Double
    Dim dDistance As Double
    Dim pPolyLine As IPolyline
    Dim pSegmentCollection As ISegmentCollection

    On Error GoTo ErrorHandler
    If (Not m_bInUse) Then
        Exit Sub
    End If

    Set pMxDocument = ThisDocument
    Set pActiveView = pMxDocument.FocusMap

    If (m_pLineSymbol Is Nothing) Then
        bFirstTime = True
    End If

    ' Get current point
    Set pPoint = pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(X, Y)

```

```

pActiveView.ScreenDisplay.StartDrawing pActiveView.ScreenDisplay.hDC, -1

If bFirstTime Then
    ' Set Line Symbol
    Set m_pLineSymbol = New SimpleLineSymbol
    m_pLineSymbol.Width = 2
    Set pRGBColor = New RGBColor
    With pRGBColor
        .Red = 222
        .Green = 222
        .Blue = 222
    End With

    m_pLineSymbol.Color = pRGBColor
    Set pSymbol = m_pLineSymbol
    pSymbol.ROP2 = esriROPXORPen

    ' Set Text Symbol
    Set m_pTextSymbol = New TextSymbol
    m_pTextSymbol.HorizontalAlignment = esriTHACenter
    m_pTextSymbol.VerticalAlignment = esriTVACenter
    m_pTextSymbol.Size = 16
    Set pSymbol = m_pTextSymbol
    Set pFont = m_pTextSymbol.Font
    pFont.Name = "Arial"
    pSymbol.ROP2 = esriROPXORPen

    ' Create point to draw text in
    Set m_pTextPoint = New Point
Else
    ' Use existing symbols and draw existing text and polyline
    pActiveView.ScreenDisplay.SetSymbol m_pTextSymbol
    pActiveView.ScreenDisplay.DrawText m_pTextPoint, m_pTextSymbol.Text
    pActiveView.ScreenDisplay.SetSymbol m_pLineSymbol
    If (m_pLinePolyline.Length > 0) Then
        pActiveView.ScreenDisplay.DrawPolyline m_pLinePolyline
    End If
End If

' Get line between from and to points, and dAngle for text
Set pLine = New esriCore.Line
pLine.PutCoords m_pStartPoint, pPoint

dAngle = pLine.angle
dAngle = dAngle * (180# / 3.14159)
If ((dAngle > 90#) And (dAngle < 180#)) Then
    dAngle = dAngle + 180#
ElseIf ((dAngle < 0#) And (dAngle < -90#)) Then
    dAngle = dAngle - 180#
ElseIf ((dAngle < -90#) And (dAngle > -180#)) Then
    dAngle = dAngle - 180#
ElseIf (dAngle > 180) Then

```

```

        dAngle = dAngle - 180#
    End If
    ' For drawing text, get text(dDistance), dAngle, and point
    dDeltaX = pPoint.X - m_pStartPoint.X
    dDeltaY = pPoint.Y - m_pStartPoint.Y
    m_pTextPoint.X = m_pStartPoint.X + dDeltaX / 2#
    m_pTextPoint.Y = m_pStartPoint.Y + dDeltaY / 2#
    m_pTextSymbol.angle = dAngle

    dDistance = Round(pLine.Length, 3)
    m_pTextSymbol.Text = "[" & dDistance & "]"

    ' Draw text
    pActiveView.ScreenDisplay.SetSymbol m_pTextSymbol
    pActiveView.ScreenDisplay.DrawText m_pTextPoint, m_pTextSymbol.Text
    ' Get polyline with blank space for text
    Set pPolyLine = New Polyline
    Set pSegmentCollection = pPolyLine
    pSegmentCollection.AddSegment pLine
    Set m_pLinePolyline = GetSmashedLine(pActiveView.ScreenDisplay, m_pTextSymbol, _
                                         m_pTextPoint, pPolyLine)

    ' Draw polyline
    pActiveView.ScreenDisplay.SetSymbol m_pLineSymbol
    If (m_pLinePolyline.Length > 0) Then
        pActiveView.ScreenDisplay.DrawPolyline m_pLinePolyline
    End If

    pActiveView.ScreenDisplay.FinishDrawing

    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UITMeasureDistance_MouseUp(ByVal Button As Long, ByVal Shift As Long, _
                                       ByVal X As Long, ByVal Y As Long)

    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView

    On Error GoTo ErrorHandler
    If (Not m_bInUse) Then
        Exit Sub
    End If
    m_bInUse = False

    If (m_pLineSymbol Is Nothing) Then
        Exit Sub
    End If

    Set pMxDocument = ThisDocument
    Set pActiveView = pMxDocument.FocusMap

```

```

'Draw measure line and text
pActiveView.ScreenDisplay.StartDrawing pActiveView.ScreenDisplay.hDC, -1

pActiveView.ScreenDisplay.SetSymbol m_pTextSymbol
pActiveView.ScreenDisplay.DrawText m_pTextPoint, m_pTextSymbol.Text
pActiveView.ScreenDisplay.SetSymbol m_pLineSymbol

If (m_pLinePolyline.Length > 0) Then
    pActiveView.ScreenDisplay.DrawPolyline m_pLinePolyline
End If

pActiveView.ScreenDisplay.FinishDrawing

Set m_pTextSymbol = Nothing
Set m_pTextPoint = Nothing
Set m_pLinePolyline = Nothing
Set m_pLineSymbol = Nothing
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Function GetSmashedLine(pDisplay As IScreenDisplay, pTextSymbol As ISymbol, _
                                pPoint As IPoint, pPolyLine As IPolyline) As IPolyline
    ' Returns a Polyline with a blank space for the text to go in
    Dim pSmashed As IPolyline
    Dim pBoundary As IPolygon
    Dim pTopologicalOperator As ITopologicalOperator
    Dim pIntersect As IPolyline

    On Error GoTo ErrorHandler
    Set pBoundary = New Polygon
    pTextSymbol.QueryBoundary pDisplay.hDC, pDisplay.DisplayTransformation, pPoint,
pBoundary

    Set pTopologicalOperator = pBoundary
    Set pIntersect = pTopologicalOperator.Intersect(pPolyLine, esriGeometryDimension)
    Set pTopologicalOperator = pPolyLine
    Set GetSmashedLine = pTopologicalOperator.Difference(pIntersect)
    Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

```

本例要实现的是如何在 ArcMap 上测量一个 Polygon 的面积。

● 要点

首先用 IRubberBand.TrackNew 方法在 ArcMap 上画出一个 Polygon，然后由这个 Polygon 获得一个 IArea 的实例，最后使用 IArea.Area 方法计算出这个 Polygon 的面积。

主要用到 IRubberBand 接口, IPolygon 接口和 IArea 接口。

- 程序说明

函数 DrawPolygon 实现在 ArcMap 上画一个 Polygon。

函数 MeasurePolygon 实现测量 pPolygon 的面积。

- 代码

```
Private Function DrawPolygon() As IPolygon
    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pSimpleFillS As ISimpleFillSymbol
    Dim pRgbColor As IRgbColor
    Dim pRubberBand As IRubberBand
    Dim pPolygon As IPolygon

    On Error GoTo ErrorHandler:
        Set pMxDocument = ThisDocument
        Set pActiveView = pMxDocument.ActiveView
        Set pSimpleFillS = New SimpleFillSymbol
        Set pRgbColor = New RgbColor
        pRgbColor.Red = 255
        pSimpleFillS.Color = pRgbColor
        Set pRubberBand = New esriCore.RubberPolygon
        Set pPolygon = pRubberBand.TrackNew(pActiveView.ScreenDisplay, pSimpleFill)
        With pActiveView.ScreenDisplay
            .StartDrawing pActiveView.ScreenDisplay.hDC, esriNoScreenCache
            .SetSymbol pSimpleFillS
            .DrawPolygon pPolygon
            .FinishDrawing
        End With
        Set DrawPolygon = pPolygon
        Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Function MeasurePolygon(pPolygon As IPolygon) As Double
    Dim pArea As IArea

    On Error GoTo ErrorHandler:
        Set pArea = pPolygon
        MeasurePolygon = Abs(pArea.Area())
        Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, _
    ByVal x As Long, ByVal y As Long)

    Dim pPolygon As IPolygon
    Dim dArea As Double

    On Error GoTo ErrorHandler:
```

```

Set pPolygon = DrawPolygon()
dArea = MeasurePolygon(pPolygon)
MsgBox "面积为: " & dArea
Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

```

1.4.5. 如何实现在 ArcMap 上选取中记录

本例要演示的是如何通过鼠标点击在 ArcMap 上选中一条记录。用到 IMap 接口的 SelectByShape 方法。

● 要点

Tool Control 的 MouseDown 事件发生时传入了鼠标的位置参数 X 和 Y，此为鼠标位置的设备坐标（屏幕坐标），要转换成逻辑坐标（地理坐标）后，SelectByShape 方法才能正常工作。

在 SelectByShape 方法中指定参数为 IPoint 对象。可选中处于当前鼠标位置（点）的记录。

● 程序说明

本例功能的实现代码较为简单，故只处理了 MouseDown 事件。为了 Tool Control 未被选中时按钮能够弹起，设置了其 Deactivate 属性为 True。

● 代码

```

Option Explicit

Private Function UITSelectFeature_Deactivate() As Boolean
    UITSelectFeature_Deactivate = True
End Function

Private Sub UITSelectFeature_MouseDown(ByVal Button As Long, ByVal Shift As Long, _
    ByVal X As Long, ByVal Y As Long)

    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pPoint As IPoint

    On Error GoTo ErrorHandler

    ' Get the ActiveView for the map
    Set pMxDocument = ThisDocument
    Set pActiveView = pMxDocument.FocusMap

    ' Store current mouse point
    Set pPoint = pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(X, Y)

    ' Select by point
    pMxDocument.FocusMap.SelectByShape pPoint, Nothing, False

```



```

' Refresh the selections
pActiveView.PartialRefresh esriViewGeoSelection, Nothing, Nothing

Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

```

1.4.6. 如何在 ArcMap 中进行动作的撤销和重做

本例要演示的是如何在 ArcMap 中对图形的移动动作进行撤销和重做，用到 IExtentStack 接口。以帮助理解 ArcMap 中对撤销和重做实现的方法。

● 要点

IActiveView 的 ExtentStack 属性保存了其 Extent 改变的“历史记录”，而 IMxDocument 的 OperationStack 属性则有能力记录更复杂的编辑动作的历史。用户只有深刻理解了概念，才能够完成特定功能“历史记录”的定制。

● 程序说明

过程 Extent_UnDo 和 Extent_RnDo 分别模拟了 ArcMap 中 Tools 工具栏上的“Go Back To Previous Extent”和“Go To Next Extent”两个按钮的功能。

● 代码

```

Option Explicit

Public Sub Extent_UnDo()
Dim pMxDocument As IMxDocument
Dim pActiveView As IActiveView
Dim pExtentStack As IExtentStack

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pActiveView = pMxDocument.FocusMap
Set pExtentStack = pActiveView.ExtentStack

If pExtentStack.CanUndo Then
pExtentStack.Undo
End If

Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

Public Sub Extent_ReDo()
Dim pMxDocument As IMxDocument
Dim pActiveView As IActiveView
Dim pExtentStack As IExtentStack

```

```

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pActiveView = pMxDocument.FocusMap
Set pExtentStack = pActiveView.ExtentStack

If pExtentStack.CanRedo Then
    pExtentStack.Redo
End If

Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.4.7. 如何画 Polygon Buffers

本例要实现的是如何利用 Polygon Buffer 自定义记录选中时的显示方式。

● 要点

首先通过 IRgbColor 接口和 ISimpleFillSymbol 接口设置 Polygon Buffer 的填充方式。然后在发生 SelectionChanged 事件时，设置选中记录被显示时的边界并将选中的 Polygon 通过 ITopologicalOperator.ConstructUnion 方法，联合成一个临时的 Polygon Buffer，使用 IActiveView.PartialRefresh 方法刷新这个 Polygon Buffer 区域，最后在发生 AfterItemDraw 事件时将这个 Polygon Buffer 画在 Map 上。

主要用到 IPolygon 接口，IEnvelope 接口，ISimpleFillSymbol 接口，IActiveView 接口，IEnumFeature 接口，IGeometryCollection 接口和 ITopologicalOperator 接口。

● 程序说明

函数 InitEvents 是初始化变量并设置 Polygon Buffer 的填充方式。

AfterItemDraw 事件实现的是画出 Polygon Buffer。

SelectionChanged 事件实现的是生成 Polygon Buffer 并设置边界。

● 代码

```

Private WithEvents ActiveViewEvents As Map

Private pMxDocument As IMxDocument
Private pBufferPolygon As IPolygon
Private pEnvelope As IEnvelope
Private pSimpleFillS As ISimpleFillSymbol

Public Sub InitEvents()
    Dim pViewManager As IViewManager

```

```

Dim pRgbColor As IRgbColor

Set pMxDocument = Application.Document
Set pViewManager = pMxDocument.FocusMap
pViewManager.VerboseEvents = True
Set ActiveViewEvents = pMxDocument.FocusMap

'Create a fill symbol
Set pSimpleFillS = New SimpleFillSymbol
Set pRgbColor = New RgbColor
pRgbColor.Red = 255
pSimpleFillS.Style = esriSFSForwardDiagonal
pSimpleFillS.Color = pRgbColor
End Sub

Private Sub ActiveViewEvents_AfterItemDraw(ByVal Index As Integer, ByVal Display As IDisplay, ByVal phase As esriDrawPhase)

'Only draw in the geography phase
If Not phase = esriDPGeography Then Exit Sub
'Draw the buffered polygon
If pBufferPolygon Is Nothing Then Exit Sub
With Display
.SetSymbol pSimpleFillS
.DrawPolygon pBufferPolygon
End With
End Sub

Private Sub ActiveViewEvents_SelectionChanged()
Dim pActiveView As IActiveView
Dim pEnumFeature As IEnumFeature
Dim pFeature As IFeature
Dim pSelectionPolygon As IPolygon
Dim pTopologicalOperator As ITopologicalOperator
Dim pGeometryCollection As IGeometryCollection
Set pActiveView = pMxDocument.FocusMap
Set pGeometryCollection = New GeometryBag

'Flag last buffered region for invalidation
If Not pEnvelope Is Nothing Then
pActiveView.PartialRefresh esriViewGeography, Nothing, pEnvelope
End If

If pMxDocument.FocusMap.SelectionCount = 0 Then
'Nothing selected; don't draw anything; bail
Set pBufferPolygon = Nothing
Exit Sub
End If

'Buffer each selected feature
Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
pEnumFeature.Reset
Set pFeature = pEnumFeature.Next
Do While Not pFeature Is Nothing
Set pTopologicalOperator = pFeature.Shape

```

```

        Set pSelectionPolygon = pTopologicalOperator.Buffer(0.1)
        pGeometryCollection.AddGeometry pSelectionPolygon
        'Get next feature
        Set pFeature = pEnumFeature.Next
    Loop

    'Union all the buffers into one polygon
    Set pBufferPolygon = New Polygon
    Set pTopologicalOperator = pBufferPolygon 'QI
    pTopologicalOperator.ConstructUnion pGeometryCollection

    Set pEnvelope = pBufferPolygon.Envelope

    'Flag new buffered region for invalidation
    pActiveView.PartialRefresh esriViewGeography, Nothing, pBufferPolygon.Envelope
End Sub

Private Sub UIButtonControl1_Click()
    InitEvents
End Sub

```

1.5. 图元编辑

1.5.1. 如何得到图形的基本属性

本例要实现的功能是得到一个 FeatureLayer 中被选择的 Feature 的基本图形属性，如，图形的维数，类型，范围，空间坐标系统等。

●要点

接口 IGeometry 的主要属性有 Dimension(维数), GeometryType(图形类型), Envelope(范围), IsEmpty(是否为空), SpatialReference(空间坐标系)等。

●程序说明

该过程在开始处使用 IEnumFeature 接口来得到所选择的 Features, 用 Next 方法取得每个 Feature。然后利用 IFeature 接口的 Shape 属性得到 Geometry。最后弹出消息框显示图形的属性信息。

●代码

```

Public Sub GetGeometryProperty()
    Dim pMxDocument As IMxDocument
    Dim pEnumFeature As IEnumFeature
    Dim pFeature As IFeature
    Dim pGeometry As IGeometry

    On Error GoTo ErrorHandler
    Set pMxDocument = Application.Document
    '得到图形集
    Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
    '重新设置图形集

```

```

pEnumFeature.Reset
' 得到第一个图形
Set pFeature = pEnumFeature.Next
' 判断是否有图形被选中
If pFeature Is Nothing Then
    MsgBox "no selection, please select a Feature"
Else
    ' 循环图形, 直到最后
    While Not pFeature Is Nothing
        Set pGeometry = pFeature.Shape
        ' 得到图形的基本属性
        MsgBox "+++Polygon::IGeometry properties..." & vbCrLf _
            & "Dimension = " & pGeometry.Dimension & vbCrLf _
            & "Geometry type = " & pGeometry.GeometryType & vbCrLf _
            & "Envelope = " & pGeometry.Envelope.XMin & ", " & pGeometry.Envelope.YMin & ", "
            & pGeometry.Envelope.XMax & ", " & pGeometry.Envelope.YMax & vbCrLf _
            & "IsEmpty = " & pGeometry.IsEmpty & vbCrLf _
            & "SpatialReference = " & pGeometry.SpatialReference.Name
        ' 指向下一个图形
        Set pFeature = pEnumFeature.Next
    Wend
End If
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.5.2. 如何将选中的点集转换成 Polygon

本例要实现的功能是根据选中的 Points 创建一个 Polygon, 并且保存到 Polygon 类型的 FeatureLayer 中, 要求被选择的 Points 最少为 3 个。

● 要点

根据选择的点创建一个 Polygon, 首先要判断生成的 Polygon 是否是 Simple, 这里用到接口 ITopologicalOperator2 的属性 IsSimple。如果不是, 则要对做 Polygon 排序等处理。此外还用到了接口 IPointCollection 的方法 ReplacePoints, 进行点的交换。将排好序的点, 按顺序创建 Segment, 运用实例化为 Ring 的 ISegmentCollection 接口方法 AddSegment 增加 Segment。实例化为 Polygon 的 IGeometryCollection 接口方法 AddGeometry 增加 Ring。这样, 通过上面的方法便可以创建 Polygon。

● 程序说明

根据接口 ITopologicalOperator2.IsSimple 属性判断 Polygon 是否 Simple。如果返回为 False, 就对 Polygon 上的点进行排序等处理, 排好序后, 找出 X 方

向上值最大和最小的点，由这两点创建一条直线，将所有点分成在直线左边和右边两部分。

● 代码

```
Public Sub ConvertPointToPolygon()  
    Dim pMxDoc As IMxDocument  
    Dim pMap As IMap  
    Dim pEnumFeature As IEnumFeature  
    Dim pMultiPoint As IPointCollection  
    Dim pMultiPointSorted As IPointCollection  
    Dim pFeature As IFeature  
    Dim pPointi As IPoint  
    Dim pTopoOp As ITopologicalOperator2  
    Dim pLine As ILine  
    Dim pGonColl As IPointCollection  
    Dim pClonei As IClone  
    Dim ptMin As IPoint  
    Dim ptMax As IPoint  
    Dim pBaseLine As ILine  
    Dim pBaseCurve As ICurve  
    Dim pOutpoint As IPoint  
    Dim pMultiRight As IPointCollection  
    Dim pMultiLeft As IPointCollection  
    Dim pGonColl2 As IGeometryCollection  
    Dim pPolygon As IPolygon  
    Dim pRing As IRing  
    Dim pFeatureClass As IFeatureClass  
    Dim pFeatureLayer As IFeatureLayer  
    Dim pFeature1 As IFeature  
    Dim pFeatureClass1 As IFeatureClass  
    Dim pFeatureLayer1 As IFeatureLayer  
    Dim pDataSet As IDataset  
    Dim pWorkspaceFactory As IWorkspaceFactory  
    Dim pWorkspaceEdit As IWorkspaceEdit  
    Dim pRingColl As ISegmentCollection  
    Dim dDistAlong As Double  
    Dim dDistFrom As Double  
    Dim bIsRight As Boolean  
    Dim i As Long  
    Dim j As Long  
    Dim lFlag As Long  
    On Error GoTo errorHandler  
    Set pMxDoc = ThisDocument  
    Set pMap = pMxDoc.FocusMap  
    Set pActiveView = pMap  
    Set pFeatureLayer = pMap.Layer(0)  
    Set pFeatureClass = pFeatureLayer.FeatureClass  
    ' 创建一个工作区，开始编辑  
    Set pDataSet = pFeatureClass  
    Set pWorkspaceFactory = New ShapefileWorkspaceFactory  
    Set pWorkspaceEdit = pWorkspaceFactory.OpenFromFile(pDataSet.Workspace.Path & "\", 0)  
    pWorkspaceEdit.StartEditOperation  
    pWorkspaceEdit.StartEditing True
```

```

Set pMultiLeft = New Multipoint
Set pMultiRight = New Multipoint
Set pGonColl = New Polygon
Set pMultiPoint = New Multipoint
Set pMultiPointSorted = New Multipoint
'得到所选择的图形集
Set pEnumFeature = pMxDoc.FocusMap.FeatureSelection
Set pFeature = pEnumFeature.Next
'增加点到 MultiPoint
While Not pFeature Is Nothing
    If pFeature.ShapeCopy.GeometryType = esriGeometryPoint Then
        pMultiPoint.AddPoint pFeature.ShapeCopy
    ElseIf pFeature.ShapeCopy.GeometryType = esriGeometryMultipoint Then
        pMultiPoint.AddPointCollection pFeature.ShapeCopy
    End If
    Set pFeature = pEnumFeature.Next
Wend
If pMultiPoint.PointCount < 3 Then
    MsgBox "Select a least 3 points !"
    Exit Sub
End If
'创建第一个 Polygon
pGonColl.AddPointCollection pMultiPoint
Set pTopoOp = pGonColl
'将 Polygon 是否是 Simple 设置成未知
pTopoOp.IsKnownSimple = False
'经判断, 如果不是 Simple, 则经过以下处理, 将其转换为 Simple
If pTopoOp.IsSimple = False and pMultiPoint.PointCount > 3 Then
    lFlag = 1
    Set pTopoOp = pMultiPoint
    pTopoOp.IsKnownSimple = False
    pTopoOp.Simplify
    '将 Multipoint 进行排序
    For i = 0 To pMultiPoint.PointCount - 1
        For j = i + 1 To pMultiPoint.PointCount - 1
            If pMultiPoint.Point(j).x < pMultiPoint.Point(i).x Or pMultiPoint.Point(i).x = _
                pMultiPoint.Point(i).x And_ pMultiPoint.Point(j).y < _
                pMultiPoint.Point(i).y Then
                Set pClonei = pMultiPoint.Point(i)
                Set pPointi = pClonei.Clone
                '交换两点
                pMultiPoint.ReplacePoints i, 1, 1, pMultiPoint.Point(j)
                pMultiPoint.ReplacePoints j, 1, 1, pPointi
            End If
        Next
    Next
    Set ptMin = New Point
    Set ptMax = New Point
    '找出 MultiPoint 中的最大和最小点
    pMultiPoint.QueryPoint 0, ptMin
    pMultiPoint.QueryPoint pMultiPoint.PointCount - 1, ptMax
    '创建一条线段
    Set pBaseLine = New Line

```

```

pBaseLine.PutCoords ptMin, ptMax
Set pBaseCurve = pBaseLine
For i = 0 To pMultiPoint.PointCount - 1
    Set pOutpoint = New Point
    pBaseCurve.QueryPointAndDistance esriNoExtension, pMultiPoint.Point(i), False,
    pOutpoint, _ dDistAlong, dDistFrom, bIsRight

    If bIsRight Then
        pMultiRight.AddPoint pMultiPoint.Point(i)
    Else
        pMultiLeft.AddPoint pMultiPoint.Point(i)
    End If
Next
Set pRingColl = New Ring
' 将左边的线添加到 Ring
For i = 0 To pMultiLeft.PointCount - 2
    Set pLine = New Line
    pLine.PutCoords pMultiLeft.Point(i), pMultiLeft.Point(i + 1)
    pRingColl.AddSegment pLine
Next
' 第一条线
Set pLine = New Line
pLine.PutCoords pMultiLeft.Point(pMultiLeft.PointCount - 1), pMultiRight.Point(0)
pRingColl.AddSegment pLine
' 将右边的先添加到 Ring
For i = (pMultiRight.PointCount - 1) To 1 Step -1
    Set pLine = New Line
    pLine.PutCoords pMultiRight.Point(i), pMultiRight.Point(i - 1)
    pRingColl.AddSegment pLine
Next
' 最后一条线
Set pLine = New Line
pLine.PutCoords pMultiRight.Point(0), pMultiLeft.Point(0)
pRingColl.AddSegment pLine
Set pRing = pRingColl
pRing.Close
Set pGonColl2 = New Polygon
pGonColl2.AddGeometry pRing
End If
If lFlag = 0 Then
    Set pPolygon = pGonColl
Else
    Set pPolygon = pGonColl2 ' QI
End If
' 画出 Polygon
Set pFeatureLayer1 = pMap.Layer(1)
Set pFeatureClass1 = pFeatureLayer1.FeatureClass
Set pFeature1 = pFeatureClass1.CreateFeature
' 把画的 Polygon 加到新建的 Feature 上
Set pFeature1.Shape = pPolygon
' 保存 Feature
pFeature1.Store
pMxDoc.ActiveView.Refresh
' 停止编辑
pWorkspaceEdit.StopEditOperation
pWorkspaceEdit.StopEditing True

```



```
Exit Sub

ErrorHandler:
    pWorkspaceEdit.AbortEditOperation
    MsgBox Err.Description
End Sub
```

1.5.3. 如何将 Multipoint 转换成 Points

本例要实现的功能是根据一个 FeatureLayer 中被选择一个或多个 MultiPoint,生成多个 Point 并把这些新生成的 Point 保存在一个 Point 类型的 Feature Layer 上。

●要点

本例将选择的 Multipoints 上的每个点都生成一个对应得 Point,并用一个接口 IPointCollection 的变量来接收。利用 IPointCollection 的方法 point(index),取出新生成的每个点,用来创建 Point 类型的 Feature。

●程序说明

本例要求在 ArcMap 中添加两个层,最上面的是层 Multipoint,下面是层 wind。根据循环得到选择的每个 Multipoint 的每个点,为 wind 层生成新的 Feature 并保存

●代码

```
Sub convertMultipointToPoints()
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pActiveView As IActiveView
    Dim pEnumFeature As IEnumFeature
    Dim pFeature0 As IFeature
    Dim pFeatureLayer0 As IFeatureLayer
    Dim pFeatureClass0 As IFeatureClass
    Dim pFeature1 As IFeature
    Dim pFeatureLayer1 As IFeatureLayer
    Dim pFeatureClass1 As IFeatureClass
    Dim pPointCollection As IPointCollection
    Dim pDataSet As IDataset
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pWorkspaceEdit As IWorkspaceEdit
    Dim lPointIndex As Long
    Dim lPointFieldIndex As Long

    On Error GoTo ErrorHandler
    '得到当前层
    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
    Set pActiveView = pMap
    '得到 0 层和 1 层的 FeatureClass
```

```

Set pFeatureLayer0 = pMxDocument.FocusMap.Layer(0)
Set pFeatureClass0 = pFeatureLayer0.FeatureClass
Set pFeatureLayer1 = pMxDocument.FocusMap.Layer(1)
Set pFeatureClass1 = pFeatureLayer1.FeatureClass
' 建立编辑工作区
Set pDataSet = pFeatureClass1
Set pWorkspaceFactory = New ShapefileWorkspaceFactory
Set pWorkspaceEdit = pWorkspaceFactory.OpenFromFile(pDataSet.Workspace.Path & ".sde", 0)
pWorkspaceEdit.StartEditOperation
pWorkspaceEdit.StartEditing True
' 得到 Feature
Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
Set pFeature0 = pEnumFeature.Next

If pFeature0 Is Nothing Then
    MsgBox "Must have Select in Position 0"
    Exit Sub
End If

' 循环, 通过每个 MultiPoint, 在 1 图层上, 生成以每个点为特征的 Points
While Not pFeature0 Is Nothing
    If pFeature0.ShapeCopy.GeometryType = esriGeometryMultipoint Then
        Set pPointCollection = pFeature0.ShapeCopy
        For nPointIndex = 0 To pPointCollection.PointCount - 1
            Set pFeature1 = pFeatureClass1.CreateFeature
            ' 在 pFeature1 上生成 Point
            Set pFeature1.Shape = pPointCollection.Point(nPointIndex)
            ' 如果两 Feature 的 FieldCount 相同, 赋每个 Field 的值, ID,
            ' TypeGeometry 的 Field 除外
            If pFeature1.Fields.FieldCount = pFeature0.Fields.FieldCount Then
                For lPointFieldIndex = 0 To pFeature1.Fields.FieldCount - 1
                    If Not pFeature1.Fields.Field(lPointFieldIndex).Type = _
                        esriFieldTypeGeometry And Not pFeature1.Fields._
                        Field(lPointFieldIndex).Type = esriFieldTypeOID Then
                        pFeature1.Value(lPointFieldIndex) = _
                            pFeature0.Value(lPointFieldIndex)
                    End If
                Next
            End If
            ' 保存 Feature
            pFeature1.Store
        Next
    Else
        MsgBox "Must have Multipoint in position 0"
        Exit Sub
    End If
    Set pFeature0 = pEnumFeature.Next
Wend
' 停止编辑
pWorkspaceEdit.StopEditOperation
pWorkspaceEdit.StopEditing True
Exit Sub

ErrorHandler:
pWorkspaceEdit.AbortEditOperation

```

```
MsgBox Err.Description
End Sub
```

1.5.4. 如何通过 Polygon 中的多个 Ring 创建多个 Polygon

本例要实现的是如何在一个 FeatureLayer 中, 选择 Polygon (Feature) 的 Shape, 如果它有多个 Ring, 则在另一个 Polygon 的图层上根据每一个 Ring 创建一个 Polygon。

● 要点

取出 Polygon 中的每个 Ring, 声明一个 IGeometryCollection 接口, 将其实例化为 Polygon, 利用此接口的方法 AddGeometry 生成一个 Polygon, 再用一个实例化为 GeometryBag 的 IGeometryCollection 接口变量来放置生成的每个 Polygon。

● 程序说明

程序中添加了两个图层, 两层都是 Polylygon 型。在第一个层选择有多个 Ring 的 Polygon, 再运行本函数, 则在第二个层由这些多个 Ring 的 Polygon 创建生成了多个 Polygon。

● 代码

```
Private Function PolygonsFromPolygonRings(pGeomColl As IGeometryCollection, bClone As Boolean) As _ IGeometryCollection
    Dim i As Long
    Dim pGeometryCollection As IGeometryCollection
    Dim pTopologicalOperator As ITopologicalOperator

    If Not pGeomColl Is Nothing Then
        If pGeomColl.GeometryCount > 0 Then
            Set PolygonsFromPolygonRings = New GeometryBag
            If bClone Then
                If TypeOf pGeomColl Is IClone Then
                    Dim pClone As IClone
                    Set pClone = pGeomColl
                    Set pGeomColl = pClone.Clone
                End If
            End If
            ' 为每个 Ring 创建一个新 Polygon, 将 Polygon 进行 simplify 后, 放在 GeometryBag 中
            For i = 0 To pGeomColl.GeometryCount - 1
                If pGeomColl.Geometry(i).GeometryType = esriGeometryRing Then
                    Set pGeometryCollection = New Polygon
                    Set pTopologicalOperator = pGeometryCollection
                    pGeometryCollection.AddGeometry pGeomColl.Geometry(i)
                    pTopologicalOperator.Simplify
                    PolygonsFromPolygonRings.AddGeometry pGeometryCollection
                End If
            Next i
        End If
    End If
End Function
```

```

End Function

Public Sub PolygonRingsToPolygons()
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pActiveView As IActiveView
    Dim pEnumFeature As IEnumFeature
    Dim pFeature0 As IFeature
    Dim pFeatureLayer0 As IFeatureLayer
    Dim pFeatureClass0 As IFeatureClass
    Dim pFeature1 As IFeature
    Dim pFeatureLayer1 As IFeatureLayer
    Dim pFeatureClass1 As IFeatureClass
    Dim pPointCollection As IPointCollection
    Dim pGeometryCollection As IGeometryCollection
    Dim pDataSet As IDataset
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pWorkspaceEdit As IWorkspaceEdit
    Dim pPolygon As IPolygon
    Dim pGeometryColPolygon As IGeometryCollection
    Dim pGeometryColPolygonNew As IGeometryCollection
    Dim pGeometryCollectionPolygon As IGeometryCollection
    Dim lGeometryIndex As Long
    Dim lPointFieldIndex As Long

    On Error GoTo ErrorHandler
    '得到当前层
    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
    Set pActiveView = pMap
    Set pPolygon1 = New Polygon
    Set pGeometryColPolygon = New Polygon
    Set pGeometryColPolygonNew = New Polygon
    Set pGeometryColPolygonNew1 = New Polygon
    Set pGeometryCollectionPolygon = New GeometryBag
    '得到 0 层和 1 层的 FeatureClass
    Set pFeatureLayer0 = pMxDocument.FocusMap.Layer(0)
    Set pFeatureClass0 = pFeatureLayer0.FeatureClass
    Set pFeatureLayer1 = pMxDocument.FocusMap.Layer(1)
    Set pFeatureClass1 = pFeatureLayer1.FeatureClass
    '建立编辑工作区
    Set pDataSet = pFeatureClass1
    Set pWorkspaceFactory = New ShapefileWorkspaceFactory
    Set pWorkspaceEdit = pWorkspaceFactory.OpenFromFile(pDataSet.Workspace.Path & ".ame", 0)
    pWorkspaceEdit.StartEditOperation
    pWorkspaceEdit.StartEditing True
    '得到 Feature
    Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
    Set pFeature0 = pEnumFeature.Next
    If pFeature0 Is Nothing Then
        MsgBox "Must have Select in Position 0"
        Exit Sub
    End If
    '将一个 Polygon 上的多个 Ring 转换成多个 Polygon
    Set pGeometryCollectionPolygon = PolygonsFromPolygonRings(pGeometryColPolygonNew,

```

```

True)
' 将转换成的多个 Polygon 添加到第二层上
For lGeometryIndex = 0 To pGeometryCollectionPolygon.GeometryCount - 1
    Set pFeature1 = pFeatureClass1.CreateFeature
    ' 把画的 Polygon 加到新建的 Feature 上
    Set pPolygon1 = pGeometryCollectionPolygon.Geometry(lGeometryIndex)
    Set pFeature1.Shape = pPolygon1
    ' 保存 Feature
    pFeature1.Store
Next
pMxDocument.ActiveView.Refresh
' 停止编辑
pWorkspaceEdit.StopEditOperation
pWorkspaceEdit.StopEditing True
Exit Sub

ErrorHandler:
    pWorkspaceEdit.AbortEditOperation
    MsgBox Err.Description
End Sub

```

1.5.5. 如何从 Polyline 创建 Polygon

本例要实现的功能是根据一个 FeatureLayer 中被选择的一条 Polyline 生成一个 Polygon，并把该 Polygon 做为一个新的 Feature 保存在一个 Polygon 类型的 FeatureLayer 中。

● 要点

通过所选择的 Polyline 创建一个新的 Polygon，即要根据 Polyline 中的每个 Path 生成相应的 Ring。程序中用到 ISegmentCollection 接口，将它实例化为 Ring，利用它的方法 AddSegmentCollection 实现了这一目的。

● 程序说明

程序中添加了两个图层，第一图层 Polyline 型，第二图层 Polygon 型。因为 Polyline 型的图层中不能放 Polygon 型的数据，所以多增加一个 Polygon 层，以便将通过 Polyline 生成的一个新的 Polygon 显示到上面，使得程序运行结果清晰明了。

函数 PolylineToPolygon(ByRef pPolyline As IPolyline) 中，通过 pSegs_Ring.AddSegmentCollection，创建了一个新 Ring，其中 pSegs_Ring 是一个实例化为 Ring 的 ISegmentCollection 接口变量。

● 代码

```

Private Function PolylineToPolygon(ByRef pPolyline As IPolyline) As IGeometryCollection
    Dim pGeoms_Polyline As IGeometryCollection

```

```

Dim pClone                As IClone
Dim pSegs_Ring            As ISegmentCollection
Dim pPolygon              As IPolygon
Dim i                    As Long

On Error Goto ErrorHandler
' 创建一个新的 Polygon geometry.
Set PolylineToPolygon = New Polyline
' 克隆即将要操作的 Polyline
Set pClone = pPolyline
Set pGeoms_Polyline = pClone.Clone
' 通过 Polyline 的每个 Path 创建为一个新的 Ring, 并把 Ring 增加到一个新的 Polygon
For i = 0 To pGeoms_Polyline.GeometryCount - 1
    Set pSegs_Ring = New Ring
    pSegs_Ring.AddSegmentCollection pGeoms_Polyline.Geometry(i)
    PolylineToPolygon.AddGeometry pSegs_Ring
Next i
' 生成的 Polygon 旋转的顺序可能不正确, 为确保正确调用 SimplifyPreserveFromTo
Set pPolygon = PolylineToPolygon
pPolygon.SimplifyPreserveFromTo
Exit Function

ErrorHandler:
    MsgBox Err.Description
End Function

Public Sub CreateNewPolygonFromPolylineGraphic()
    Dim pMxDocument        As IMxDocument
    Dim pEnumFeature        As IEnumFeature
    Dim pFeature0           As IFeature
    Dim pFeatureClass0     As IFeatureClass
    Dim pFeatureLayer0     As IFeatureLayer
    Dim pFeature1          As IFeature
    Dim pFeatureClass1     As IFeatureClass
    Dim pFeatureLayer1     As IFeatureLayer
    Dim pDataSet           As IDataset
    Dim pWorkspaceFactory  As IWorkspaceFactory
    Dim pWorkspaceEdit     As IWorkspaceEdit
    Dim pPolygon            As IPolygon
    Dim pPolyline          As IPolyline
    Dim pMap               As IMap
    Dim pActiveView        As IActiveView

    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    ' 得到当前层
    Set pMap = pMxDocument.FocusMap
    Set pActiveView = pMap
    ' 得到 0, 1 层的 FeatureClass, pFeatureClass0, pFeatureClass1
    Set pFeatureLayer0 = pMxDocument.FocusMap.Layer(0)
    Set pFeatureClass0 = pFeatureLayer0.FeatureClass
    Set pFeatureLayer1 = pMxDocument.FocusMap.Layer(1)
    Set pFeatureClass1 = pFeatureLayer1.FeatureClass
    ' 创建一个编辑工作区
    Set pDataSet = pFeatureClass1

```

```

Set pWorkspaceFactory = New ShapefileWorkspaceFactory
Set pWorkspaceEdit = pWorkspaceFactory.OpenFromFile(pDataSet.Workspace.Path ame, 0)
' 开始编辑
pWorkspaceEdit.StartEditOperation
pWorkspaceEdit.StartEditing True
' 从当前层上得到选择的 Feature
Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
Set pFeature0 = pEnumFeature.Next
' 循环 Feature
While Not pFeature0 Is Nothing
    If pFeature0.ShapeCopy.GeometryType = esriGeometryPolygon Then
        ' Copy 当前层上的一个 Feature 到 Polygon
        Set pPolyline = pFeature0.ShapeCopy
        ' 将 Polyline 创建为 Polygon
        Set pPolygon = PolygonToPolyline(pPolyline)
        ' 将创建的 Polygon, 加到 Polygon 层上, 新建的 Feature 中
        Set pFeature1 = pFeatureClass1.CreateFeature
        Set pFeature1.Shape = pPolygon
        ' 保存 Feature
        pFeature1.Store
    Else
        MsgBox "Must have Polygon in position 0"
        Exit Sub
    End If
    Set pFeature0 = pEnumFeature.Next
Wend
pMxDocument.ActiveView.Refresh
' 停止编辑
pWorkspaceEdit.StopEditOperation
pWorkspaceEdit.StopEditing True
Exit Sub
ErrorHandler:
    pWorkspaceEdit.AbortEditOperation
    MsgBox Err.Description
End Sub

```

1.5.6. 如何从 Polygon 创建 Polyline

本例要实现的功能是根据一个 FeatureLayer 中被选择的一个 Polygon 生成一条 Polyline, 并把该 Polyline 做为一个新的 Feature 保存在一个 Polyline 类型的 FeatureLayer 中。

● 要点

通过所选择 Polygon 创建一个新的 Polyline, 即要根据 Polygon 中的每个 Ring 生成相应的 Path, 程序中用到 ISegmentCollection 接口, 将它实例化为 Path, 利用它的方法 AddSegmentCollection 实现了这一目的。

● 程序说明

程序中添加了两个图层, 第一图层 Polyline 型, 第二图层 Polygon 型。因

为 Polygon 型的图层中不能放 Polyline 型的数据，所以多增加一个 Polyline 层，以便将通过 Polygon 来创建的一个新的 Polyline 显示到上面，使得程序运行结果清晰明了。

函数 PolygonToPolyline(ByRef pPolygon As IPolygon) 中，pSegmentCollectionPath.AddSegmentCollection 创建了一个新 Ring，其中 pSegmentCollectionPath 是一个实例化为 Ring 的 ISegmentCollection 接口变量。

● 代码

```
Private Function PolygonToPolyline(ByRef pPolygon As IPolygon) As IGeometryCollection
    Dim pGeometryCollectionPolygon As IGeometryCollection
    Dim pClone As IClone
    Dim pSegmentCollectionPath As ISegmentCollection
    Dim i As Long

    On Error GoTo ErrorHandler
    ' 创建一个新的 Polyline geometry.
    Set PolygonToPolyline = New Polyline
    ' 克隆即将要操作的 Polygon
    Set pClone = pPolygon
    Set pGeometryCollectionPolygon = pClone.Clone
    ' 把 Polygon 的每个 Ring 创建为一个新的 Path, 并把 Path 增加到一个新的 Polyline
    For i = 0 To pGeometryCollectionPolygon.GeometryCount - 1
        Set pSegmentCollectionPath = New Path
        pSegmentCollectionPath.AddSegmentCollection
    pGeometryCollectionPolygon.Geometry(i)
        PolygonToPolyline.AddGeometry pSegmentCollectionPath
    Next i
    Exit Function

ErrorHandler:
    MsgBox Err.Description
End Function

Public Sub CreateNewPolylineFromPolygonGraphic()
    Dim pMxDocument As IMxDocument
    Dim pEnumFeature As IEnumFeature
    Dim pFeature0 As IFeature
    Dim pFeatureClass0 As IFeatureClass
    Dim pFeatureLayer0 As IFeatureLayer
    Dim pFeature1 As IFeature
    Dim pFeatureClass1 As IFeatureClass
    Dim pFeatureLayer1 As IFeatureLayer
    Dim pDataSet As IDataset
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pWorkspaceEdit As IWorkspaceEdit
    Dim pPolygon As IPolygon
    Dim pPolyline As IPolyline
    Dim pMap As IMap
```



```

Dim pActiveView As IActiveView

On Error GoTo ErrorHandler
Set pActiveView = pMap
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
'得到 0, 1 层的 FeatureClass, pFeatureClass0, pFeatureClass1
Set pFeatureLayer0 = pMxDocument.FocusMap.Layer(0)
Set pFeatureClass0 = pFeatureLayer0.FeatureClass
Set pFeatureLayer1 = pMxDocument.FocusMap.Layer(1)
Set pFeatureClass1 = pFeatureLayer1.FeatureClass
'创建一个编辑工作区
Set pDataSet = pFeatureClass1
Set pWorkspaceFactory = New ShapefileWorkspaceFactory
Set pWorkspaceEdit = pWorkspaceFactory.OpenFromFile(pDataSet.Workspace.Path & ".ame", 0)
'开始编辑
pWorkspaceEdit.StartEditOperation
pWorkspaceEdit.StartEditing True
'从当前层上得到选择的 Feature
Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
Set pFeature0 = pEnumFeature.Next
'循环 Feature
While Not pFeature0 Is Nothing
    If pFeature0.ShapeCopy.GeometryType = esriGeometryPolygon Then
        'Copy 当前层上的一个 Feature 到 Polygon
        Set pPolygon = pFeature0.ShapeCopy
        '将 Polygon 创建为 Polyline
        Set pPolyline = PolygonToPolyline(pPolygon)
        '将创建的 Polyline, 加到 Polyline 层上, 新建的 Feature 中
        Set pFeature1 = pFeatureClass1.CreateFeature
        Set pFeature1.Shape = pPolyline
        '保存 Feature
        pFeature1.Store
    Else
        MsgBox "Must have Polygon in position 0"
        Exit Sub
    End If
    Set pFeature0 = pEnumFeature.Next
Wend
pMxDocument.ActiveView.Refresh
'停止编辑
pWorkspaceEdit.StopEditOperation
pWorkspaceEdit.StopEditing True
Exit Sub

ErrorHandler:
    pWorkspaceEdit.AbortEditOperation
    MsgBox Err.Description
End Sub

```

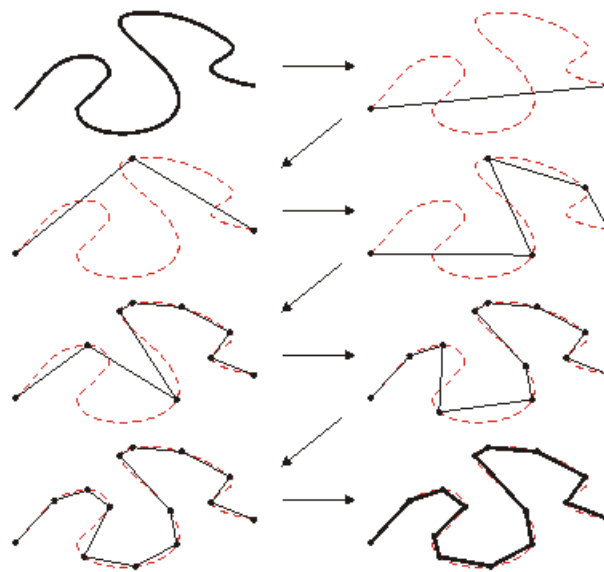
1.5.7. 如何将 Polygon/PolyCurve 一般化(Generalize)

本例要实现的功能是使用道格拉斯-普克(Douglas-Poiker)算法对 Polyline

或 Polygon 做抽稀运算。

●要点

所谓道格拉斯-普克抽稀算法，是用来对大量冗余的图形数据点进行压缩以提取必要的数据点。该算法实现抽稀的过程是：先将一条曲线首尾点虚连一条直线，求其余各点到该直线的距离，取其最大者与规定的临界值相比较，若小于临界值，则将直线两端间各点全部舍去，否则将离该直线距离最大的点保留，并将原线条分成两部分，对每部分线条再实施该抽稀过程，直到结束。抽稀结果点数随选取限差临界值的增大而减少，应用时应根据精度来选取限差临界值，以获得最好的效果。下面是该抽稀算法的过程示意图。



本例中使用接口 IPolycurve2 的方法 `Generalize (maxAllowableOffset)` 来实现 Polygon 或 Polyline 的抽稀运算，其中的参数 `maxAllowableOffset`，就是拉斯-普克抽稀算法中的限差临界值。

●程序说明

在本例的实现中，从一个 Element 的 Geometry 属性得到一条曲线，并将其抽稀。限差临界值设为该线条长度的三十分之一。

●代码

```
Public Sub Generalize()  
    Dim pMxDocument As IMxDocument  
    Dim pGraphicsContainerSelect As IGraphicsContainerSelect
```

```

Dim pGeometry As IGeometry
Dim pElement As IElement
Dim pPolycurve As IPolycurve2

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
'在一般化 Polyline 或 Polygon 之前，首先要找到将要一般化的 Geometry
Set pGraphicsContainerSelect = pMxDocument.ActiveView
If pGraphicsContainerSelect.ElementSelectionCount = 1 Then
    '得到当前选择的 Element
    Set pElement = pGraphicsContainerSelect.SelectedElement(0)
    Set pGeometry = pElement.Geometry
    '一般化 Polygon 或 Polyline
    If (pGeometry.GeometryType = esriGeometryPolygon) Or _
        (pGeometry.GeometryType = esriGeometryPolyline) Then
        Set pPolycurve = pGeometry
        pPolycurve.Generalize pPolycurve.Length / 30
        pElement.Geometry = pGeometry
    End If
    pMxDocument.ActiveView.PartialRefresh esriViewGraphics, Nothing, Nothing
End If
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1.5.8. 如何获得 Polygon 的中点

本例要实现的功能是根据一个 Polygon 类型的 FeatureLayer 中被选择的 Polygon, 得到它们的中心。并根据它们的中心点在 Point 型的 FeatureLayer 中新建 Feature。

●要点

本例使用接口 IArea 的方法 QueryCentroid (*Center*), 来得到一个 Polygon 的中心点。

●程序说明

程序中加了两个层, 第一个层是 Point 类型的, 第二个层是 Polygon 类型的。在选择了一个或多个 Polygon 之后, 运行本例程序, 就会在 Point 层上新生成这些 Polygon 的中心点。

●代码

```

Public Sub ConvertPointToPolygon()
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pEnumFeature As IEnumFeature
    Dim pActiveView As IActiveView

```

```

Dim pFeature          As IFeature
Dim pPoint            As IPoint
Dim pArea             As IArea
Dim pFeatureClass     As IFeatureClass
Dim pFeatureLayer     As IFeatureLayer
Dim pFeature1         As IFeature
Dim pFeatureClass1    As IFeatureClass
Dim pFeatureLayer1    As IFeatureLayer
Dim pDataSet          As IDataset
Dim pWorkspaceFactory As IWorkspaceFactory
Dim pWorkspaceEdit    As IWorkspaceEdit

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pActiveView = pMap
Set pFeatureLayer = pMap.Layer(0)
Set pFeatureClass = pFeatureLayer.FeatureClass
Set pFeatureLayer1 = pMap.Layer(1)
Set pFeatureClass1 = pFeatureLayer1.FeatureClass
' 创建一个工作区, 开始编辑
Set pDataSet = pFeatureClass1
Set pWorkspaceFactory = New ShapefileWorkspaceFactory
Set pWorkspaceEdit = pWorkspaceFactory.OpenFromFile(pDataSet.Workspace.Path & "\ame", 0)
pWorkspaceEdit.StartEditOperation
pWorkspaceEdit.StartEditing True

' 将变量进行实例化
Set pPoint = New Point
' 得到所选择的图形集
Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
Set pFeature = pEnumFeature.Next

If pFeature Is Nothing Then
    MsgBox "Select Polygon"
    Exit Sub
End If
' 得到 Polygon 的中心
While Not pFeature Is Nothing
    If pFeature.ShapeCopy.GeometryType = esriGeometryPolygon Then
        Set pArea = pFeature.Shape
        ' 得到 Polygon 的中心 pPoint
        pArea.QueryCentroid pPoint
        Set pFeature1 = pFeatureClass1.CreateFeature
        ' 中心储存在第二层
        Set pFeature1.Shape = pPoint
        pFeature1.Store
    End If
    Set pFeature = pEnumFeature.Next
Wend

pMxDocument.ActiveView.Refresh
' 停止编辑
pWorkspaceEdit.StopEditOperation
pWorkspaceEdit.StopEditing True

```

```
Exit Sub

ErrorHandler:
    pWorkspaceEdit.AbortEditOperation
    MsgBox Err.Description
End Sub
```

1.5.9. 如何判断图形间的逻辑运算

本小节以 Polyline (Polygon 类似) 为例, 讲解如何判断图形间的逻辑关系, 主要用到的接口是 IRelationalOperator。

● 要点

在本例中, 使用 Relational Operator 对两个图形进行比较, 返回一个布尔值来指出这两个图形间是否存在特定的关系。一些关系的判断是要求两个图形要有相同的维数的 (如必须 Polyline 之间或 Polygon 之间), 而另外一些对图形维数就没有太多限制。大多数已定义的关系操作符是互斥的。RelationalOperator 的具体方法有:

Contains: 判断一个图形是否包含另外一个图形。

Within: 判断一个图形是否被另外一个图形所包含。

Crosses: 判断两个图形是否在维数较少的那个图形的内部相交。

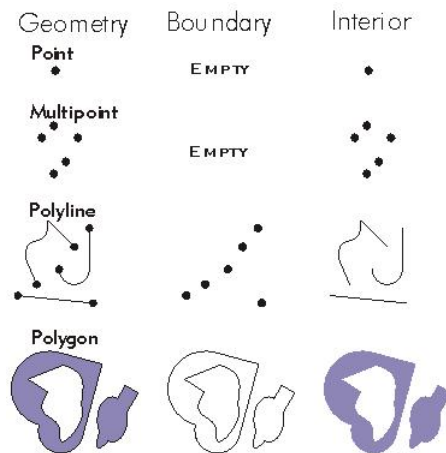
Disjoint: 判断两个图形间是否没有相同点。

Equals: 判断两个图形是否是同一个类型并且在平面上的点是否是相同的位置。如果返回值为真, 则它们应该包含 (Contains) 另外一个图形同时也被另外一个图形所包含 (Within)。

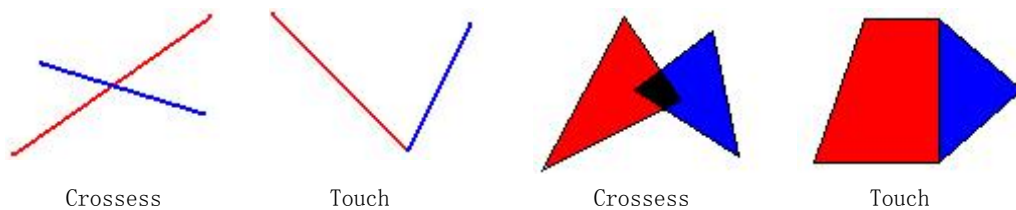
Overlaps: 判断两个图形的交集是否和其中的一个图形拥有相同的维数, 并且他们交集不能和其中任何一个图形相等。该方法只使用与两个 Polyline 之间或者两个 Polygon 之间。

Touch: 判断两个图形的边界是否相交, 如果两个图形的交集不为空, 但两个图形内部的交集为空, 则返回值为真。

下图为几个图形的边界 (Boundary) 和内部 (Interior) 概念的图解:



下面针对较易混淆的两个概念，Crosses 和 Touch 进行举例说明 (Polyline/Polyline):



●程序说明

首先创建两条 Polyline:pLine1,pLine2, 然后对这两条 Polyline 进行各种逻辑运算。针对不同的逻辑运算创建不同位置的 Polyline 以进行验证。

●代码

```
Public Sub RelationalOperatorsDemo()
    Dim pLine1 As ILine
    Dim pLine2 As ILine
    Dim pBaseSegmentC As ISegmentCollection
    Dim pCompSegmentC As ISegmentCollection
    Dim pRelationalOperator As IRelationalOperator
    Dim pBaseGeometry As IGeometry
    Dim pCompGeometry As IGeometry
    Dim pPoints(0 To 1) As Ipoint

    On Error GoTo ErrorHandler
    Set pLine1 = New Line
    Set pLine2 = New Line
    Set pBaseSegmentC = New Polyline
```

```

Set pCompSegmentC = New Polyline
Set pRelationalOperator = pBaseSegmentC
Set pBaseGeometry = pBaseSegmentC
Set pCompGeometry = pCompSegmentC
Set pPoints(0) = New Point
Set pPoints(1) = New Point

pPoints(0).PutCoords 0, 0
pPoints(1).PutCoords 100, 100
pLine1.PutCoords pPoints(0), pPoints(1)

'Example of Contains
pPoints(0).PutCoords 20, 20
pPoints(1).PutCoords 80, 80
pLine2.PutCoords pPoints(0), pPoints(1)
' (0,0) to (100,100)
pBaseSegmentC.AddSegment pLine1
' (20,20) to (80,80)
pCompSegmentC.AddSegment pLine2
MsgBox "Contains: " & pRelationalOperator.Contains(pCompSegmentC)
' Set polylines empty so they can be reused
pBaseGeometry.SetEmpty
pCompGeometry.SetEmpty

'Example of Within (Within is the complement of Contains)
' (0,0) to (100,100)
pCompSegmentC.AddSegment pLine1
' (20,20) to (80,80)
pBaseSegmentC.AddSegment pLine2
MsgBox "Within: " & pRelationalOperator.Within(pCompSegmentC)
pBaseGeometry.SetEmpty
pCompGeometry.SetEmpty

'Example of Equals
pPoints(0).PutCoords 0, 0
pPoints(1).PutCoords 100, 100
pLine2.PutCoords pPoints(0), pPoints(1)
' (0,0) to (100,100)
pBaseSegmentC.AddSegment pLine1
' (0,0) to (100,100)
pCompSegmentC.AddSegment pLine2
' If Equals is True, then Within and Contains should also be true
If pRelationalOperator.Equals(pCompSegmentC) Then
    MsgBox "Equals is true"
    MsgBox "Within: " & pRelationalOperator.Within(pCompSegmentC) & vbCrLf
    "Contains: " & pRelationalOperator.Contains(pCompSegmentC)
End If
pBaseGeometry.SetEmpty
pCompGeometry.SetEmpty
'Example of Disjoint
pPoints(0).PutCoords -40, 40
pPoints(1).PutCoords -100, 80
pLine2.PutCoords pPoints(0), pPoints(1)

```

```

' (0,0) to (100,100)
pBaseSegmentC.AddSegment pLine1
' (-40,40) to (-100,80)
pCompSegmentC.AddSegment pLine2
MsgBox "Disjoint: " & pRelationalOperator.Disjoint(pCompSegmentC)
pBaseGeometry.SetEmpty
pCompGeometry.SetEmpty

' Example of Touches (Objects touch on their boundaries)
pPoints(0).PutCoords 100, 100
pPoints(1).PutCoords 200, 150
pLine2.PutCoords pPoints(0), pPoints(1)
' (0,0) to (100,100)
pBaseSegmentC.AddSegment pLine1
' (100,100) to (200,150)
pCompSegmentC.AddSegment pLine2
MsgBox "Touches: " & pRelationalOperator.Touches(pCompSegmentC)
pBaseGeometry.SetEmpty
pCompGeometry.SetEmpty

' Example of Overlaps
pPoints(0).PutCoords 140, 140
pPoints(1).PutCoords 60, 60
pLine2.PutCoords pPoints(0), pPoints(1)
' (0,0) to (100,100)
pBaseSegmentC.AddSegment pLine1
' (140,140) to (60,60)
pCompSegmentC.AddSegment pLine2

MsgBox "Overlaps: " & pRelationalOperator.Overlaps(pCompSegmentC)
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1. 5. 10. 如何进行图形间的逻辑运算

本例主要运用 ITopologicalOperator 接口来实现一个或多个图形间的逻辑运算。

● 要点

ITopologicalOperator 接口主要用来对已存在的一个或多个图形进行拓扑逻辑关系运算，从而生成一个新的图形。该接口主要有以下几种方法：

Buffer: 由距离某个图形特定长度内的所有点的轨迹的集合构建的一个多边形区域。

Clip: 得到一个图形和一个矩形线框相交部分区域。

ConvexHull:得到一个图形的凸包, 即该图形的最小外接多边形。

Difference:构建一个图形, 它包含了这个图形中的所有点但不包含另外一个图形中的任何点。

Intersect:得到两个图形的交集, 即两个具有相同维数的图形的重叠区域。

Union:得到两个相同维数的图形的合集, 两个图形的相同部分在该图形中只存在一个。

● 程序说明

本例以 ITopologicalOperator 接口中的 Buffer 方法为例, 对地图中选中的一个或多个

Polygon 使用 Buffer 方法生成新多边形区域。当地图放大或缩小时, 该多变形区域会跟着放大或缩小。

● 代码

```
' 对模块中的全局变量进行声明。
' 对 Map 定义一个名为 ActiveViewEvents 的事件
Private WithEvents ActiveViewEvents As Map

Private m_pMxDocument          As IMxDocument
Private m_pBufferPolygon        As IPolygon
Private m_pBufferEnvelope       As IEnvelope
Private m_pSimpleFillS          As ISimpleFillSymbol

' 事件 ActiveViewEvent 执行前的初始化, 设置多边形的填充属性。
Public Sub InitEvents()
    Dim pViewManager             As IViewManager
    Dim pRgbColor                 As IRgbColor

    On Error GoTo ErrorHandler:

    Set m_pMxDocument = ThisDocument
    Set pViewManager = m_pMxDocument.FocusMap
    pViewManager.VerboseEvents = True
    Set ActiveViewEvents = m_pMxDocument.FocusMap

    ' Create a fill symbol
    Set m_pSimpleFillS = New SimpleFillSymbol
    Set pRgbColor = New RgbColor
    pRgbColor.Blue = 255
    ' 设置填充符的类型和颜色
    m_pSimpleFillS.Style = esriSFSForwardDiagonal
    m_pSimpleFillS.Color = pRgbColor

    Exit sub
ErrorHandler:
```

```

        MsgBox Err.Description
    End Sub

' 事件 ActiveViewEvents 在画了 Item 后要执行的过程，本例中当一个或多个
' 多边形进行 Buffer 处理后，重画这些多边形。
Private Sub ActiveViewEvents_AfterItemDraw(ByVal Index As Integer, ByVal display As
IDisplay, ByVal phase As esriDrawPhase)

On Error GoTo ErrorHandler:
    'Only draw in the geography phase
    If Not phase = esriDPGeography Then Exit Sub
    'Draw the buffered polygon
    If m_pBufferPolygon Is Nothing Then Exit Sub
    '将逻辑运算后的图形具体显示出来
    With display
        .SetSymbol m_pSimpleFillS
        .DrawPolygon m_pBufferPolygon
    End With

    Exit sub
ErrorHandler:
    MsgBox Err.Description
End Sub

' 事件 ActiveViewEvents 在改变所选的图形后要执行的过程。本例中，对所
' 选择的图形做 Buffer 运算。
Private Sub ActiveViewEvents_SelectionChanged()

    Dim pActiveView As IActiveView
    Dim pEnumFeature As IEnumFeature
    Dim pFeature As IFeature
    Dim pPolygon As IPolygon
    Dim pTopologicalOperator As ITopologicalOperator
    Dim pGeometryCollection As IGeometryCollection

On Error GoTo ErrorHandler:

    Set pActiveView = m_pMxDocument.FocusMap
    Set pGeometryCollection = New GeometryBag

    'Flag last buffered region for invalidation
    If Not m_pBufferEnvelope Is Nothing Then
        pActiveView.PartialRefresh esriViewGeography, Nothing, m_pBufferEnvelope
    End If

    If m_pMxDocument.FocusMap.SelectionCount = 0 Then
        'Nothing selected; don't draw anything
        Set m_pBufferPolygon = Nothing
        Exit Sub
    End If

    'Buffer each selected feature

```

```

' 将用户选中的 Polygon 都放入 pEnumFeature 中
Set pEnumFeature = m_pMxDocument.FocusMap.FeatureSelection
pEnumFeature.Reset
' 将 pEnumFeature 中的第一个 Polygon 赋给 pFeature
Set pFeature = pEnumFeature.Next
' 将 pEnumFeature 中所有的 Polygon 进行 Buffer 逻辑运算，并且将运算后的 Buffers 加入到
pGeometryCollection 的队列中
Do While Not pFeature Is Nothing
    Set pTopologicalOperator = pFeature.Shape
    Set pPolygon = pTopologicalOperator.Buffer(0.5)
    pGeometryCollection.AddGeometry pPolygon
    ' Get next feature
    Set pFeature = pEnumFeature.Next
Loop
' 通过 Union 运算将所有的 Buffers 连接成一个 Polygon
Set m_pBufferPolygon = New Polygon
Set pTopologicalOperator = m_pBufferPolygon
pTopologicalOperator.ConstructUnion pGeometryCollection

Set m_pBufferEnvelope = m_pBufferPolygon.Envelope
' Flag new buffered region for invalidation
pActiveView.PartialRefresh esriViewGeography, Nothing, m_pBufferPolygon.Envelope

Exit sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.5.11. 如何创建 Envelope 的 Boundary

本例主要实现对一个或多个 Polygon 的 Envelope 创建 Boundary。

● 要点

在本例的实现过程中，首先运用 ITopologicalOperator 接口对多个 Polygon 进行 Union 操作，将其合并为单个 Polygon。但由于 Envelope 类未实现 ITopologicalOperator 接口，从而不能直接返回 Boundary 这个属性，因此需要创建一个函数来得到这个 Envelope 的 Boundary。

● 程序说明

ActiveViewEvents_SelectionChanged() 过程是将用户选中的多个 Polygon 进行 Union 逻辑运算得到一个新的 Polygon，然后求出这个 Polygon 的 Envelope。

CreateBoundaryOfEnvelope() 过程是求出 Envelope 的 Boundary，并运用 CreateLine 这个函数将 Boundary 画出。

ActiveViewEvents_AfterDraw() 是将 Boundary 在屏幕上具体显示出来。

使用该程序时，先运行宏的 InitEvents() 过程，然后在地图上选择若干个

Polygon 即可得到这若干个 Polygon 的 Envelope 的 Boundary。

● 代码

```
Option Explicit
Private WithEvents ActiveViewEvents As Map
Private m_pMxDocument As IMxDocument
Private m_pPolygon As IPolygon
Private m_pEnvelope As IEnvelope
Private m_pPolycurve As IPolycurve
Private m_pGeometryCollection As IGeometryCollection
Private m_pSegmentCollection As ISegmentCollection

Public Sub InitEvents()
    Set m_pMxDocument = ThisDocument
    Set ActiveViewEvents = m_pMxDocument.FocusMap
End Sub

Private Sub CreateBoundaryOfEnvelope(ByVal pEnvelope As IEnvelope, ByRef pPolycurve As IPolycurve)
    On Error GoTo ErrorHandler

    'Check we have valid parameters.
    'pPolycurve must be initialized as either a Polygon or Polyline, and must be empty.
    If pEnvelope Is Nothing Or pPolycurve Is Nothing Then Exit Sub
    pPolycurve.SetEmpty

    Dim pLine As ILine
    'Build the boundary of the Envelope, pEnvelope.
    '根据 pPolycurve 的类型来设置 m_pSegmentCollection
    If TypeOf pPolycurve Is IPolygon Then
        Set m_pSegmentCollection = New Ring
    ElseIf TypeOf pPolycurve Is IPolyline Then
        Set m_pSegmentCollection = New Path
    Else
        Exit Sub
    End If
    '画出 Boundary, 并将其存储在 m_pSegmentCollection 中
    Set pLine = CreateLine(pEnvelope.UpperLeft, pEnvelope.UpperRight)
    m_pSegmentCollection.AddSegment pLine
    Set pLine = CreateLine(pEnvelope.UpperRight, pEnvelope.LowerRight)
    m_pSegmentCollection.AddSegment pLine
    Set pLine = CreateLine(pEnvelope.LowerRight, pEnvelope.LowerLeft)
    m_pSegmentCollection.AddSegment pLine
    Set pLine = CreateLine(pEnvelope.LowerLeft, pEnvelope.UpperLeft)
    m_pSegmentCollection.AddSegment pLine

    Set m_pGeometryCollection = pPolycurve
    m_pGeometryCollection.AddGeometry m_pSegmentCollection

    Exit Sub
ErrorHandler:

```

```

        MsgBox Err.Description

End Sub

Private Function CreateLine(ByVal pFrom As IPoint, ByVal pTo As IPoint) As ILi e

    'This function creates a new Line object with the passed in From and To po nts.
    Set CreateLine = New esriCore.Line
    CreateLine.PutCoords pFrom, pTo

End Function

Private Sub ActiveViewEvents_AfterDraw(ByVal Display As IDisplay, ByVal p ase As
esriViewDrawPhase)
    Dim i                As Integer
    Dim pSimpleLineS     As ISimpleLineSymbol
    Dim pRgbColor        As IRgbColor
    Dim pScreenDisplay   As IScreenDisplay

    On Error GoTo ErrorHandler
    If m_pGeometryCollection Is Nothing Then Exit Sub
    If m_pGeometryCollection.GeometryCount = 0 Then Exit Sub
    ' 设置填充类型为 Line, 并将颜色设置为红色
    Set pSimpleLineS = New SimpleLineSymbol
    Set pRgbColor = New RgbColor
    pRgbColor.Blue = 0
    pRgbColor.Green = 0
    pRgbColor.Red = 255
    pSimpleLineS.Color = pRgbColor

    If TypeOf Display Is IScreenDisplay Then
        Set pScreenDisplay = Display
        ' 画出 Bondery
        With pScreenDisplay
            .StartDrawing pScreenDisplay.hDC, esriNoScreenCache
            .SetSymbol pSimpleLineS
            .DrawPolyline m_pGeometryCollection
            .FinishDrawing
        End With
    End If

    Exit Sub

ErrorHandler:
    MsgBox Err.Description

End Sub

Private Sub ActiveViewEvents_SelectionChanged()
    Dim pActiveview      As IActiveView
    Dim pEnumFeature     As IEnumFeature
    Dim pFeature         As IFeature

```

```

Dim pPolygon As IPolygon
Dim pGeometryCollection As IGeometryCollection
Dim pTopologicalOperator As ITopologicalOperator
Set pActiveview = m_pMxDocument.FocusMap
Set pGeometryCollection = New GeometryBag

On Error GoTo ErrorHandler

' 如果 Envelope 不为空，则刷新屏幕
If Not m_pEnvelope Is Nothing Then
    pActiveview.PartialRefresh esriViewGeography, Nothing, m_pEnvelope
End If

If m_pMxDocument.FocusMap.SelectionCount = 0 Then
    ' Nothing selected; don't draw anything
    Set m_pPolygon = Nothing
    Exit Sub
End If

Set pEnumFeature = m_pMxDocument.FocusMap.FeatureSelection
pEnumFeature.Reset

Set pFeature = pEnumFeature.Next
' 将所选的 Polygon 加入到 pGeometryCollection 中
Do While Not pFeature Is Nothing
    Set pPolygon = pFeature.Shape
    pGeometryCollection.AddGeometry pPolygon
    ' Get next feature
    Set pFeature = pEnumFeature.Next
Loop
' 合并所有选定的 Polygon
Set m_pPolygon = New Polygon
Set pTopologicalOperator = m_pPolygon
pTopologicalOperator.ConstructUnion pGeometryCollection
Set m_pEnvelope = m_pPolygon.Envelope
' 初始化 m_pPolycurve
Set m_pPolycurve = New Polygon
' 调用 CreateBoundaryOfEnvelope 函数作出 Boundary
CreateBoundaryOfEnvelope m_pEnvelope, m_pPolycurve

Exit Sub

ErrorHandler:
MsgBox Err.Description

End Sub

```

1. 5. 12. 如何通过鼠标移动图形

本例主要通过引用 IMoveLineFeedback 和 IMovePolygonFeedback 两个接口实现对地图中的 Polyline 和 Polygon 的移动。

●要点

对于类型为 Polygon 的 Feature, 本例使用接口 IMovePolygonFeedback 的 Start 和 Stop 方法来移动选定的 Feature。其他类型的 Feature 类似, 只需相应地改变接口类型即可。

●程序说明

在工具条上设置一个 ToolButton, 通过响应该 Button 的MouseDown(),MouseMove(),MouseUp()事件来实现对图形的移动。本例只列举了 Polyline 和 Polygon 两种类型的图形, 其他类型的图形可类似操作。

●代码

```
Dim m_pFeature As IFeature
Dim m_pMxDocument As IMxDocument
Dim m_pDisplayFeedback As IDisplayFeedback
Dim m_pScreenDisplay As IScreenDisplay
Dim m_pMouseCursor As IMouseCursor

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long _
    ByVal x As Long, ByVal y As Long)

    Dim pGeometry As IGeometry
    Dim pSelectionSet As ISelectionSet
    Dim pFeatureLayer As IFeatureLayer
    Dim pFeatureCursor As IFeatureCursor
    Dim pSpatialFilter As ISpatialFilter
    Dim pPoint As IPoint
    Dim pEnvelope As IEnvelope
    Dim index As Integer

    On Error GoTo ErrorHandler
    Set m_pMxDocument = ThisDocument
    Set m_pScreenDisplay = m_pMxDocument.ActiveView.ScreenDisplay
    If m_pMxDocument.FocusMap.LayerCount = 0 Then Exit Sub
    '得到鼠标点击的起点坐标
    Set pPoint = m_pScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
    '得到当前鼠标位置的 Envelope
    Set pGeometry = m_pMxDocument.CurrentLocation.Envelope
    Set pSpatialFilter = New SpatialFilter
    Set pEnvelope = pGeometry.Envelope
    '扩大 Envelope 的范围便于搜索
    pEnvelope.Expand 0.2, 0.2, False
    '设置空间检索的条件
    With pSpatialFilter
        Set .Geometry = pEnvelope
        .GeometryField = "SHAPE"
        .SpatialRel = esriSpatialRelIntersects
    End With
    '在当前 Map 的所有 FeatureLayer 中查找所要移动的图形
```

```

Dim i As Integer
i = 0
index = m_pMxDocument.FocusMap.LayerCount
Do While i < index
    Set pFeatureLayer = m_pMxDocument.FocusMap.Layer(i)
    Set pFeatureCursor = pFeatureLayer.FeatureClass.Search(pSpatialFilter, True)
    Set m_pFeature = pFeatureCursor.NextFeature
    If Not m_pFeature Is Nothing Then
        Exit Do
    Else
        i = i + 1
    End If
Loop
If m_pFeature Is Nothing Then Exit Sub
' 针对不同的 Feature 类型调用不同的接口进行操作
Select Case m_pFeature.Shape.GeometryType
    ' 若 Feature 类型为 Polyline
    Case 3:
        Set m_pDisplayFeedback = New MoveLineFeedback
        Set m_pDisplayFeedback.Display = m_pScreenDisplay

        Dim pMoveLineF As IMoveLineFeedback
        Set pMoveLineF = m_pDisplayFeedback

        pMoveLineF.Start m_pFeature.Shape, pPoint
        ' 若 Feature 类型为 Polygon
    Case 4:
        Set m_pDisplayFeedback = New MovePolygonFeedback
        Set m_pDisplayFeedback.Display = m_pScreenDisplay

        Dim pMovePolygonF As IMovePolygonFeedback
        Set pMovePolygonF = m_pDisplayFeedback

        pMovePolygonF.Start m_pFeature.Shape, pPoint
        ' 若为其他类型，本例则省略，不进行操作
    Case Else
        MsgBox "Other SHP Type"
        Set m_pFeature = Nothing
        Set m_pScreenDisplay = Nothing
        Exit Sub
End Select
m_pMxDocument.ActiveView.Refresh
' 设置鼠标外观
Set m_pMouseCursor = New MouseCursor
m_pMouseCursor.SetCursor 5
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControl1_MouseMove(ByVal button As Long, ByVal shift As Long, _
    ByVal x As Long, ByVal y As Long)

```



```

On Error GoTo ErrorHandler
If Not m_pDisplayFeedback Is Nothing Then
    Dim pPoint As IPoint
    ' 得到鼠标点击位置在地图上的坐标
    Set pPoint = m_pScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
    m_pDisplayFeedback.MoveTo pPoint
End If
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControll_MouseUp(ByVal button As Long, ByVal shift As Long, _
    ByVal x As Long, ByVal y As Long)

    Dim pGeometry As IGeometry

    On Error GoTo ErrorHandler
    ' 检查是否存在一个元素
    If Not m_pFeature Is Nothing Then
        ' 检测此元素的类型
        Select Case m_pFeature.Shape.GeometryType
            Case 3:
                Dim pMoveLineF As IMoveLineFeedback
                Set pMoveLineF = m_pDisplayFeedback
                Set pGeometry = pMoveLineF.Stop
            Case 4:
                Dim pMovePolygonF As IMovePolygonFeedback
                Set pMovePolygonF = m_pDisplayFeedback
                Set pGeometry = pMovePolygonF.Stop
        End Select

        ' 更新元素
        Set m_pFeature.Shape = pGeometry
        m_pFeature.Store

        Set m_pDisplayFeedback = Nothing
        m_pMxDocument.ActiveView.Refresh
        ' 将鼠标外观还原
        Set m_pMouseCursor = New MouseCursor
        m_pMouseCursor.SetCursor 0
    End If
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1. 5. 13. 如何为一个图形添加一个顶点

本例要实现的功能是为一个FeatureLayer 中被选择的一个Feature 的Shape

增加一个顶点。实现的方法中用到了接口 `IPointCollection` 的方法 `AddPoint`，和接口 `IHitTest` 的方法 `HitTest`。

● 要点

为图形增加顶点，用到接口 `IPointCollection` 中的方法 `AddPoint` (`inPoint`, [, `before`] [, `after`]) 如果 `before` 和 `after` 默认，则点加到点集合的最后；如果指定了 `before` 或 `after`，则点加到指定的位置。

接口 `IHitTest` 的方法：`HitTest(QueryPoint, searchRadius, geometryPart, hitPoint, hitDistance, hitPartIndex, hitSegmentIndex, RightSide)`，其中 `QueryPoint` 是被用来查询的点，`hitPoint` 返回被点击图形中离查询点最近的一个点。使用该方法来判断要添加的点是否在图形设定的偏差范围内，是就在该位置为图形添加一个顶点。

● 程序说明

本例是为一个 `Polygon` 添加一个顶点，如果该 `Polygon` 只有一个 `Ring`，则就在该 `Ring` 上添加顶点，如果该 `Polygon` 多于一个 `Ring`，则在 `Polygon` 的第一个 `Ring` 上添加顶点。函数 `AddVertex(pPolygon As IPolygon, pNewPoint As IPoint, LAfterIndex As Long)`，将点 `pNewPoint` 插入到 `pPolygon` 中，在索引为 `LAfterIndex` 的顶点之后。

● 代码

```
Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x
                                     As Long, _
                                     ByVal y As Long)
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pActiveView As IActiveView
    Dim pFeatureLayer As IFeatureLayer
    Dim pFeatureClass As IFeatureClass
    Dim pEnumFeature As IEnumFeature
    Dim pFeature As IFeature
    Dim pHitTest As IHitTest
    Dim pHitPoint As IPoint
    Dim DhitDist As Double
    Dim LpartIndex As Long
    Dim LvertexIndex As Long
    Dim BvertexHit As Boolean
    Dim pPoint As IPoint
    Dim pPolygon As IPolygon

    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
```

```

Set pActiveView = pMap
' 将鼠标点下的点坐标转换为地图坐标
Set pPoint = pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
' 得到第一层
Set pFeatureLayer = pMap.Layer(0)
' 得到第一层的 FeatureClass
Set pFeatureClass = pFeatureLayer.FeatureClass
' 得到第一层所选择的 EnumFeature

Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
pEnumFeature.Reset
Set pFeature = pEnumFeature.Next
' 判断是否有 Feature 被选中
If pFeature Is Nothing Then
    MsgBox "no selection, please select a polygon"
    Exit Sub
Else
    Set pHitPoint = New Point
    Set pHitTest = pFeature.Shape
    ' 判断增加的点是否在 Polygon 所能容忍的边界上
    ' 在偏差值为 4 的情况下, 返回 True, 并返回增加点的前面 Polygon 的一个索引号
    ' LvertexIndex
    If pHitTest.HitTest(pPoint, 4, esriGeometryPartBoundary, pHitPoint, Dh:Dist,
        LpartIndex, LvertexIndex, False) Then
        Set pPolygon = pFeature.Shape
        ' 为 Polygon 增加一个顶点
        Call AddVertex(pPolygon, pPoint, LvertexIndex + 1)
        Set pFeature.Shape = pPolygon
        pFeature.Store
        Set pFeature = Nothing
        pMxDocument.FocusMap.ClearSelection
        pMxDocument.ActiveView.Refresh
    Else
        ' 增加的点不在偏差值范围内, 信息框提示重新选点
        MsgBox "增加的点不在容忍范围内, 请冲重选"
        Exit Sub
    End If
End If
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Public Sub AddVertex(pPolygon As IPolygon, pNewPoint As IPoint, LAfterIndex As Long)
    Dim pPointCollection As IPointCollection
    Dim pGeoCollection As IGeometryCollection
    Dim pRing As IRing

    On Error GoTo ErrorHandler
    If pPolygon Is Nothing Then Exit Sub
    Set pGeoCollection = pPolygon
    ' 判断 Polygon 是否是一个 Part
    If pGeoCollection.GeometryCount = 1 Then
        ' 是一个 Part 就把 pPolygon 赋给 pPointCollection

```

```

        Set pPointCollection = pPolygon
    Else
        ' 如果有多个 Part, 就把 Feature 的第一个 Part 赋给 pPointCollection
        If pGeoCollection.Geometry(0).GeometryType = esriGeometryRing Then
            Set pRing = pGeoCollection.Geometry(0)
            Set pPointCollection = pRing
        End If
    End If
    ' 给 Polygon 增加一个顶点
    pPointCollection.AddPoint pNewPoint, LafterIndex
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.5.14. 如何删除一个图形上的一个顶点

本例要实现的功能是为一个 FeatureLayer 中被选择的一个 Feature 的 Shape 删除一个顶点。实现的方法中用到了接口 IPointCollection 的方法 RemovePoint, 和接口 IHitTest 的方法 HitTest。

● 要点

删除图形的顶点，用到接口 IPointCollection 中的方法 RemovePoints (*Index*, *Count*)：从指定的位置起移除指定个数的点。

接口 IHitTest 的方法:HitTest(QueryPoint, earchRadius, eometryPart, itPoint, itDistance, itPartIndex, hitSegmentIndex, bRightSide), 用来判断要删除的点是否在图形设定的偏差范围内，如果是就将在该位置的顶点删除。其中 QueryPoint 是被用来查询的点，hitPoint 返回被点击图形中离查询点最近的一个点。

● 程序说明

本例是为一个 Polygon 删除一个顶点，如果该 Polygon 只有一个 Ring，则就在该 Ring 上删除顶点；如果该 Polygon 多于一个 Ring，则删除 Polygon 的第一个 Ring 上的顶点。函数 RemoveVertex(pPolygon As IPolygon, lRemovePointIndex As Long)，是将索引为 lRemovePointIndex 的顶点从 pPolygon 中删除。

● 代码

```

Private Sub UIToolControll1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x
As Long, _
                                     ByVal y As Long)
    Dim pMxDocument As IMxDocument

```

```

Dim pMap As IMap
Dim pActiveView As IActiveView
Dim pFeatureLayer As IFeatureLayer
Dim pFeatureClass As IFeatureClass
Dim pEnumFeature As IEnumFeature
Dim pFeature As IFeature
Dim pHitTest As IHitTest
Dim pHitPoint As IPoint
Dim DhitDist As Double
Dim LpartIndex As Long
Dim LvertexIndex As Long
Dim BvertexHit As Boolean
Dim pPoint As IPoint
Dim pPolygon As IPolygon
Dim pPointCollection As IPointCollection

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pActiveView = pMap
'将鼠标点下的点坐标转换为地图坐标
Set pPoint = pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
'得到第一层
Set pFeatureLayer = pMap.Layer(0)
'得到第一层的 FeatureClass
Set pFeatureClass = pFeatureLayer.FeatureClass
'得到第一层所选择的特征
Set pEnumFeature = pMxDocument.FocusMap.FeatureSelection
pEnumFeature.Reset
Set pFeature = pEnumFeature.Next
If pFeature Is Nothing Then
    MsgBox "no selection, please select a polygon"
    Exit Sub
End If
Set pPointCollection = pFeature.Shape
If pPointCollection.PointCount <= 4 Then
    MsgBox "Polygon 的顶点数小于 4 时, 不能再删"
    Exit Sub
End If
'判断是否有 Featrue 被选中
Set pHitPoint = New Point
Set pHitTest = pFeature.Shape
'判断增加点是否在 Polygon 的所能容忍的边界上
'如果点在偏差范围为 4 的情况下, 返回 True, 返回 LvertexIndex 的值是要删除 I 点的索引号
If pHitTest.HitTest(pPoint, 4, esriGeometryPartVertex, pHitPoint, DhitDist, LpartIndex, _
    LvertexIndex, False) Then
    Set pPolygon = pFeature.Shape
    '删除 Polygon 上选中的顶点
    Call RemoveVertex(pPolygon, LvertexIndex, LpartIndex)
    Set pFeature.Shape = pPolygon
    pFeature.Store
    Set pFeature = Nothing

```

```

        pMxDocument.ActiveView.Refresh
    Else
        ' 如果没有在偏差范围内, 提示重新选点
        MsgBox "点不在 Polygon 偏差范围内, 请重新再点"
        Exit Sub
    End If

    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Public Sub RemoveVertex(pPolygon As IPolygon, lRemovePointIndex As Long, lRemovePartIndex
as long)
    Dim pPointCollection As IPointCollection
    Dim pGeoCollection As IGeometryCollection
    Dim pRing As IRing

    On Error GoTo ErrorHandler
    If pPolygon Is Nothing Then Exit Sub
    Set pGeoCollection = pPolygon
    ' 判断 Polygon 是否是一个 Part
    If pGeoCollection.GeometryCount = 1 Then
        ' 是一个 Part 就把 pPolygon 赋给 pPointCollection
        Set pPointCollection = pPolygon
    Else
        ' 如果有多个 Part, 就把 Feature 的第一个 Part 赋给 pPointCollection
        If pGeoCollection.Geometry(lRemovePartIndex).GeometryType = esriGeometryRing
Then
            Set pRing = pGeoCollection.Geometry(lRemovePartIndex)
            Set pPointCollection = pRing
        End If
    End If
    ' 删除 Polygon 中索引为 lRemovePointIndex 的一个顶点
    pPointCollection.RemovePoints lRemovePointIndex, 1
    pPointCollection.UpdatePoint 0, pPointCollection.Point(0)
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1. 5. 15. 如何移动一个图形上的一个顶点

本例通过引用接口 IPolygonMovePointFeedBack 实现在一个 Polygon 上面移动一个顶点。

● 要点

对于类型为 Polygon 的 Feature, 本例使用 IPolygonMovePointFeedBack 这个接口的 Start 和 Stop 方法来移动 Polygon 上被鼠标选中的顶点。其中 Start

(*Polygon*, *pointIndex*, *Point*)方法中, 参数 *Polygon* 是被移动顶点所在的 *Polygon*, *pointIndex* 是顶点在 *Polygon* 的索引位置, *Point* 是鼠标点击的位置点。要移动 *Polyline* 或其它图形的顶点方法类似, 只需相应地改变接口类型即可。

●程序说明

在工具条上添加一个 *ToolButton*, 通过对该 *ToolButton* 的 *MouseDown()*, *MouseMove()*, *MouseUp()* 事件的响应来实现对用户所选 *Polygon* 的顶点的移动。该方法仅适用于单个 *Part* 的 *Polygon*, 如果一个 *Polygon* 是由多个 *Part* 组成则本方法不适用。

●代码

```
Private m_pMxDocument      As IMxDocument
Private m_pActiveView      As IActiveView
Private m_pScreenDisplay   As IScreenDisplay
Private m_pPolygonMovePF   As IPolygonMovePointFeedback
Private m_pFeature         As IFeature

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long _
                                     ByVal x As Long, ByVal y As Long)

    Dim pPoint              As IPoint
    Dim pPolygon            As IPolygon
    Dim pGeometry           As IGeometry
    Dim pSpatialFilter      As ISpatialFilter
    Dim pEnvelope           As IEnvelope
    Dim pFeatureLayer       As IFeatureLayer
    Dim pFeatureCursor      As IFeatureCursor
    Dim pHitTest            As IHitTest
    Dim pPtHit              As IPoint
    Dim DblHitDis           As Double
    Dim LngPrtIdx           As Long
    Dim LngSegIdx           As Long
    Dim i                   As Integer
    Dim index               As Integer
    Dim BoolHitRt           As Boolean
    Dim BoolHitTest         As Boolean

    On Error GoTo ErrorHandler
    Set m_pMxDocument = ThisDocument
    Set m_pActiveView = m_pMxDocument.ActiveView
    Set m_pScreenDisplay = m_pActiveView.ScreenDisplay
    '得到鼠标点击的起点坐标
    Set pPoint = m_pScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
    '得到当前鼠标位置的 Envelope
    Set pGeometry = m_pMxDocument.CurrentLocation.Envelope
    Set pSpatialFilter = New SpatialFilter
    Set pEnvelope = pGeometry.Envelope
```

```

' 扩大 Envelope 的范围便于搜索
pEnvelope.Expand 0.2, 0.2, False
' 设置空间检索的条件
With pSpatialFilter
    Set .Geometry = pEnvelope
    .GeometryField = "SHAPE"
    .SpatialRel = esriSpatialRelIntersects
End With
' 在当前 Map 的所有 FeatureLayer 中查找所要改动的图形
i = 0
index = m_pMxDocument.FocusMap.LayerCount
Do While i < index
    Set pFeatureLayer = m_pMxDocument.FocusMap.Layer(i)
    Set pFeatureCursor = pFeatureLayer.FeatureClass.Search(pSpatialFilter, True)
    Set m_pFeature = pFeatureCursor.NextFeature

    If Not m_pFeature Is Nothing Then
        Exit Do
    Else
        i = i + 1
    End If
Loop
' 若用户所选图形不为空, 则移动所选顶点
If Not m_pFeature Is Nothing Then
    Set pPolygon = m_pFeature.Shape
    Set pHitTest = pPolygon
    ' 查询用户选中的点是否是图形的顶点
    BoolHitTest = pHitTest.HitTest(pPoint, 1, esriGeometryPartVertex, pPtHit,
    DblHitDis, LngPrtIdx, LngSegIdx, BoolHitRt)
    If BoolHitTest Then
        Set m_pPolygonMovePF = New PolygonMovePointFeedback
        Set m_pPolygonMovePF.Display = m_pScreenDisplay
        ' 根据顶点索引 LngSegIdx 找到所选顶点并进行移动
        m_pPolygonMovePF.Start pPolygon, LngSegIdx, pPoint
    End If
End If
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControl1_MouseMove(ByVal button As Long, ByVal shift As Long, _
    ByVal x As Long, ByVal y As Long)

    On Error GoTo ErrorHandler
    If Not m_pPolygonMovePF Is Nothing Then
        Dim pPoint As IPoint
        ' 得到鼠标点击的当前位置在地图上的坐标
        Set pPoint = m_pScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
        m_pPolygonMovePF.MoveTo pPoint
    End If
Exit Sub

```



```

ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControl1_MouseUp(ByVal button As Long, ByVal shift As Long,
    ByVal x As Long, ByVal y As Long)

    Dim pPolygon As IPolygon

    On Error GoTo ErrorHandler
    If Not m_pPolygonMovePF Is Nothing Then
        '得到顶点移动后的图形并更新
        Set pPolygon = m_pPolygonMovePF.Stop
        Set m_pFeature.Shape = pPolygon
        m_pFeature.Store
        Set m_pPolygonMovePF = Nothing
        m_pActiveView.Refresh
    End If
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.6. Element

1.6.1. 如何创建 MarkerElement

本例要实现的功能是根据鼠标在 Map 上点击的位置增加 MarkerElement。

● 要点

本例中，首先需要在 Map 上创建一个 MarkerElement 元素，然后再设置该元素的属性。实现此功能，用到了 IMarkerElement 接口。以下是该接口的主要属性的介绍：

Symbol: 设置 Marker 元素的风格。在设置 Marker 元素风格的时候可以使用很多 MarkerSymbol 接口,这里重点介绍一下本例中用到的 ISimpleMarkerSymbol 的使用。

Angle: 设置旋转角度

Color: 设置颜色

Outline: 是否显示边框

OutlineSize: 边框宽度

OutlineColor: 边框颜色

Size: 大小

Style: Marker 的风格, 有圆形, 正方形, 叉型等可供选择

● 程序说明

函数 AddTextToLayout 根据传入的点 (x,y) 参数在 Map 上添加一个 MarkerElement。

● 代码

```

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x

```

```

As Long, _
    ByVal y As Long)
    AddMarkerElement x, y
End Sub

Sub AddMarkerElement(x, y)
    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pGraphicsContainer As IGraphicsContainer
    Dim pMarkerElement As IMarkerElement
    Dim pSimpleMarkerS As ISimpleMarkerSymbol
    Dim pElement As IElement
    Dim pRgbColor As IRgbColor

    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    Set pActiveView = pMxDocument.FocusMap
    Set pGraphicsContainer = pMxDocument.FocusMap
    Set pMarkerElement = New MarkerElement

    Set pRgbColor = New RgbColor
    pRgbColor.Red = 0
    pRgbColor.Blue = 0
    pRgbColor.Green = 100
    ' 设置大小, 颜色, 风格, 旋转角度
    Set pSimpleMarkerS = New SimpleMarkerSymbol
    pSimpleMarkerS.Size = 50
    pSimpleMarkerS.Color = pRgbColor
    pSimpleMarkerS.Style = esriSMSCross
    pSimpleMarkerS.Angle = 45

    pMarkerElement.Symbol = pSimpleMarkerS
    Set pElement = pMarkerElement
    pElement.Geometry = pActiveView.ScreenDisplay.DisplayTransformation._
        oMapPoint(x, y)
    pGraphicsContainer.AddElement pMarkerElement, 0
    pMxDocument.ActiveView.Refresh
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.6.2. 如何创建 TextElement

本例要实现的功能是根据鼠标在 Map 上点击的位置添加 Text 元素。

● 要点

实现本例的功能，首先需要在 Map 上创建一个 Text 元素，然后再设置该元素的属性。主要通过接口 ITextElement 和 IGraphicsContainer 来实现。

ITextElement 接口是用来控制 Text 元素，以下是它的几个主要属性：

ScaleText: BOOL 型，表示地图比例尺变化时 Text 大小是否变化；

Symbol: 用来设置 Text 元素的风格；

Text: 用来设置 Text 元素的内容。

IGraphicsContainer 是用来控制 PageLayout, Map 等对象上图形元素的接口。

以下是它的几个主要属性和方法:

AddElement: 向层中增加一个元素;

DeleteAllElements: 删除所有的元素;

FindFrame: 查找可以放在该容器中的某对象, 例如 Text 元素;

Next: 返回该容器中的下一个对象;

UpdateElement: 更新某个元素。

● 程序说明

函数 AddText 根据鼠标点击的位置点(x, y)在 PageLayout 上添加一个文本元素。

● 代码

```
Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _  
    ByVal y As Long)  
    AddText x, y  
End Sub  
  
Sub AddText(x As Long, y As Long)  
    Dim pMxDocument As IMxDocument  
    Dim pActiveView As IActiveView  
    Dim pGraphicsContainer As IGraphicsContainer  
    Dim pTextElement As ITextElement  
    Dim pElement As IElement  
  
    On Error GoTo ErrorHandler  
    Set pMxDocument = ThisDocument  
    Set pActiveView = pMxDocument.FocusMap  
    Set pGraphicsContainer = pMxDocument.FocusMap  
    Set pTextElement = New TextElement  
    Set pElement = pTextElement  
    ' 设置 Text 元素的内容  
    pTextElement.Text = "My Map"  
    ' 将元素的图形定位在点(x, y)处  
    pElement.Geometry = pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(x, y)  
    ' 向 Map 中添加元素  
    pGraphicsContainer.AddElement pTextElement, 0  
    ' 刷新  
    pMxDocument.ActiveView.PartialRefresh esriViewGraphics, Nothing, Nothing  
    Exit Sub  
  
ErrorHandler:  
    MsgBox Err.Description  
End Sub
```

1.6.3. 如何创建 Balloon Callout

本例要实现的功能是在 ActiveView 的中心位置添加一个 Balloon Callout。

- 要点

Balloon Callout 是一种具有特殊特征的 TextElement。普通的 TextElement 的 Symbol 属性是 ITextSymbol 类型的，但是要创建 Balloon Callout 就要用到 IFormattedTextSymbol 和 ICallout 两个接口。通过 IFormattedTextSymbol 中的 Background 属性可以创建各种特殊的 TextElement，在本例中它用类 BalloonCallout 进行实例化。

- 程序说明

运行函数 AddBalloonCallout 将在 ActiveView 的中心位置添加一个 BalloonCallout。

- 代码

```
Public Sub AddBalloonCallout()  
    Dim pMxDocument As IMxDocument  
    Dim pTextElement As ITextElement  
    Dim pElement As IElement  
    Dim pPoint As IPoint  
    Dim pCallout As ICallout  
    Dim pTextSymbol As IFormattedTextSymbol  
    Dim pGraphicsContainer As IGraphicsContainer  
    Dim midX As Double  
    Dim midY As Double  
  
    On Error GoTo ErrorHandler  
    Set pMxDocument = ThisDocument  
    ' 创建一个 TextElement  
    Set pTextElement = New TextElement  
    Set pElement = pTextElement  
    pTextElement.Text = "Text callout" & vbCrLf & "In the middle of the screen"  
    ' 把 TextElement 放在屏幕的中央  
    midX = (pMxDocument.ActiveView.Extent.XMax + pMxDocument.ActiveView.Extent.XMin) / 2  
    midY = (pMxDocument.ActiveView.Extent.YMax + pMxDocument.ActiveView.Extent.YMin) / 2  
    Set pPoint = New esriCore.Point  
    pPoint.PutCoords midX, midY  
    pElement.Geometry = pPoint  
    ' 设置 TextElement 的风格为默认气球类型的 callout  
    Set pTextSymbol = New TextSymbol  
    Set pCallout = New BalloonCallout  
    Set pTextSymbol.Background = pCallout  
    ' 利用如下规则计算出 TextElement 的锚点的位置  
    pPoint.PutCoords midX - pMxDocument.ActiveView.Extent.Width / 4, _  
        midY + pMxDocument.ActiveView.Extent.Width / 20  
    pCallout.AnchorPoint = pPoint  
    pTextElement.Symbol = pTextSymbol  
    Set pGraphicsContainer = pMxDocument.ActiveView
```

```

pGraphicsContainer.AddElement pElement, 0
pElement.Activate pMxDocument.ActiveView.ScreenDisplay
pMxDocument.ActiveView.PartialRefresh esriViewGraphics, pElement, Nothing
Exit Sub

ErrorHandler:
Msgbox Err.Description
End Sub

```

1.6.4. 如何创建 PolygonElement

本例要实现的功能是在 ActiveView 的中心位置添加一个 Polygon Element。

● 要点

在 ArcMap 中创建二维图形的 Element 主要是通过接口 IFillShapeElement 实现的。实现 IFillShapeElement 接口的类有 5 个：CircleElement、EllipseElement、MultiPatchElement、PolygonElement 和 RectangleElement，分别实现 5 种图形类型的 Element。接口 IFillShapeElement 只有一个属性 Symbol，类型为 IFillSymbol，用来填充二维图形的 Element。

创建 PolygonElement，就是用类 PolygonElement 实例化 IFillShapeElement 接口的变量，如果要创建其它图形类型的 Element，方法也是类似。一般一个二维的 Element 都需要用 Symbol 进行填充，通过 IFillSymbol 实现。

● 程序说明

通过鼠标点击来确定 Polygone 的顶点，再用指定好的风格来填充这个 Polygon。

● 代码

```

Private Sub UIToolControl1_MouseDown(ByVal button As Long, _
    ByVal shift As Long, ByVal x As Long, ByVal y As Long)
    Dim pMxDoc As IMxDocument
    Dim pActiveView As IActiveView
    Dim pFillShapeElement As IFillShapeElement
    Dim pElement As IElement
    Dim pSFillSymbol As ISimpleFillSymbol
    Dim pRubberBand As IRubberBand
    Dim pPolygon As IPolygon
    Dim pColor As IColor

    On Error GoTo ErrorHandler
    Set pMxDoc = ThisDocument
    Set pActiveView = pMxDoc.FocusMap
    ' 创建一个 PolygonElement
    Set pFillShapeElement = New PolygonElement
    Set pElement = pFillShapeElement
    Set pColor = New RgbColor

```

```

pColor.RGB = RGB(0, 0, 255)
Set pSFillSymbol = New SimpleFillSymbol
With pSFillSymbol
    .Color = pColor
    .Style = esriSFSDiagonalCross
End With
' 设置 PolygonElement 的填充 Symbol
pFillShapeElement.Symbol = pSFillSymbol
' 画 polygon
Set pRubberBand = New RubberPolygon
Set pPolygon = pRubberBand.TrackNew(pActiveView.ScreenDisplay, Nothing)
pElement.Geometry = pPolygon
' 把 TextElement 加到 Map 上
pActiveView.GraphicsContainer.AddElement pElement, 0
pActiveView.PartialRefresh esriViewGraphics, Nothing, Nothing
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.6.5. 如何选中一个 Element

本例要实现的功能是选择 Map 上在鼠标点击位置上的或者在鼠标拖动范围内的 Elements。

● 要点

实现本例的功能用到两个重要的接口 IGraphicsContainerSelect 和 IGraphicsContainer。

IGraphicsContainerSelect 是用来控制 Graphics Container 中选择的元素。以下是它的几个主要方法：

SelectAllElements：选中所有元素；

SelectElement：选中指定的元素；

UnselectAllElements：不选中任何元素。

IGraphicsContainer 是用来控制 Graphics Container 中所有元素。在本例中主要用到它以下两个方法：

LocateElements：根据点和偏差值选择元素；

LocateElementsByEnvelope：根据范围选择元素。

● 程序说明

通过鼠标点击或者拖动来选择 Map 上的 Element。

● 代码

```
Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, _
```

```

                                ByVal x As Long, ByVal y As Long)
Dim pMxDocument                As IMxDocument
Dim pMap                       As IMap
Dim pElement                   As IElement
Dim pMapGraphicsSelect         As IGraphicsContainerSelect
Dim pGraphicsContainer         As IGraphicsContainer
Dim pRubberBand                As IRubberBand
Dim pEnv                       As IEnvelope
Dim pEnumElem                  As IEnumElement
Dim DSrchDis                   As Double
Dim pPoint                     As IPoint

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pGraphicsContainer = pMap
Set pMapGraphicsSelect = pMap
Set pRubberBand = New RubberEnvelope
Set pEnv = pRubberBand.TrackNew(pMxDocument.ActiveView.ScreenDisplay, Nothing)
pMapGraphicsSelect.UnselectAllElements
' 根据 pEnv 确定哪些元素需要选中
If pEnv.IsEmpty Then
    Set pPoint = _
        pMxDocument.ActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
    DSrchDis = pMxDocument.ActiveView.Extent.Width / 200
    Set pEnumElem = pGraphicsContainer.LocateElements(pPoint, DSrchDis)
Else
    Set pEnumElem = pGraphicsContainer.LocateElementsByEnvelope(pEnv)
End If
' 选中指定的元素
If Not pEnumElem Is Nothing Then
    pMapGraphicsSelect.UnselectAllElements
    pEnumElem.Reset
    pMapGraphicsSelect.SelectElements pEnumElem
End If
pMxDocument.ActiveView.Refresh
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.6.6. 如何移动 Element

本例实现的功能是让一个 Element 跟着鼠标移动，包括 MarkerElement、LineElement、PolygonElement、RectangleElement 等。

● 要点

本例中使用了一个重要的接口 IDisplayFeedback，该接口有众多针对具体图形的子接口，如 IMovePolygonFeedback、IMoveLineFeedback、IMoveTextFeedback 等。接口 IDisplayFeedback 具有如下属性和方法：

Display: 设置 Feedback 所利用的显示属性;
MoveTo: 当鼠标移动时, 设置鼠标点所在的坐标;
Refresh: 刷新画面;
Symbol: 设置显示风格。

- 程序说明

函数 GetHitElement 通过传入的坐标点, 返回 GraphicsContainer 中被该点选中的一个 Element。

- 代码

```
Option Explicit

Private m_pMxDocument As IMxDocument
Private m_pActiveView As IActiveView
Private m_pScreenDisplay As IScreenDisplay
Private m_pDisplayFeedback As IDisplayFeedback
Private m_pElement As IElement
Private m_pGraphicsContainer As IGraphicsContainer
Private m_pMouseCursor As IMouseCursor

Private Sub UIToolControl_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)

    Dim pPoint As IPoint
    Dim pGeometry As IGeometry

    On Error GoTo ErrorHandler
    ' 得到鼠标点击位置在地图上的坐标
    Set pPoint = m_pScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
    ' 得到鼠标点击位置的第一个元素 (如果存在)
    Set m_pElement = GetHitElement(pPoint)
    ' 如果存在一个元素, 则检测该元素的类型 (Point, Polyline, Envelope or Polygon)
    If Not m_pElement Is Nothing Then
        m_pMouseCursor.SetCursor 5
        Set pGeometry = m_pElement.Geometry
        ' Point
        If TypeOf pGeometry Is IPoint Then
            Set m_pDisplayFeedback = New MovePointFeedback
            Set m_pDisplayFeedback.Display = m_pScreenDisplay
            Dim pMovePointF As IMovePointFeedback
            Set pMovePointF = m_pDisplayFeedback
            pMovePointF.Start pGeometry, pPoint
        ' Polyline
        ElseIf TypeOf pGeometry Is esriCore.IPolyline Then
            Set m_pDisplayFeedback = New MoveLineFeedback
            Set m_pDisplayFeedback.Display = m_pScreenDisplay
            Dim pMoveLineF As IMoveLineFeedback
            Set pMoveLineF = m_pDisplayFeedback
            pMoveLineF.Start pGeometry, pPoint
        ' Rectangle (Envelope)
        ElseIf TypeOf pGeometry Is IEnvelope Then
```



```

        Set m_pDisplayFeedback = New MoveEnvelopeFeedback
        Set m_pDisplayFeedback.Display = m_pScreenDisplay
        Dim pMoveEnvelopeF As IMoveEnvelopeFeedback
        Set pMoveEnvelopeF = m_pDisplayFeedback
        pMoveEnvelopeF.Start pGeometry, pPoint
    ' Polygon
ElseIf TypeOf pGeometry Is IPolygon Then
    Set m_pDisplayFeedback = New MovePolygonFeedback

    Set m_pDisplayFeedback.Display = m_pScreenDisplay
    Dim pMovePolygonF As IMovePolygonFeedback
    Set pMovePolygonF = m_pDisplayFeedback
    pMovePolygonF.Start pGeometry, pPoint
End If
End If
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControl1_MouseMove(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)

    On Error GoTo ErrorHandler
    If Not m_pDisplayFeedback Is Nothing Then
        Dim pPoint As IPoint
        ' 得到鼠标点击位置在地图上的坐标
        Set pPoint = m_pScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
        m_pDisplayFeedback.MoveTo pPoint
    End If
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControl1_MouseUp(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)

    Dim pGeomResult As IGeometry
    Dim pGeometry As IGeometry

    On Error GoTo ErrorHandler
    ' 是否存在一个元素
    If Not m_pElement Is Nothing Then
        Set pGeometry = m_pElement.Geometry
        ' 检测此元素的类型(Point, Polyline, Envelope or Polygon)
        ' Point
        If TypeOf pGeometry Is IPoint Then
            Dim pMovePointF As IMovePointFeedback
            Set pMovePointF = m_pDisplayFeedback
            Set pGeomResult = pMovePointF.Stop
        ' Polyline
        ElseIf TypeOf pGeometry Is IPolyline Then
            Dim pMoveLineF As IMoveLineFeedback
            Set pMoveLineF = m_pDisplayFeedback

```

```

        Set pGeomResult = pMoveLineF.Stop
    'Envelope
    ElseIf TypeOf pGeometry Is IEnvelope Then
        Dim pMoveEnvelopeF As IMoveEnvelopeFeedback
        Set pMoveEnvelopeF = m_pDisplayFeedback
        Set pGeomResult = pMoveEnvelopeF.Stop
    'Polygon
    ElseIf TypeOf pGeometry Is IPolygon Then
        Dim pMovePolygonF As IMovePolygonFeedback
        Set pMovePolygonF = m_pDisplayFeedback
        Set pGeomResult = pMovePolygonF.Stop
    End If
    '更新元素
    m_pElement.Geometry = pGeomResult
    m_pGraphicsContainer.UpdateElement m_pElement
    Set m_pDisplayFeedback = Nothing
    Set m_pElement = Nothing
    m_pActiveView.Refresh
    m_pMouseCursor.SetCursor 0
End If
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIToolControll1_Refresh(ByVal hDC As Long)
    Set m_pMxDocument = ThisDocument
    Set m_pActiveView = m_pMxDocument.ActiveView
    Set m_pScreenDisplay = m_pActiveView.ScreenDisplay
    Set m_pMouseCursor = New MouseCursor
    m_pMouseCursor.SetCursor 0
End Sub

Private Sub UIToolControll1_Select()
    Set m_pMxDocument = ThisDocument
    Set m_pActiveView = m_pMxDocument.ActiveView
    Set m_pScreenDisplay = m_pActiveView.ScreenDisplay
    Set m_pMouseCursor = New MouseCursor
    m_pMouseCursor.SetCursor 0
End Sub

Private Function GetHitElement(pInPt As IPoint) As IElement
    Dim pEnumElem As IEnumElement
    Dim DSrchDis As Double

    On Error GoTo ErrorHandler
    Set m_pGraphicsContainer = m_pActiveView
    '计算出一个缓冲范围
    DSrchDis = m_pActiveView.Extent.Width / 200
    Set pEnumElem = m_pGraphicsContainer.LocateElements(pInPt, DSrchDis)
    If Not pEnumElem Is Nothing Then
        Set GetHitElement = pEnumElem.Next
    End If

```

```
Exit Function

ErrorHandler:
    MsgBox Err.Description
End Function
```

1.6.7. 如何排列 Element

本例实现的功能是按照占支配地位的 Element 所在的位置排列选中的 Elements。比如按照 Top、Bottom、Left、Right 等方式排列。在 ArcMap 中最后被选中的 Element 作为占支配地位的 Element。

● 要点

本例中使用了一个重要的接口 ITransform2D，该接口具有如下方法：

Move：根据输入的 dx, dy 移动对象；

MoveVector：根据输入的矢量移动对象；

Rotate：根据输入的角度旋转对象。

● 程序说明

本例中编写了三个函数，下面逐一介绍：

Move：根据输入的 dx, dy 移动指定的元素；

AlignPos：返回 pEnvelopeMove 向 pEnvelopeHome 对齐时需要移动的距离，参数 lControl 用来设置对齐的种类。

Align：参数 lControl 设置对齐的种类。

0 表示左对齐；

1 表示水平居中对齐；

2 表示右对齐；

3 表示置顶对齐

4 表示垂直居中对齐；

5 表示底部对齐。

本例通过 DLL 实现，如何添加菜单参见本书 1.2.4 节和 1.2.5 节。

● 代码

```
Public Sub Move(pElementMove As IElement, dx As Long, dy As Long)
    Dim pMxDocument As IMxDocument
    Dim pTransform2D As ITransform2D
    Dim pGraphicsContainer As IGraphicsContainer
    Dim pGraphicsContainerS As IGraphicsContainerSelect
    On Error GoTo ErrorHandler
    Set pMxDocument = m_pApplication.Document
```

```

Set pGraphicsContainer = pMxDocument.ActiveView
Set pGraphicsContainerS = pMxDocument.ActiveView

pGraphicsContainer.Reset
If pGraphicsContainerS.ElementSelectionCount = 0 Then
    Exit Sub
End If
'检索所有选中的元素
Set pTransform2D = pElementMove
'移动选中的元素
pTransform2D.Move dx, dy
pGraphicsContainer.UpdateElement pElementMove
Exit Function
ErrorHandler:
MsgBox Err.Description
End Sub

Private Function AlignPos(pEnvelopeHome As IEnvelope, pEnvelopeMove As IEnvelope, _
    lControl As Long) As IPoint
    Dim pPoint As IPoint
    On Error GoTo ErrorHandler

    Set pPoint = New Point
    Select Case lControl
        'align left
        Case 0
            pPoint.X = pEnvelopeHome.XMin - pEnvelopeMove.XMin
            pPoint.Y = 0
        'align center
        Case 1
            pPoint.X = (pEnvelopeHome.XMin + pEnvelopeHome.Width / 2) _
                - (pEnvelopeMove.XMin + pEnvelopeMove.Width / 2)
            pPoint.Y = 0
        'align right
        Case 2
            pPoint.X = pEnvelopeHome.XMax - pEnvelopeMove.XMax
            pPoint.Y = 0
        'align top
        Case 3
            pPoint.X = 0
            pPoint.Y = pEnvelopeHome.YMax - pEnvelopeMove.YMax
        'align vertical center
        Case 4
            pPoint.X = 0
            pPoint.Y = (pEnvelopeHome.YMin + pEnvelopeHome.Height / 2) _
                - (pEnvelopeMove.YMin + pEnvelopeMove.Height / 2)
        'align bottom
        Case 5
            pPoint.X = 0
            pPoint.Y = pEnvelopeHome.YMin - pEnvelopeMove.YMin
    End Select
    Set AlignPos = pPoint
    Exit Function
ErrorHandler:

```

```

        MsgBox Err.Description
End Function

Public Function Align(lControl As Long)
    Dim pMxDocument As IMxDocument
    Dim pGraphicsContainerS As IGraphicsContainerSelect
    Dim pElementHome As IElement
    Dim pElementMove As IElement
    Dim pPoint As IPoint
    Dim pEnumElement As IEnumElement
    Dim pEnvelopeHome As IEnvelope
    Dim pEnvelopeMove As IEnvelope
    On Error GoTo ErrorHandler

    Set pMxDocument = m_pApplication.Document
    Set pGraphicsContainerS = pMxDocument.ActiveView
    Set pEnumElement = pGraphicsContainerS.SelectedElements
    If pGraphicsContainerS.ElementSelectionCount = 0 Then
        Exit Function
    End If
    pEnumElement.Reset
    ' 设定
    Set pEnvelopeHome = New Envelope
    Set pEnvelopeMove = New Envelope
    Set pElementHome = pEnumElement.Next
    ' 得到元素所显示的 Envelope
    pElementHome.QueryBounds pMxDocument.ActiveView.ScreenDisplay, pEnvelopeHome
    Set pElementMove = pEnumElement.Next
    pElementMove.QueryBounds pMxDocument.ActiveView.ScreenDisplay, pEnvelopeMove

    Do While Not pElementMove Is Nothing
        Set pPoint = AlignPos(pEnvelopeHome, pEnvelopeMove, lControl)
        ' 移动
        Move pElementMove, pPoint.X, pPoint.Y

        Set pElementMove = pEnumElement.Next
        If Not pElementMove Is Nothing Then
            pElementMove.QueryBounds pMxDocument.ActiveView.ScreenDisplay, _
            pEnvelopeMove
        End If
    Loop
    pMxDocument.ActiveView.Refresh
    Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

```

1.6.8. 如何通过名字查询 Element

本例实现了两个功能：

1. 在鼠标点击的位置创建一个 Element，并且根据输入的内容为该 Element 设定一个名字；

2. 根据输入的名字查询 Element，并将其设为选中状态。

- 要点

设定元素的名字通过接口 IElementProperties 实现。IElementProperties 是控制元素属性的接口。其主要属性有：

Name：元素的名字；

Type：元素的类型。

- 程序说明

本例中编写了两个函数，下面逐一介绍：

AddMarkerElement：根据输入的 x, y 坐标添加一个 Element，在这个过程中可以设定 Element 的名字；

SelectByName：通过输入名字来查询元素，并将其设为选中状态。

- 代码

```
Sub AddMarkerElement(x As Long, y As Long)
    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pGraphicsContainer As IGraphicsContainer
    Dim pMarkerElement As IMarkerElement
    Dim pSimpleMarkerS As ISimpleMarkerSymbol
    Dim pElement As IElement
    Dim pRgbColor As IRgbColor
    Dim pElementProperties As IElementProperties

    On Error GoTo ErrorHandler
    frmNameInput.Show
    Set pMxDocument = ThisDocument
    Set pActiveView = pMxDocument.FocusMap
    Set pGraphicsContainer = pMxDocument.FocusMap
    Set pMarkerElement = New MarkerElement

    Set pRgbColor = New RgbColor
    pRgbColor.Red = 0
    pRgbColor.Blue = 0
    pRgbColor.Green = 100

    Set pSimpleMarkerS = New SimpleMarkerSymbol
    pSimpleMarkerS.Size = 50
    pSimpleMarkerS.Color = pRgbColor
    pSimpleMarkerS.Style = esriSMSCross
    pSimpleMarkerS.Angle = 45

    pMarkerElement.Symbol = pSimpleMarkerS
    Set pElement = pMarkerElement
    pElement.Geometry = pActiveView.ScreenDisplay.DisplayTransformation._
```

```

ToMapPoint(x, y)
pGraphicsContainer.AddElement pMarkerElement, 0
' 设定元素名字
Set pElementProperties = pMarkerElement
pElementProperties.Name = NameInput
NameInput = ""
pMxDocument.ActiveView.Refresh
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Sub SelectByName()
    Dim pElementProperties As IElementProperties
    Dim pGraphicsContainer As IGraphicsContainer
    Dim pGraphicsContainerS As IGraphicsContainerSelect
    Dim pMxDocument As ImxDocument

    On Error GoTo ErrorHandler
    frmNameInput.Show
    Set pMxDocument = ThisDocument
    Set pGraphicsContainer = pMxDocument.ActiveView
    Set pGraphicsContainerS = pMxDocument.ActiveView
    pGraphicsContainer.Reset
    pGraphicsContainerS.UnselectAllElements
    Set pElementProperties = pGraphicsContainer.Next
    ' 循环检索
    Do While Not pElementProperties Is Nothing
        If pElementProperties.Name = NameInput Then
            pGraphicsContainerS.SelectElement pElementProperties
        End If
        Set pElementProperties = pGraphicsContainer.Next
    Loop
    pMxDocument.ActiveView.Refresh
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.6.9. 如何拷贝 Element

本例实现的功能是复制各种类型的 Element，包括 MarkerElement、LineElement、PolygonElement、CircleElement、EllipseElement、RectangleElement、TextElement 和 PictureElement 等。

● 要点

本例综合了创建各种元素的方法，使用的很多接口大部分都在前面章节讲解过，这里就不再多做介绍。主要的实现方式是先获取被复制的 Element 的属性，

在按照这些属性生成一个同类型的 Element。

- 程序说明

函数 CopyElement 实现了通过判断元素的类型来复制多种元素的功能，并将复制的元素添加到 Graphics Container 中。

- 代码

```
Dim m_pMxDocument As IMxDocument
Dim m_pGraphicsContainerS As IGraphicsContainerSelect

Private Sub UIButtonControll_Click()
    Dim pElementProperties As IElementProperties
    Dim pElement As IElement
    Dim pEnumElement As IEnumElement

    On Error GoTo ErrorHandler
    Set m_pMxDocument = ThisDocument
    Set m_pGraphicsContainerS = m_pMxDocument.ActiveView
    Set pEnumElement = m_pGraphicsContainerS.SelectedElements
    pEnumElement.Reset
    Set pElementProperties = pEnumElement.Next
    ' 对所有选中的元素进行复制
    Do While Not pElementProperties Is Nothing
        CopyElement pElementProperties
        Set pElementProperties = pEnumElement.Next
    Loop
    m_pMxDocument.ActiveView.Refresh
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Sub CopyElement(pElementProperties As IElementProperties)
    Dim pSymbol As ISymbol
    Dim pEnvelope As IEnvelope
    Dim pGeometry As IGeometry
    Dim pCopyElement As IElement
    Dim pElement As IElement

    On Error GoTo ErrorHandler
    Set pElement = pElementProperties
    ' 根据不同的类型创建元素，并设置其属性
    Select Case pElementProperties.Type
        Case "Marker"
            Dim pMarkerElement As IMarkerElement
            Dim pCopyMarkerElement As IMarkerElement

            Set pMarkerElement = pElement
            Set pCopyMarkerElement = New MarkerElement
            Set pSymbol = pMarkerElement.Symbol
            pCopyMarkerElement.Symbol = pSymbol
```



```

        Set pCopyElement = pCopyMarkerElement
    Case "Line"
        Dim pLineElement          As ILineElement
        Dim pCopyLineElement      As ILineElement
        Set pLineElement = pElement
        Set pCopyLineElement = New LineElement
        Set pSymbol = pLineElement.Symbol
        pCopyLineElement.Symbol = pSymbol
        Set pCopyElement = pCopyLineElement
    Case "Text"
        Dim pTextElement          As ITextElement
        Dim pCopyTextElement      As ITextElement

        Set pTextElement = pElement
        Set pCopyTextElement = New TextElement
        Set pSymbol = pTextElement.Symbol
        With pCopyTextElement
            .Symbol = pSymbol
            .Text = pTextElement.Text
        End With
        Set pCopyElement = pCopyTextElement
    Case "Circle"
        Dim pCFillShapeE          As IFillShapeElement
        Dim pCCopyFillShapeE      As IFillShapeElement

        Set pCFillShapeE = pElement
        Set pCCopyFillShapeE = New CircleElement
        Set pSymbol = pCFillShapeE.Symbol
        pCCopyFillShapeE.Symbol = pSymbol
        Set pCopyElement = pCCopyFillShapeE
    Case "Ellipse"
        Dim pEFillShapeE          As IFillShapeElement
        Dim pECopyFillShapeE      As IFillShapeElement

        Set pEFillShapeE = pElement
        Set pECopyFillShapeE = New EllipseElement
        Set pSymbol = pEFillShapeE.Symbol
        pECopyFillShapeE.Symbol = pSymbol
        Set pCopyElement = pECopyFillShapeE
    Case "Polygon"
        Dim pPFillShapeE          As IFillShapeElement
        Dim pPCopyFillShapeE      As IFillShapeElement

        Set pPFillShapeE = pElement
        Set pPCopyFillShapeE = New PolygonElement
        Set pSymbol = pPFillShapeE.Symbol

        pPCopyFillShapeE.Symbol = pSymbol
        Set pCopyElement = pPCopyFillShapeE
    Case "Rectangle"
        Dim pRFillShapeE          As IFillShapeElement
        Dim pRCopyFillShapeE      As IFillShapeElement

        Set pRFillShapeE = pElement
        Set pRCopyFillShapeE = New RectangleElement

```

```

        Set pSymbol = pRFillShapeE.Symbol

        pRCopyFillShapeE.Symbol = pSymbol
        Set pCopyElement = pRCopyFillShapeE
    Case "Picture"
        Dim pPictureElement As IPictureElement
        Dim pCopyPictureElement As IPictureElement

        Set pPictureElement = pElement
        If TypeOf pPictureElement Is BmpPictureElement Then
            Set pCopyPictureElement = New BmpPictureElement
        Else
            Set pCopyPictureElement = New EmfPictureElement
        End If
        With pCopyPictureElement
            .ImportPictureFromFile pElementProperties.Name
            .SavePictureInDocument = pPictureElement.SavePictureInDocument
        End With

        Set pCopyElement = pCopyPictureElement
    Case Else
        MsgBox pElementProperties.Type
        Exit Sub
End Select

Set pGeometry = pElement.Geometry
pCopyElement.Geometry = pGeometry

Dim pGraphicsContainer As IGraphicsContainer
Set pGraphicsContainer = m_pMxDocument.ActiveView
pGraphicsContainer.AddElement pCopyElement, 0
' 移动 Element
Dim pTransform2D As ITransform2D
Set pTransform2D = pCopyElement
pTransform2D.Move 20, -20
m_pMxDocument.ActiveView.Refresh
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.6.10. 如何沿着折线路径显示 Text

本例实现的功能是让新生成的 Text 沿着折线路径显示。

● 要点

本例使用接口 ITextPath 来实现控制 TextElement 显示路径的功能。本例主要用到接口 ITextPath 的 Geometry 属性：TextElement 显示路径的参照图形。

● 代码

```

DPrivate Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Lon , _

```

```

ByVal x As Long, ByVal y As Long)

Dim pMxDocument As IMxDocument
Dim pRubberBand As IRubberBand
Dim pGeometry As IGeometry
Dim pSimpleTextS As ISimpleTextSymbol
Dim pTextPath As ITextPath
Dim pElement As IElement
Dim pTextElement As ITextElement
Dim pGraphicsContainer As IGraphicsContainer

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pRubberBand = New RubberLine
'画折线
Set pGeometry = pRubberBand.TrackNew(pMxDocument.ActiveView.ScreenDisplay, Nothing)
Set pTextPath = New SimpleTextPath
'设置 Text 路径
Set pTextPath.Geometry = pGeometry
Set pSimpleTextS = New TextSymbol
Set pSimpleTextS.TextPath = pTextPath
Set pTextElement = New TextElement
With pTextElement
    .Symbol = pSimpleTextS
    '设置 Text 内容
    .Text = InputBox("Text:", "Text2")
End With
Set pElement = pTextElement
pElement.Geometry = pGeometry
Set pGraphicsContainer = pMxDocument.ActiveView
'添加元素
pGraphicsContainer.AddElement pElement, 0
pMxDocument.ActiveView.Refresh
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1.7. Symbol 和 Renderer

1.7.1. 如何为一个层设置 Simple Renderer

本例要实现的是如何为一个层设置 Simple Renderer。用同一颜色填充 polygon。

● 要点

首先实例化接口 IGeoFeatureLayer，然后创建 IFillSymbol 接口对象和 ILineSymbol 接口对象来设置填充的颜色和外线框 Symbol，创建 ISimpleRenderer 接口对象并对属性设置，最后赋值给 IGeoFeatureLayer.Render 属性。

- 程序说明

本程序的加载的数据是“WorldCountries.shp”。点击 UIButtonControll 程序开始运行。过程 SimpleRenderere 根据 pSimpleRenderere 的属性值完成本图层的 Simple Renderere。

在过程 SimpleRenderere 运行中颜色数据都由调用函数 GetRGBColor 得到。

- 代码

```
Private Sub UIButtonControll_Click()  
    'call SimpleRenderere  
    SimpleRenderere  
End Sub  
  
Private Sub SimpleRenderere()  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pGeoFeatureL As IGeoFeatureLayer  
    Dim pSimpleRenderere As ISimpleRenderere  
    Dim pSimpleFillS As IFillSymbol  
    Dim pLineSymbol As ILineSymbol  
On Error GoTo ErrorHandler:  
    Set pMxDocument = ThisDocument  
    Set pMap = pMxDocument.FocusMap  
    Set pSimpleFillS = New SimpleFillSymbol  
    Set pGeoFeatureL = pMap.Layer(0)  
    ' Initialise a color object to Lilac  
    pSimpleFillS.Color = GetRGBColor(235, 202, 250)  
  
    ' Now initialise the line symbol used for the outline, set it to black  
    ' and make it width of 1 point.  
    ' Assign this into the fill symbols outline propetry.  
    Set pLineSymbol = New SimpleLineSymbol  
    pLineSymbol.Color = GetRGBColor(0, 0, 0)  
    pLineSymbol.Width = 1  
    pSimpleFillS.Outline = pLineSymbol  
  
    ' Now initialise the simple renderere and assign it a fill symbol,  
    ' by default it doesn't have a symbol  
    Set pSimpleRenderere = New SimpleRenderere  
    Set pSimpleRenderere.Symbol = pSimpleFillS  
  
    ' Now set the layers renderere property to be this simple renderere,  
    ' and refresh the screen  
    ,  
  
    Set pGeoFeatureL.Renderere = pSimpleRenderere  
    pMxDocument.ActiveView.Refresh  
    pMxDocument.UpdateContents  
    Exit Sub  
ErrorHandler:  
    MsgBox Err.Description  
End Sub
```

```

Private Function GetRGBColor(yourRed As Long, yourGreen As Long, _
                             yourBlue As Long) As IRgbColor
    Dim pRGB As IRgbColor
    Set pRGB = New RgbColor
    With pRGB
        .Red = yourRed
        .Green = yourGreen
        .Blue = yourBlue
        .UseWindowsDithering = True
    End With
    Set GetRGBColor = pRGB
End Function

```

1.7.2. 如何为一个层设置 UniqueValue Renderer

本例要实现的是如何在一个层中设置 UniqueValue Renderer，根据“PLACENAME”字段填充上不同的颜色的 polygon，并在 Table of Contents 窗口中显示出对其记数和描述。

● 要点

首先实例化接口 IGeoFeatureLayer，通过类 UniqueValueRenderer 实现 IUniqueValueRender 接口的对象实例，通过对 IUniqueValueRender 的属性进行赋值，最后赋值给 IGeoFeatureLayer.Render 属性

● 程序说明

本程序的加载的数据是“Contenties.shp”。点击 UIButtonControl1 程序开始运行。

过程 UniqueValueRenderer 根据 pUniqueValueR 的属性值填充颜色记录。

过程 UniqueValues_LabelCount_and_DescripFromField 调用函数 GetLabelDescription 实现记数和描述的功能。

● 代码

```

'Declare global variable
Dim m_pMxdDocument As IMxdDocument
Dim m_pMap As IMap
Dim m_pGeoFeatureL As IGeoFeatureLayer
Const DESCRIP_FIELD = "PLACENAME" ' Field name for unique value Renderer
Const CONCATENATE_TO_BUILD_DESCRIPTION = True
Const CONCAT_CHAR = vbNewLine

Private Sub UIButtonControl1_Click()
    ' call sub UniqueValueRenderer and UniqueValues_LabelCount_and_DescripFromF eld
    UniqueValueRenderer
    UniqueValues_LabelCount_and_DescripFromField

```

```

End Sub

Private Sub UniqueValueRenderer()
    Dim pUniqueValueR As IUniqueValueRenderer
    Dim pFillSymbol As IFillSymbol
    Dim pColor As IColor
    Dim pNextUniqueColor As IColor
    Dim pEnumRamp As IEnumColors
    Dim pTable As ITable
    Dim lfieldNumber As Long
    Dim pNextRow As IRow
    Dim pNextRowBuffer As IRowBuffer
    Dim pCursor As ICursor
    Dim pQueryFilter As IQueryFilter
    Dim codeValue As Variant
    Dim pColorRamp As IRandomColorRamp
    ' A field for the shapefile
    Const strNameField = DESCRIP_FIELD

On Error GoTo ErrorHandler:
    Set m_pMxDocument = ThisDocument
    Set m_pMap = m_pMxDocument.FocusMap
    Set m_pGeoFeatureL = m_pMap.Layer(0)
    ' Iterate through the class and a random color ramp
    ' retrieve a state name, and a corresponding random color. Put the name an
    ' color into the unique value renderer. When complete assign the renderer to the
    ' layer and refresh to display the symbology.
    ' Create a color ramp, color object and a unique value renderer to be set p
    ' later on
    ,

    Set pUniqueValueR = New UniqueValueRenderer
    ' QI the table from the geoFeatureLayer and get the field number of
    ,

    Set pTable = m_pGeoFeatureL
    lfieldNumber = pTable.FindField(strNameField)
    If lfieldNumber = -1 Then
        MsgBox "Can't find field called " & strNameField
        Exit Sub
    End If
    ' Specify the field to render unique values with
    ,

    pUniqueValueR.FieldCount = 1
    pUniqueValueR.Field(0) = strNameField
    ' Set up the Color ramp, this came from looking at ArcMaps Color Ramp
    ' properties for Pastels.
    ,

    Set pColorRamp = New RandomColorRamp
    pColorRamp.StartHue = 0
    pColorRamp.MinValue = 99
    pColorRamp.MinSaturation = 15
    pColorRamp.EndHue = 360
    pColorRamp.MaxValue = 100
    pColorRamp.MaxSaturation = 30
    pColorRamp.Size = 100
    pColorRamp.CreateRamp True

```

```

Set pEnumRamp = pColorRamp.Colors
Set pNextUniqueColor = Nothing
' Get a enumerator on the first row of the Layer
,

Set pQueryFilter = New QueryFilter
pQueryFilter.AddField strNameField
Set pCursor = pTable.Search(pQueryFilter, True)
Set pNextRow = pCursor.NextRow
' Iterate through each row, adding values and a color to the unique value renderer
' Note we don't bother filtering out duplicates,
' if we add in a second value that is already there
' the symbol changes but the value remains
,

Do While Not pNextRow Is Nothing
    ' QI the row buffer from the row and get the value
    ,

    Set pNextRowBuffer = pNextRow
    codeValue = pNextRowBuffer.Value(lfieldNumber)
    ' Get a Color object from the color ramp and advance the enumerator
    ' if we've run out then reset and start again
    ,

    Set pNextUniqueColor = pEnumRamp.Next
    If pNextUniqueColor Is Nothing Then
        pEnumRamp.Reset
        Set pNextUniqueColor = pEnumRamp.Next
    End If
    ' Set the symbol to the Color and add it to render a given value
    ,

    Set pFillSymbol = New SimpleFillSymbol
    pFillSymbol.Color = pNextUniqueColor
    pUniqueValueR.AddValue codeValue, codeValue, pFillSymbol
    ' Advance the cursor to the next row, or end of the dataset
    Set pNextRow = pCursor.NextRow
Loop
' Now set the layers renderer to the unique value renderer
Set m_pGeoFeatureL.Renderer = pUniqueValueR
m_pMxDocument.ActiveView.Refresh
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UniqueValues_LabelCount_and_DescripFromField()
    Dim pUniqueValueR          As IUniqueValueRenderer
    Dim sFieldName              As String
    Dim i                       As Integer
    Dim varValue                As Variant
    Dim pFeatureClass           As IFeatureClass
    Dim varLabelDescrip          As Variant
On Error GoTo ErrorHandler:
    If Not TypeOf m_pGeoFeatureL.Renderer Is IUniqueValueRenderer Then
        MsgBox "Current symbology is not Unique values. Exiting."
        Exit Sub
    End If

```

```

Set pUniqueValueR = m_pGeoFeatureL.Renderer

If pUniqueValueR.FieldCount > 1 Then
    MsgBox "Current Unique values symbology is based on multiple fields. Exiting."
    Exit Sub
End If

sFieldName = pUniqueValueR.Field(0)

Set pFeatureClass = m_pGeoFeatureL.FeatureClass

For i = 0 To pUniqueValueR.ValueCount - 1
    varValue = pUniqueValueR.Value(i)
    varLabelDescrip = GetLabelDescription(pFeatureClass, _
                                         pUniqueValueR.Field(0), varValue)
    pUniqueValueR.Label(varValue) = varLabelDescrip(0)
    pUniqueValueR.Description(varValue) = varLabelDescrip(1)
Next i

m_pMxDocument.ActiveView.ContentChanged
m_pMxDocument.UpdateContents
m_pMxDocument.ActiveView.Refresh
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Function GetLabelDescription(pFeatureClass As IFeatureClass, _
    ValField As String, Value As Variant) As Variant
    ' returns an array of length 2
    ' (0) is the new label (string) appended with count of features
    ' (1) is the new descrip (string) driven from DESCRIP_FIELD
    Dim pQueryFilter As IQueryFilter
    Dim pFeatureCursor As IFeatureCursor
    Dim pFeature As IFeature
    Dim sDescrip As String
    Dim iDescrip As Integer
    Dim iCount As Integer
    Dim bCountsDetermined As Boolean
    Dim sLabel As String
    Dim sReturnArray(2) As String

On Error GoTo ErrorHandler:
    Set pQueryFilter = New QueryFilter
    pQueryFilter.WhereClause = ValField & " = '" & CStr(Value) & "'"
    pQueryFilter.AddField DESCRIP_FIELD
    Set pFeatureCursor = pFeatureClass.Search(pQueryFilter, False)
    ' -----
    ' Description
    iDescrip = pFeatureClass.Fields.FindField(DESCRIP_FIELD)
    Set pFeature = pFeatureCursor.NextFeature

    iCount = 0
    bCountsDetermined = False

```



```

If CONCATENATE_TO_BUILD_DESCRIPTION Then
    bCountsDetermined = True
    Do While Not pFeature Is Nothing
        iCount = iCount + 1
        If sDescrip <> "" Then sDescrip = sDescrip + CONCAT_CHAR
        ' get value from DESCRIP_FIELD
        sDescrip = sDescrip + CStr(pFeature.Value(iDescrip))
        Set pFeature = pFeatureCursor.NextFeature
    Loop
Else ' only get descrip from first feature found
    If Not pFeature Is Nothing Then
        ' get value from DESCRIP_FIELD
        sDescrip = CStr(pFeature.Value(iDescrip))
    End If
End If

' -----
' Label
If Not bCountsDetermined Then
    ' optimization: re-query only if we don't
    ' already have the counts from above
    iCount = pFeatureClass.FeatureCount(pQueryFilter)
End If

sLabel = Value & " (" & iCount & ") "
' -----
' setup return array and return
sReturnArray(0) = sLabel
sReturnArray(1) = sDescrip
GetLabelDescription = sReturnArray
Exit Function
ErrorHandler:
    MsgBox Err.Description
End Function

```

1.7.3. 如何为一个层设置 ClassBreaks Renderer

本例要实现的是在一个层设置 ClassBreaks Renderer，在 states 层中根据人口（Pop1990）字段的数据，将它分成五个等级，填充以不同的颜色。

● 要点

首先实例化 IGeoFeatureLayer，再创建 ITableHistogram 接口实例来接受数据，通过 IHistogram 和 IClassify 接口对象对数据进行处理分成五个等级，创建 ISimpleFillSymbol 接口对象，创建 IClassBreaksRenderer 接口对象接收 symbol 和 break 的设置信息，最后赋值给 IGeoFeatureLayer.Renderer 属性

● 程序说明

本程序的加载的数据是“states.shp”。点击 UIButtonControl11 程序开始运

行。

过程 ClassBreaksRenderer 根据 pClassBreaksRenderer 的属性值来控制 Map 的显示效果。

● 代码

```
Private Sub UIButtonControl1_Click()  
    'call sub ClassBreaksRenderer  
    ClassBreaksRenderer  
End Sub  
  
Private Sub ClassBreaksRenderer()  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pGeoFeatureL As IGeoFeatureLayer  
    Dim pTable As ITable  
    Dim pClassify As IClassify  
    Dim pTableHistogram As ITableHistogram  
    Dim pHistogram As IHistogram  
    Dim dataFrequency As Variant  
    Dim dataValues As Variant  
    Dim strOutput As String  
    Dim Classes() As Double  
    Dim ClassesCount As Long  
    Dim pClassBreaksRenderer As IClassBreaksRenderer  
    Dim pFromColor As IHsvColor  
    Dim pToColor As IHsvColor  
    Dim pAlgorithmicCR As IAlgorithmicColorRamp  
    Dim pEnumColors As IEnumColors  
    Dim ok As Boolean  
    Dim pColor As IColor  
    Dim pSimpleFillS As ISimpleFillSymbol  
    Dim lbreakIndex As Long  
    'A field for the shapefile  
    Const strPopField = "POP1990"  
    Const numDesiredClasses As Long = 5  
    ' We're going to retrieve frequency data from a population field  
    ' and then classify this data  
    ,  
  
On Error GoTo ErrorHandler:  
    Set pMxDocument = ThisDocument  
    Set pMap = pMxDocument.FocusMap  
    Set pGeoFeatureL = pMap.Layer(0)  
    Set pTable = pGeoFeatureL  
    Set pTableHistogram = New TableHistogram  
    Set pHistogram = pTableHistogram  
    ' Get values and frequencies for the population field  
    ' into a table histogram object  
    pTableHistogram.Field = strPopField  
    Set pTableHistogram.Table = pTable  
    pHistogram.GetHistogram dataValues, dataFrequency  
    ' Put the values and frequencies into an Equal Interval classify object  
    ,
```

```

Set pClassify = New EqualInterval
pClassify.SetHistogramData dataValues, dataFrequency
' Now a generate the classes
' Note:
' 1/ The number of classes returned may be different from requested
'    (depends on classification algorithm)
' 2/ The classes array starts at index 0 and has datavalues starting
'    from the minumum value, going to maximum
'
pClassify.Classify numDesiredClasses
Classes = pClassify.ClassBreaks
ClassesCount = UBound(Classes)

' Initialise a new class breaks renderer and supply the number of
' class breaks and the field to perform the class breaks on.
'
Set pClassBreaksRenderer = New ClassBreaksRenderer
pClassBreaksRenderer.Field = strPopField
pClassBreaksRenderer.BreakCount = ClassesCount
pClassBreaksRenderer.SortClassesAscending = True

' Use an algorithmic color ramp to generate an range of colors between
' yellow to red (taken from ArcMaps colorramp properties)
'
' Set the initial color to yellow
'
Set pFromColor = New HsvColor
pFromColor.Hue = 60      ' Yellow
pFromColor.Saturation = 100
pFromColor.Value = 96
' Set the final color to be red
'
Set pToColor = New HsvColor
pToColor.Hue = 0        ' Red
pToColor.Saturation = 100
pToColor.Value = 96
' Set up the HSV colour ramp to span from yellow to red
'
Set pAlgorithmicCR = New AlgorithmicColorRamp
pAlgorithmicCR.Algorithm = esriHSVAlgorithm
pAlgorithmicCR.FromColor = pFromColor
pAlgorithmicCR.ToColor = pToColor
pAlgorithmicCR.Size = ClassesCount
pAlgorithmicCR.CreateRamp ok
Set pEnumColors = pAlgorithmicCR.Colors
' Iterate through each class brake, setting values and corresponding
' fill symbols for each polygon, note we skip the minimum value (classes(0) )
'
For lbreakIndex = 0 To ClassesCount - 1
    ' Retrieve a color and set up a fill symbol,
    ' put this in the symbol array corresponding to the class value
    '
    Set pColor = pEnumColors.Next
    Set pSimpleFillS = New SimpleFillSymbol
    pSimpleFillS.Color = pColor

```

```

pSimpleFillS.Style = esriSFSSolid
pClassBreaksRenderer.Symbol(lbreakIndex) = pSimpleFillS
pClassBreaksRenderer.Break(lbreakIndex) = Classes(lbreakIndex + 1)
' Store each break value for user output
strOutput = strOutput & "- " & Classes(lbreakIndex + 1) & vbNewLine
Next lbreakIndex
' Assign the renderer to the layer and update the display
,

Set pGeoFeatureL.Renderer = pClassBreaksRenderer
pMxDocument.ActiveView.Refresh
pMxDocument.UpdateContents
Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

```

1.7.4. 如何为一个层设置 ProportionalSymbol Renderer

本例要实现的是如何在一个层中设置 ProportionalSymbol Renderer。按 states 层中人口字段（Pop1990）值的大小按比例画出 Symbols。

● 要点

首先实例化 IGeoFeatureLayer, 创建 IDataStatistics 接口对象来计算数据的最大最小值, 然后创建 ICharacterMarkerSymbol 接口对象, 将 Min Symbol 设置为 character marker symbol。再创建 IproportionalSymbolRenderer 接口对象接收所有 Symbol 的设置信息并将数据分段。最后赋值给 IGeoFeatureLayer.Render 属性

● 程序说明

本程序的加载的数据是“states.shp”。点击 UIButtonControl1 程序开始运行。

过程 ProportionalSymbol 根据 pProportionalSymbolR 的属性值画出预定的 Symbol。背景的填充由 pFillSymbol.Color 调用函数 GetRGBColor 实现, Symbol 颜色由 pCharater

MarkerS.Color 调用函数 GetRGBColor 实现。

● 代码

```

Private Sub UIButtonControl1_Click()
'call sub ProportionalSymbol
ProportionalSymbol
End Sub

Private Sub ProportionalSymbol()
Dim pMxDocument As IMxDocument

```

```

Dim pMap As IMap
Dim pFeatureLayer As IFeatureLayer
Dim pGeoFeatureLayer As IGeoFeatureLayer
Dim pProportionalSymbolR As IProportionalSymbolRenderer
Dim pTable As ITable
Dim pQueryFilter As IQueryFilter
Dim pCursor As ICursor
Dim pFillSymbol As IFillSymbol
Dim pCharaterMarkerS As ICharacterMarkerSymbol
Dim pColor As IColor
Dim pOutlineColor As IColor
Dim pDataStatistics As IDataStatistics
Dim pStatisticsResult As IStatisticsResults
Dim pFontDisp As IFontDisp
Dim pRotationRenderer As IRotationRenderer

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap

'Get the first layer
Set pFeatureLayer = pMap.Layer(0)
Set pGeoFeatureLayer = pFeatureLayer

'QI the feature layer for the table interface
Set pTable = pGeoFeatureLayer
'Initialise a query and get a cursor
Set pQueryFilter = New QueryFilter
pQueryFilter.AddField ""

Set pCursor = pTable.Search(pQueryFilter, True)
'Use the statistics objects to calculate the max value
'and the min value
Set pDataStatistics = New DataStatistics
Set pDataStatistics.Cursor = pCursor
'Set statistical field
pDataStatistics.Field = "POP1990"
'Get the result of statistics
Set pStatisticsResult = pDataStatistics.Statistics

If pStatisticsResult Is Nothing Then
    MsgBox "Failed to gather stats on the feature class"
    Exit Sub
End If
' Set up the background fill color
Set pFillSymbol = New SimpleFillSymbol
pFillSymbol.Color = GetRGBColor(239, 228, 190)
'Set up the min symbol to a special character marker symbol
Set pCharaterMarkerS = New CharacterMarkerSymbol
Set pFontDisp = New StdFont
pFontDisp.Name = "ESRI Business"
pFontDisp.Size = 10
With pCharaterMarkerS
    .Font = pFontDisp
    .CharacterIndex = 95

```

```

        .Color = GetRGBColor(0, 0, 0)
        .Size = 4
    End With
    ' Create a new proportional symbol renderer to draw pop1990
    Set pProportionalSymbolR = New ProportionalSymbolRenderer
    With pProportionalSymbolR
        .ValueUnit = esriUnknownUnits
        .Field = "POP1990"
        .FlanneryCompensation = False
        .MinDataValue = pStatisticsResult.Minimum
        .MaxDataValue = pStatisticsResult.Maximum
        .BackgroundSymbol = pFillSymbol
        .MinSymbol = pCharaterMarkerS
        .LegendSymbolCount = 5
        .CreateLegendSymbols
    End With
    ' set rotation renderer by pop1990
    Set pRotationRenderer = pProportionalSymbolR
    pRotationRenderer.RotationField = "POP1990"
    pRotationRenderer.RotationType = esriRotateSymbolGeographic
    ' Set the cities layers renderer to the proportional
    ' symbol renderer and refresh the display
    Set pGeoFeatureLayer.Renderer = pProportionalSymbolR
    pMxDocument.ActiveView.Refresh
    pMxDocument.UpdateContents
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Public Function GetRGBColor(red As Integer, _
                             green As Integer, _
                             blue As Integer) As IColor
    Dim pColor As IColor
    Set pColor = New RgbColor
    pColor.RGB = RGB(red, green, blue)
    Set GetRGBColor = pColor
End Function

```

1.7.5. 如何为一个层设置 Chart Renderer

本例要实现的是为一个层设置 Chart Renderer, 使用一个 bar chart symbol 和一个 chart Renderer 来显示在每个 US State 人口的情况。用图解的方法说明 states 层的两个字段 Pop1990（紫色显示）和 Pop1999（绿色显示）之间人口的对比情况。

● 要点

首先实例化 IGeoFeatureLayer, 创建 IBarChartSymbol 接口对象, 设置 bar 的宽度; 创建 IMarkerSymbol 接口对象设置 Renderer; 创建 IChartSymbol 接口

对象找出最大值传递给 IMarkerSymbol 来设置 bar 的最大高度，创建 IFillSymbol 接口对象来设置填充的颜色，最后赋值给 IGeoFeatureLayer.Render 属性。

- 程序说明

本程序的加载的数据是“states.shp”。点击 UIButtonControl1 程序开始运行。

过程 BarChartRenderer 根据 pChartRenderer 的属性值画出预定的 Symbol。所有颜色的设置都由 pFillSymbol.Color 调用函数 GetRGBColor 完成。

- 代码

```
Private Sub UIButtonControl1_Click()  
    'call sub BarChartRenderer  
    BarChartRenderer  
End Sub  
  
Private Sub BarChartRenderer()  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pGeoFeatureL As IGeoFeatureLayer  
    Dim pChartRenderer As IChartRenderer  
    Dim pRendererFields As IRendererFields  
    Dim pTable As ITable  
    Dim pCursor As ICursor  
    Dim pQueryFilter As IQueryFilter  
    Dim pRowBuffer As IRowBuffer  
    ' Number of bars  
    Const numFields As Long = 2  
    Dim fieldIndecies(0 To numFields - 1) As Long  
    Dim lfieldIndex As Long  
    Dim dmaxValue As Double  
    Dim firstValue As Boolean  
    Dim dfieldValue As Double  
    Dim pBarChartSymbol As IBarChartSymbol  
    Dim pFillSymbol As IFillSymbol  
    Dim pMarkerSymbol As IMarkerSymbol  
    Dim pSymbolArray As ISymbolArray  
    Dim pChartSymbol As IChartSymbol  
    Const strPopField1 = "POP1990"  
    Const strPopField2 = "POP1999"  
  
On Error GoTo ErrorHandler:  
    Set pChartRenderer = New ChartRenderer  
    ' Set up the fields to draw charts of  
    Set pRendererFields = pChartRenderer  
    pRendererFields.AddField strPopField1  
    pRendererFields.FieldAlias(0) = pRendererFields.Field(0)  
    pRendererFields.AddField strPopField2  
    pRendererFields.FieldAlias(1) = pRendererFields.Field(1)  
    ' Calculate the max value of the data fields to allow the bar chart
```

```

' to scale the bars correctly
' Do this by looking through all the data fields of all the features
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pGeoFeatureL = pMap.Layer(0)
Set pTable = pGeoFeatureL
Set pQueryFilter = New QueryFilter
pQueryFilter.AddField strPopField1
pQueryFilter.AddField strPopField2
Set pCursor = pTable.Search(pQueryFilter, True)
' Make an array of the field numbers to iterate accross,
' this is to keep the code generic in the number of bars to draw.
fieldIndecies(0) = pTable.FindField(strPopField1)
fieldIndecies(1) = pTable.FindField(strPopField2)
firstValue = True
dmaxValue = 0
' Iterate across each feature
Set pRowBuffer = pCursor.NextRow
Do While Not pRowBuffer Is Nothing
    ' iterate through each data field and update the maxVal if needed
    For lfieldIndex = 0 To numFields - 1
        dfieldValue = pRowBuffer.Value(fieldIndecies(lfieldIndex))
        If firstValue Then
            ' Special case for the first value in a feature class
            dmaxValue = dfieldValue
            firstValue = False
        Else
            If dfieldValue > dmaxValue Then
                ' we've got a new biggest value
                dmaxValue = dfieldValue
            End If
        End If
    Next lfieldIndex
    Set pRowBuffer = pCursor.NextRow
Loop

If (dmaxValue <= 0) Then
    MsgBox "Failed to calculate the maximum value or max value is 0."
    Exit Sub
End If
' Set up the chart marker symbol to use with the renderer
Set pBarChartSymbol = New BarChartSymbol
Set pChartSymbol = pBarChartSymbol
pBarChartSymbol.Width = 6
Set pMarkerSymbol = pBarChartSymbol
' Finally we've got the biggest value, set this into the symbol
pChartSymbol.maxValue = dmaxValue
' This is the maximum height of the bars
pMarkerSymbol.Size = 15
' Now set up symbols for each bar
Set pSymbolArray = pBarChartSymbol
' Add some colours in for each bar
Set pFillSymbol = New SimpleFillSymbol
' This is a pastel purple

```



```

pFillSymbol.Color = GetRGBColor(213, 212, 252)
pSymbolArray.AddSymbol pFillSymbol
Set pFillSymbol = New SimpleFillSymbol
' This is a pastel green
pFillSymbol.Color = GetRGBColor(193, 252, 179)
pSymbolArray.AddSymbol pFillSymbol
' Now set the barchart symbol into the renderer
Set pChartRenderer.ChartSymbol = pBarChartSymbol
pChartRenderer.Label = "Population"
' set up the background symbol to use tan color
Set pFillSymbol = New SimpleFillSymbol
pFillSymbol.Color = GetRGBColor(239, 228, 190)
Set pChartRenderer.BaseSymbol = pFillSymbol
' Disable overposter so that charts appear in the centre of polygons
pChartRenderer.UseOverposter = False
' Update the renderer and refresh the screen
Set pGeoFeatureL.Renderer = pChartRenderer
pMxDocument.ActiveView.Refresh
pMxDocument.UpdateContents
Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

Private Function GetRGBColor(yourRed As Long, yourGreen As Long, _
                             yourBlue As Long) As IRgbColor
    Dim pRGB As IRgbColor
    Set pRGB = New RgbColor
    With pRGB
        .Red = yourRed
        .Green = yourGreen
        .Blue = yourBlue
        .UseWindowsDithering = True
    End With
    Set GetRGBColor = pRGB
End Function

```

1.7.6. 如何为一个层设置 DotDensity Renderer

本例要实现的是为一个层设置 DotDensity Renderer。一般情况下，用一个 Dot Density Fill Symbol 和一个 Dot Density Renderer 来描述层中某一特征的分布密度。本例中描述的是 states 层中人口的密度分布（字段 Pop1990），并且每个点表示 20000 人。

● 要点

首先实例化接口 IGeoFeatureLayer, 通过 DotDensityFillSymbol 类来创建 IDotDensityFillSymbol 接口对象来设置 Dot Density Symbol, 然后创建 ISimpleMarkerSymbol 接口对象, 将 Dot Density Symbol 的类型设置成为 marker

类型，再将 Symbol 赋给 IDotDensityRenderer 属性，最后赋值给 IGeoFeatureLayer.Render 属性

- 程序说明

本程序的加载的数据是“states.shp”。点击 UIButtonControl1 程序开始运行。

过程 DotDensityRenderer 根据 pDotDensityRenderer 的属性值画出预定的 Symbol。所有颜色设置都调用函数 GetRGBColor 传入过程。

- 代码

```
Private Sub UIButtonControl1_Click()  
    'call sub DotDensityRenderer  
    DotDensityRenderer  
End Sub  
  
Private Sub DotDensityRenderer()  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pGeoFeatureL As IGeoFeatureLayer  
    Dim pDotDensityRenderer As IDotDensityRenderer  
    Dim pDotDensityFillS As IDotDensityFillSymbol  
    Dim pRendererFields As IRendererFields  
    Dim pSymbolArray As ISymbolArray  
    Dim pSimpleMarkerS As ISimpleMarkerSymbol  
    Const strPopField = "POP1990" ' Population fields  
  
On Error GoTo ErrorHandler:  
    Set pMxDocument = ThisDocument  
    Set pMap = pMxDocument.FocusMap  
    Set pGeoFeatureL = pMap.Layer(0)  
    Set pDotDensityRenderer = New DotDensityRenderer  
  
    ' Set up the fields to draw charts of  
    Set pRendererFields = pDotDensityRenderer  
    pRendererFields.AddField strPopField  
    ' Set up dot density symbol  
    Set pDotDensityFillS = New DotDensityFillSymbol  
    pDotDensityFillS.DotSize = 3  
    pDotDensityFillS.Color = GetRGBColor(0, 0, 0)  
    ' color of tan  
    pDotDensityFillS.BackgroundColor = GetRGBColor(239, 228, 190)  
  
    ' Put one marker type into the dot density symbol  
    Set pSymbolArray = pDotDensityFillS  
    Set pSimpleMarkerS = New SimpleMarkerSymbol  
    pSimpleMarkerS.Style = esriSMSCircle  
    pSimpleMarkerS.Size = 3  
    pSimpleMarkerS.Color = GetRGBColor(0, 0, 0) ' Black  
    pSymbolArray.AddSymbol pSimpleMarkerS  
    Set pDotDensityRenderer.DotDensitySymbol = pDotDensityFillS
```

```

' This relates to the number of dots per polygon,
' this value works for the US population
pDotDensityRenderer.DotValue = 200000
' Update the renderer and refresh the screen
Set pGeoFeatureL.Renderer = pDotDensityRenderer
pMxDocument.ActiveView.Refresh
Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

Private Function GetRGBColor(yourRed As Long, yourGreen As Long, _
                             yourBlue As Long) As IRgbColor
    Dim pRGB As IRgbColor
    Set pRGB = New RgbColor
    With pRGB
        .Red = yourRed
        .Green = yourGreen
        .Blue = yourBlue
        .UseWindowsDithering = True
    End With
    Set GetRGBColor = pRGB
End Function

```

1.8. Layout 和打印

1.8.1. 如何在 Page Layout 上添加 Text

本例要实现的功能是根据鼠标在 Page Layout 上点击的位置添加 Text 元素。

● 要点

要实现本例的功能，首先需要在 PageLayout 上创建一个 Text 元素，然后再设置该元素的属性。其中主要使用了两个接口 ITextElement 和 IGraphicsContainer。

ITextElement 接口是用来控制 Text 元素，以下是它的几个主要属性：

- ScaleText: BOOL 型，表示地图比例尺变化时 Text 大小是否变化；
- Symbol: 用来设置 Text 元素的风格；
- Text: 用来设置 Text 元素的内容。

IGraphicsContainer 是用来控制 PageLayout, Map 等对象上图形元素的接口。

以下是它的几个主要属性和方法：

- AddElement: 向层中增加一个元素；
- DeleteAllElements: 删除所有的元素；
- FindFrame: 查找可以放在该容器中的某对象，例如 Text 元素；
- Next: 返回该容器中的下一个对象；
- UpdateElement: 更新某个元素。。

● 程序说明

函数 AddTextToLayout 根据鼠标点击的位置点 (x, y) 在 PageLayout 上添加一个文本元素。

- 代码

```
Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)
    AddTextToLayout x, y
End Sub

Sub AddTextToLayout(x As Long, y As Long)
    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pGraphicsContainer As IGraphicsContainer
    Dim pTextElement As ITextElement
    Dim pElement As IElement

    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    ' 确保 ArcMap 在 layout 模式下
    If Not pMxDocument.ActiveView Is pMxDocument.PageLayout Then
        Exit Sub
    End If
    Set pActiveView = pMxDocument.PageLayout
    Set pGraphicsContainer = pMxDocument.PageLayout

    Set pTextElement = New TextElement
    Set pElement = pTextElement
    ' 设置 Text 的内容
    pTextElement.Text = "My Map"
    ' 将元素的图形定位在点 (x, y) 处
    pElement.Geometry = pActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
    ' 向 PageLayout 中添加一个元素
    pGraphicsContainer.AddElement pTextElement, 0
    ' 刷新
    pMxDocument.ActiveView.PartialRefresh esriViewGraphics, Nothing, Nothing
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub
```

1.8.2. 如何在 Page Layout 上添加 Legend

本例要实现的功能是以鼠标在 Page Layout 上画的 Envelope 为范围在 PageLayout 增加一个跟第一个层相关联的图例。

- 要点

要实现本例的功能首先需要在 PageLayout 上创建一个 Legend 元素，然后再设置该元素的属性，其中用到了两个主要的接口：ILegend 和 IlegendItem。

ILegend 用来控制 Legend（图例）。以下是该接口成员的介绍：

Layer: 实现与相关层的关联;
Columns: 图例以几列显示;
ShowDescription、ShowHeading、ShowLabels、ShowLayerName: 分别表示描述、标题、分类、层名称是否显示;
IlegendItem 用来设置 Legend 的风格。以下是该接口成员的介绍:
AddItem : 在图例的最后添加一项;
ClearItem: 清除所有项;
Title: 设置标题。

● 程序说明

函数 CreateLegend 根据传入的 pExtent 参数在 PageLayout 上添加一个 Legend 元素。

● 代码

```
Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x
As Long, _
    ByVal y As Long)
    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pEnvelope As IEnvelope
    Dim pFeatureLayer As IFeatureLayer
    Dim pRubberBand As IRubberBand
    On Error GoTo ErrorHandler

    Set pMxDocument = ThisDocument
    ' 确保 AcrMap 在 Layout 模式下
    ' 确保 AcrMap 中有数据
    If Not pMxDocument.ActiveView Is pMxDocument.PageLayout Or
pMxDocument.FocusMap.LayerCount = 0 Then
        Exit Sub
    End If
    ' 初始设定
    Set pActiveView = pMxDocument.PageLayout
    Set pRubberBand = New RubberEnvelope
    ' IRubberBand 接口用于画 Envelope, Polygon 等
    Set pEnvelope = pRubberBand.TrackNew(pMxDocument.ActiveView.ScreenDisplay, Nothing)
    Set pFeatureLayer = pMxDocument.FocusMap.Layer(0)

    CreateLegend pEnvelope, pFeatureLayer, pActiveView
    pActiveView.Refresh
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Public Sub CreateLegend(pExtent As IEnvelope, pFeatureLayer As IFeatureLayer,
    pActiveView As IActiveView)
    Dim pMapFrame As IMapFrame
    Dim pMapSurroundF As IMapSurroundFrame
    Dim pMapSurround As IMapSurround
    Dim pLegend As ILegend
    Dim pLegendItem As ILegendItem
```

```

Dim pElement As IElement
Dim pAreaLayer As IFeatureLayer
Dim pTextSymbol As ITextSymbol
Dim pFillSymbol As IFillSymbol
Dim pLineSymbol As ILineSymbol
Dim pColor As IColor
Dim pSymbolBackground As ISymbolBackground
Dim pUID As New UID

On Error GoTo ErrorHandler

If pFeatureLayer Is Nothing Then Exit Sub
If pActiveView Is Nothing Then Exit Sub
If Not TypeOf pActiveView Is IPageLayout Then Exit Sub
' 得到 MapFrame
Set pMapFrame = pActiveView.GraphicsContainer.FindFrame(pActiveView.FocusMa )
pUID.Value = "esriCore.Legend"
Set pMapSurroundF = pMapFrame.CreateSurroundFrame(pUID, Nothing)
' 创建底图 Symbol
Set pSymbolBackground = New esriCore.SymbolBackground
Set pFillSymbol = New esriCore.SimpleFillSymbol
Set pLineSymbol = New esriCore.SimpleLineSymbol
Set pColor = New esriCore.RgbColor
pColor.RGB = RGB(255, 255, 255)
pLineSymbol.Color = pColor
pFillSymbol.Color = pColor
pFillSymbol.Outline = pLineSymbol
pSymbolBackground.FillSymbol = pFillSymbol
pMapSurroundF.Background = pSymbolBackground
Set pElement = pMapSurroundF
pElement.Geometry = pExtent
Set pMapSurround = pMapSurroundF.MapSurround
Set pLegend = pMapSurround
' 创建一个水平的 LegendItem
Set pLegendItem = New esriCore.HorizontalLegendItem
' 设置 LegendItem 的相关层和列数
With pLegendItem
    Set .Layer = pFeatureLayer
    .Columns = 1
    .ShowDescriptions = True
    .ShowHeading = True
    .ShowLabels = True
    .ShowLayerName = True
End With
' 先清除所有的 LegendItem
pLegend.ClearItems
' 在 Legend 上添加一个 LegendItem
With pLegend
    .AddItem pLegendItem
    .Title = "New Legend"
End With
pActiveView.GraphicsContainer.AddElement pElement, 0
Exit Sub
ErrorHandler:
MsgBox Err.Description

```

End Sub

1.8.3. 如何在 Page Layout 上添加 North Arrow

本例要实现的功能是以鼠标在 Page Layout 上画的 Envelope 为范围在 PageLayout 上增加指北针。

● 要点

要实现本例的功能首先要在 PageLayout 上创建一个 North Arrow 元素，然后再设置该元素的属性。其中用到了两个主要接口：IMarkerNorthArrow 和 ICharacterMarkerSymbol。

IMarkerNorthArrow 用来控制 NorthArrow。以下是该接口主要的属性介绍：

MarkerSymbol：设置 North Arrow 的风格

ICharacterMarkerSymbol 用来设置特征标志的风格。以下是该接口主要的属性介绍：

Size：该标志的大小。

CharacterIndex：Long 型，该标志使用风格的索引。

● 程序说明

函数 CreateNorthArrow 根据传入的 pExtent 参数在 PageLayout 上添加一个 NorthArrow 元素。

● 代码

```
Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _  
    ByVal y As Long)  
    Dim pMxDocument As IMxDocument  
    Dim pActiveView As IActiveView  
    Dim pEnvelope As IEnvelope  
    Dim pRubberBand As IRubberBand  
  
    On Error GoTo ErrorHandler  
    Set pMxDocument = ThisDocument  
    ' 确保 AcrMap 在 Layout 模式下  
    If Not pMxDocument.ActiveView Is pMxDocument.PageLayout Then  
        Exit Sub  
    End If  
    ' 初始设定  
    Set pActiveView = pMxDocument.PageLayout  
    Set pRubberBand = New RubberEnvelope  
    Set pEnvelope = pRubberBand.TrackNew(pMxDocument.ActiveView.ScreenDisplay, Nothing)  
  
    CreateNorthArrow pEnvelope, pActiveView  
  
    pActiveView.Refresh  
    Exit Sub  
ErrorHandler:  
    MsgBox Err.Description  
End Sub
```

```

Public Sub CreateNorthArrow(pExtent As IEnvelope, pActiveView As IActiveView)
    Dim pMapFrame As IMapFrame
    Dim pMapSurroundF As IMapSurroundFrame
    Dim pMapSurround As IMapSurround
    Dim pElement As IElement
    Dim pMarkerNorthA As IMarkerNorthArrow
    Dim pCharacterMarkerS As ICharacterMarkerSymbol
    Dim pUID As New UID

    On Error GoTo ErrorHandler
    If Not TypeOf pActiveView Is IPageLayout Then Exit Sub
    Set pMapFrame = pActiveView.GraphicsContainer.FindFrame(pActiveView.FocusM p)
    pUID.Value = "esriCore.MarkerNorthArrow"
    ' 根据 UID 创建 North Arrow
    Set pMapSurroundF = pMapFrame.CreateSurroundFrame(pUID, Nothing)
    Set pElement = pMapSurroundF
    pElement.Geometry = pExtent

    Set pMapSurround = pMapSurroundF.MapSurround
    ' 得到创建的 North Arrow
    Set pMarkerNorthA = pMapSurround
    Set pCharacterMarkerS = pMarkerNorthA.MarkerSymbol
    ' 设置 North Arrow 的 Size
    pCharacterMarkerS.Size = 40
    ' 设置 North Arrow 的特征值，即显示风格
    pCharacterMarkerS.CharacterIndex = 173
    pMarkerNorthA.MarkerSymbol = pCharacterMarkerS
    pActiveView.GraphicsContainer.AddElement pElement, 0
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.8.4. 如何在 Page Layout 上添加 Scale bar

本例要实现的功能是以鼠标在 Page Layout 上画的 Envelope 为范围在 PageLayout 上添加 Scale Bar。

● 要点

首先需要在 PageLayout 上创建一个 Scale Bar 元素，然后再设置该元素的属性。要实现此功能，用到了接口 IScaleBar，用来控制 ScaleBar。以下是该接口的属性介绍：

Division: 设置比例尺的分割单位；
 DivisionsBeforeZero: 设置比例尺原点左侧显示的段数；
 Divisions: 设置比例尺的总段数（包括原点左侧的段数）；
 Subdivisions: 设置主比例尺分为几个子段；
 Units: 设置比例尺的单位；
 UnitLabel: String 型，设置单位标签上的内容；
 UnitLabelPosition: 设置单位标签显示的位置；
 LabelPosition: 比例尺数字标签的显示位置；

LabelFrequency: 比例尺数字标签的风格。

- 程序说明

函数 CreateScaleBar 根据传入的 pExtent 参数在 PageLayout 上添加一个 ScaleBar 元素。

- 代码

```
Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)
    Dim pMxDocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pEnvelope As IEnvelope
    Dim pRubberBand As IRubberBand

    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    ' 确保 AcrMap 在 Layout 模式下
    If Not pMxDocument.ActiveView Is pMxDocument.PageLayout Then
        Exit Sub
    End If
    ' 初始设定
    Set pActiveView = pMxDocument.PageLayout
    Set pRubberBand = New RubberEnvelope
    Set pEnvelope = pRubberBand.TrackNew(pMxDocument.ActiveView.ScreenDisplay, Nothing)

    CreateScaleBar pEnvelope, pActiveView

    pActiveView.Refresh
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Public Sub CreateScaleBar(pExtent As IEnvelope, pActiveView As IActiveView)
    Dim pMapFrame As IMapFrame
    Dim pMapSurroundF As IMapSurroundFrame
    Dim pMapSurround As IMapSurround
    Dim pElement As IElement
    Dim pFillSymbol As IFillSymbol
    Dim pLineSymbol As ILineSymbol
    Dim pColor As IColor
    Dim pSymbolBackground As ISymbolBackground
    Dim pScaleBar As IScaleBar
    Dim pUID As New UID

    On Error GoTo ErrorHandler
    If Not TypeOf pActiveView Is IPageLayout Then
        Exit Sub
    End If
    Set pMapFrame = pActiveView.GraphicsContainer.FindFrame(pActiveView.FocusMap)
    pUID.Value = "esriCore.ScaleBar"
    ' 根据 UID 创建 Scale Bar
```

```

Set pMapSurroundF = pMapFrame.CreateSurroundFrame(pUID, Nothing)

Set pSymbolBackground = New esriCore.SymbolBackground
Set pFillSymbol = New esriCore.SimpleFillSymbol
Set pLineSymbol = New esriCore.SimpleLineSymbol
Set pColor = New esriCore.RgbColor
pColor.RGB = RGB(255, 255, 255)
pLineSymbol.Color = pColor
pFillSymbol.Color = pColor
pFillSymbol.Outline = pLineSymbol
pSymbolBackground.FillSymbol = pFillSymbol
pMapSurroundF.Background = pSymbolBackground

Set pElement = pMapSurroundF
pElement.Geometry = pExtent

Set pMapSurround = pMapSurroundF.MapSurround
' 创建一个 single alternating scale bar
Set pScaleBar = New esriCore.AlternatingScaleBar
Set pMapSurround = pScaleBar
' 设置 Scale Bar 的属性，如分割单位、数字显示的位置、label 和 label 显示位置等
With pScaleBar
    .Division = 100
    .DivisionsBeforeZero = 0
    .Divisions = 2
    .Subdivisions = 4
    .Units = esriMeters
    .UnitLabel = "m"
    .UnitLabelPosition = esriScaleBarAfterBar
    .LabelPosition = esriAbove
    .LabelFrequency = esriScaleBarDivisionsAndFirstMidpoint
End With

Set pMapSurroundF.MapSurround = pMapSurround
Set pElement = pMapSurroundF
pActiveView.GraphicsContainer.AddElement pElement, 0
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.8.5. 如何在 Page Layout 上添加 Scale Text

本例要实现的功能是以鼠标在 Page Layout 上画的 Envelope 为范围在 PageLayout 上添加 Scale Text。

● 要点

本例中，首先需要在 PageLayout 上创建一个 Scale Text 元素，然后再设置该元素的属性。实现此功能，用到了接口 IScaleText，该接口是用来控制 Scale Text，下面是接口属性的介绍：

Symbol：设置 Scale Text 的 Text 的属性

Style：设置 Scale Text 风格。Scale Text 有两种风格：

esriScaleTextAbsolute(例: 1: 5000)和 esriScaleTextRelative
(例: 1 inch equals 800 miles)

- 程序说明

函数 CreateScaleText 根据传入的 pExtent 参数在 PageLayout 上添加一个 Scale Text 元素。

- 代码

```
Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x
                                     As Long, _ ByVal y As Long)

    Dim pMxDocument          As IMxDocument
    Dim pActiveView           As IActiveView
    Dim pEnvelope             As IEnvelope
    Dim pRubberBand           As IRubberBand

    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    ' 确保 AcrMap 在 Layout 模式下
    If Not pMxDocument.ActiveView Is pMxDocument.PageLayout Then
        Exit Sub
    End If
    ' 初始设定
    Set pActiveView = pMxDocument.PageLayout
    Set pRubberBand = New RubberEnvelope
    Set pEnvelope = pRubberBand.TrackNew(pMxDocument.ActiveView.ScreenDisplay, Nothing)

    CreateScaleText pEnvelope, pActiveView

    pActiveView.Refresh
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Public Sub CreateScaleText(pExtent As IEnvelope, pActiveView As IActiveView)

    Dim pMapFrame             As IMapFrame
    Dim pMapSurroundF         As IMapSurroundFrame
    Dim pMapSurround          As IMapSurround
    Dim pElement              As IElement
    Dim pScaleText            As IScaleText
    Dim pTextSymbol           As ITextSymbol
    Dim pUID                  As New UID

    On Error GoTo ErrorHandler
    If Not TypeOf pActiveView Is IPageLayout Then Exit Sub
    Set pMapFrame = pActiveView.GraphicsContainer.FindFrame(pActiveView.FocusMap)
    pUID.Value = "esriCore.ScaleText"
    ' 根据 UID 创建 Scale Text
    Set pMapSurroundF = pMapFrame.CreateSurroundFrame(pUID, Nothing)
    Set pElement = pMapSurroundF
    pElement.Geometry = pExtent

    Set pMapSurround = pMapSurroundF.MapSurround
```

```

Set pScaleText = pMapSurround
Set pTextSymbol = New esriCore.TextSymbol
pTextSymbol.Size = 20
' 设置 Scale Text 的 Symbol 属性
pScaleText.Symbol = pTextSymbol
' 设置 Scale Text 的 Style
pScaleText.Style = esriScaleTextRelative
pActiveView.GraphicsContainer.AddElement pElement, 0
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1.8.6. 如何在 Page Layout 上添加 Picture

本例要实现的功能以鼠标在 Page Layout 上画的 Envelope 为范围在 PageLayout 上增加 Picture。

● 要点

本例中，首先需要在 PageLayout 上创建一个 Picture 元素，然后再设置该元素的属性。实现此功能，用到了 IPictureElement 接口。以下是该接口中主要属性和方法的介绍：

ImportPictureFromFile: 需要图片路径；

MaintainAspectRatio: 是否保证图片的纵横比。

● 程序说明

函数 CreatePicture 根据传入的 pExtent 参数在 PageLayout 上添加一个图片，该图片是存放在 mxd 文件所在目录下的 arcgisbook.bmp 文件。

● 代码

```

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, ByVal x As Long, _
    ByVal y As Long)
    Dim pMxdocument As IMxDocument
    Dim pActiveView As IActiveView
    Dim pEnvelope As IEnvelope
    Dim pRubberBand As IRubberBand

    On Error GoTo ErrorHandler
    Set pMxdocument = ThisDocument
    ' 确保 AcrMap 在 Layout 模式下
    If Not pMxdocument.ActiveView Is pMxdocument.PageLayout Then
        Exit Sub
    End If
    ' 初始设定
    Set pActiveView = pMxdocument.PageLayout
    Set pRubberBand = New RubberEnvelope
    Set pEnvelope = pRubberBand.TrackNew(pMxdocument.ActiveView.ScreenDisplay, Nothing)
    CreatePicture pEnvelope, pActiveView
    pActiveView.Refresh

```

```

Exit Sub
ErrorHandler:
MsgBox Err.Description
End Sub

Sub CreatePicture(pExtent As IEnvelope, pActiveView As IActiveView)
Dim pElement As IElement
Dim pPictureElement As IPictureElement
Dim pVBProject As VBProject
Dim sPicPath As String

On Error GoTo ErrorHandler
Set pPictureElement = New BmpPictureElement
' 得到 mxd 文件的路径
Set pVBProject = ThisDocument.VBProject
sPicPath = pVBProject.FileName
sPicPath = Left(sPicPath, InStrRev(sPicPath, "\"))
' 设置图片路径
pPictureElement.ImportPictureFromFile sPicPath & "arcgisbook.bmp"
' 保持纵横比
pPictureElement.MaintainAspectRatio = False
Set pElement = pPictureElement
pElement.Geometry = pExtent
pActiveView.GraphicsContainer.AddElement pElement, 0
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1.8.7. 如何创建、删除地图网格 (Map Grid)

本例主要实现 Layerout 中 Map Grid 的创建和删除

● 要点

通过 IMapGridFactory 接口可创建 Map Grid 并对其命名。然后运用 IMapGrids 和 IMapGrid 接口来实现对 Map Grid 的添加，删除等操作。IMapGrids 只能被 MapFrame 这个对象来实现。通过这个接口，可以对一个具体的 MapFrame 所展示的网格进行接收和设置。IMapGrid 是个可以对所有类型网格 (Grid) 的属性进行设置的接口，四种类型的 Grid 类实现了 IMapGrids 接口。它们是 IMeasuredGrid, IGraticule, IndexGrid, ICustomGridOverlay。

● 程序说明

在宏里定义了两个模块：Add 和 Delete。Add 模块里定义了一个过程 CreateMapGrid(), 是用来创建一个当前 MapFrame 的 IndexGrid。Delete 模块里定义了一个过程 DeleteMapGrids(), 用来删除当前 MapFrame 里的所有 Grid。运

用时，先在 ArcMap 主菜单中单击 View，然后选择 Layout View。接着在宏里先运行 Add 模块，可以看到在 ArcMap 主窗口中创建了一个网格。要删除该网格，运行 Delete 模块即可。

- 代码

```
Private Sub CreateMapGrid()  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pGraphicsContainer As IGraphicsContainer  
    Dim pMapFrame As IMapFrame  
    Dim pActiveView As IActiveView  
    Dim pMapGrids As IMapGrids  
    Dim pMapGrid As IMapGrid  
    Dim pMapGridFactory As IMapGridFactory  
  
    On Error GoTo ErrorHandler  
    Set pMxDocument = ThisDocument  
    Set pMap = pMxDocument.FocusMap  
    Set pGraphicsContainer = pMxDocument.PageLayout  
    Set pMapFrame = pGraphicsContainer.FindFrame(pMap)  
  
    ' Create map grid  
    ' 创建类型为 IndexGrid 的 Map Grid  
    Set pMapGridFactory = New IndexGridFactory  
    ' 还可创建其他类型，如下  
    '' Set pMapGridFactory = New GraticuleFactory  
    '' Set pMapGridFactory = New MeasuredGridFactory  
  
    ' 创建一个 Map Grid  
    Set pMapGrid = pMapGridFactory.Create(pMapFrame)  
  
    ' 将 Map Grid 添加到当前的 MapFrame 中，并刷新  
    Set pMapGrids = pMapFrame  
    pMapGrids.AddMapGrid pMapGrid  
    Set pActiveView = pMxDocument.PageLayout  
    pActiveView.PartialRefresh esriViewBackground, Nothing, Nothing  
    Exit Sub  
  
ErrorHandler:  
    MsgBox Err.Description  
End Sub  
  
Public Sub DeleteMapGrids()  
  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pActiveView As IActiveView  
    Dim pGraphicsContainer As IGraphicsContainer  
    Dim pMapFrame As IMapFrame  
    Dim pMapGrids As IMapGrids
```

```

Dim pMapGrid          As IMapGrid
Dim i                  As Long
Dim count              As Long

On Error GoTo ErrorHandler:
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pActiveView = pMxDocument.PageLayout
Set pGraphicsContainer = pMxDocument.PageLayout
Set pMapFrame = pGraphicsContainer.FindFrame(pMap)
Set pMapGrids = pMapFrame

' 通过循环操作删除所有的 Map Grid 并且刷新
count = pMapGrids.MapGridCount
For i = count - 1 To 0 Step -1
    ' if you remove grid(0) first, then grid(1) becomes grid(0)
    Set pMapGrid = pMapGrids.MapGrid(i)
    pMapGrids.DeleteMapGrid pMapGrid
    Set pMapGrid = Nothing
Next i
pActiveView.PartialRefresh esriViewBackground, Nothing, Nothing
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1.8.8. 如何设置 Layout 中 MapFrame 的外观风格属性

本例要完成的功能是设置 Page Layout 中 MapFrame 的边框（Border）、背景（Background）和阴影（Shadow）等属性。

● 要点

本例主要运用 IBorder、IBackground、IShadow 等接口对 MapFrame 的 Border, Background, Shadow 等属性的颜色和宽度进行设置，并且将其在 Page Layout 中显示出来。

● 程序说明

通过 IMapFrame 的 Border 属性设置 Border 的颜色和宽度；通过对 IMapFrame 的 Background 属性设置 Background 的颜色；通过 IFrameProperties 的 Shadow 属性设置 MapFrame 的 Shadow 颜色。

● 代码

```

Private Sub cmdDraw_Click()
    Dim pMxDocument      As IMxDocument
    Dim pMap              As IMap
    Dim pGraphicsContainer As IGraphicsContainer

```

```

Dim pMapFrame          As IMapFrame
Dim pLineSymbol        As ILineSymbol
Dim pColor             As IColor
Dim pRgbcolor          As IRgbColor
Dim pBorder            As IBorder
Dim pSymbolborder      As ISymbolBorder
Dim pBackGround        As IBackground
Dim pSymbolBackG       As ISymbolBackground
Dim pFillSymbol        As IFillSymbol
Dim pShadow            As IShadow
Dim pSymbolShadow      As ISymbolShadow
Dim pFrameProperties    As IFrameProperties

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pGraphicsContainer = pMxDocument.PageLayout
Set pMapFrame = pGraphicsContainer.FindFrame(pMap)
Set pFrameProperties = pMapFrame

' 对用户输入的 Border 数据进行检测
If txtBorderRed.Value >= 0 And txtBorderRed.Value <= 255 And _
   txtBorderBlue.Value >= 0 And txtBorderBlue.Value <= 255 And _
   txtBorderGreen.Value >= 0 And txtBorderGreen.Value <= 255 And _
   txtBorderWidth.Value > 0 Then
    ' 设置 Border 的颜色和宽度
    Set pRgbcolor = New RgbColor
    pRgbcolor.Red = txtBorderRed.Value
    pRgbcolor.Blue = txtBorderBlue.Value
    pRgbcolor.Green = txtBorderGreen.Value
    Set pColor = pRgbcolor

    Set pLineSymbol = New SimpleLineSymbol
    pLineSymbol.Color = pColor
    pLineSymbol.Width = txtBorderWidth.Value

    Set pSymbolborder = New SymbolBorder
    pSymbolborder.LineSymbol = pLineSymbol
    Set pBorder = pSymbolborder
    pMapFrame.Border = pBorder
Else
    MsgBox "You input a wrong number in border set!"
End If
' 对用户输入的 Background 数据进行检测
If txtBackRed.Value >= 0 And txtBackRed.Value <= 255 And _
   txtBackBlue.Value >= 0 And txtBackBlue.Value <= 255 And _
   txtBackGreen.Value >= 0 And txtBackGreen.Value <= 255 Then
    ' 设置 BackGround 的颜色
    Set pRgbcolor = New RgbColor
    pRgbcolor.Red = txtBackRed.Value
    pRgbcolor.Blue = txtBackBlue.Value
    pRgbcolor.Green = txtBackGreen.Value
    Set pColor = pRgbcolor

```



```

Set pFillsymbol = New SimpleFillSymbol
pFillsymbol.Color = pColor

Set pSymbolBackG = New SymbolBackground
pSymbolBackG.FillSymbol = pFillsymbol
Set pBackground = pSymbolBackG
pMapFrame.Background = pBackground
Else
    MsgBox "You input a wrong number in Background Set!"
End If

' 对用户输入的 Shadow 数据进行检测
If txtShadowRed.Value >= 0 And txtShadowRed.Value <= 255 And _
txtShadowBlue.Value >= 0 And txtShadowBlue.Value <= 255 And _
txtShadowGreen.Value >= 0 And txtShadowGreen.Value <= 255 Then
    ' 设置 Shadow 的颜色
    Set pRgbcolor = New RgbColor
    pRgbcolor.Red = txtShadowRed.Value
    pRgbcolor.Blue = txtShadowBlue.Value
    pRgbcolor.Green = txtShadowGreen.Value
    Set pColor = pRgbcolor
    Set pFillsymbol = New SimpleFillSymbol
    pFillsymbol.Color = pColor

    Set pSymbolShadow = New SymbolShadow
    pSymbolShadow.FillSymbol = pFillsymbol
    Set pShadow = pSymbolShadow
    pFrameProperties.Shadow = pShadow
Else
    MsgBox "You input a wrong number in Shadow Set!"
End If
' 刷新 Page Layout
pMxDocument.ActiveView.Refresh
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1.8.9. 何设置 Layout 中 Page 的边框（Border）和背景（Background）

本例要完成的功能是设置 Page Layout 中 Page 的边框（Border）和背景（Background）属性。

● 要点

IStyleGallery.items：根据不同的参数可以得到系统中所有的 Border, Background, Shadow 等的风格属性。

IPage.Border: 设置Page的边框式样。

IPage.Background: 设置Page的背景式样。

- 程序说明

先用 IStyleGallery.items 得到系统中所有的 Border, Background 式样的属性值, 把它们分别放在对应的 ListBox 中; 然后再把选择的 Border 或 Background 属性值赋给 IPage.Border 或 IPage.Background, 这样就可以实现设置 Page 的边框或背景的效果。

- 代码

```
' 模块变量的定义:
Option Explicit
Dim m_pStylesArray(0 To 2) As IArray
Dim m_pStyleGallery As IstyleGallery

' 设置 Background 的过程
Private Sub lstBackground_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
    Dim pSymbol As IBackground
    Dim pMxDocument As IMxDocument
    Dim pPageLayout As IPageLayout
    Dim pPage As IPage

    On Error GoTo ErrorHandler
    Set pSymbol = m_pStylesArray(1).Element(lstBackground.ListIndex)
    Set pMxDocument = ThisDocument
    Set pPageLayout = pMxDocument.PageLayout
    Set pPage = pPageLayout.Page
    pPage.Background = pSymbol
    pMxDocument.ActivatedView.Refresh
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

' 设置 Border 的过程
Private Sub lstBorder_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
    Dim pSymbol As IBorder
    Dim pMxDocument As IMxDocument
    Dim pPageLayout As IPageLayout
    Dim pPage As IPage

    On Error GoTo ErrorHandler
    Set pSymbol = m_pStylesArray(0).Element(lstBorder.ListIndex)
    Set pMxDocument = ThisDocument
    Set pPageLayout = pMxDocument.PageLayout
    Set pPage = pPageLayout.Page
    pPage.Border = pSymbol
```

```

' 设置 Shadow 的 Sub
pMxDocument.ActivatedView.Refresh
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UserForm_Initialize()
    UpdateArrayAndListBoxFromStyleGallery "Borders", m_pStylesArray(0), lstBorder
    UpdateArrayAndListBoxFromStyleGallery "Backgrounds", m_pStylesArray(1),
lstBackground
    UpdateArrayAndListBoxFromStyleGallery "Shadows", m_pStylesArray(2), lstShadow
End Sub

Public Sub UpdateArrayAndListBoxFromStyleGallery(styleClass As String, & _
                                                ByRef pArray As IArray, & _
                                                ByRef pListBox As ListBox)

    'Get IStyleGalleryClass interface
    Dim pStyleClass As IStyleGalleryClass
    Dim pEnumStyleGallery As IEnumStyleGalleryItem
    Dim pStyleItem As IStyleGalleryItem

    On Error GoTo ErrorHandler:
    'If the StyleGallery hasn't been used before
    If m_pStyleGallery Is Nothing Then
        Set m_pStyleGallery = New StyleGallery
    End If
    'Get IEnumStyleGalleryItem interface and retrieve all styles within the class
    Set pEnumStyleGallery = m_pStyleGallery.Items(styleClass, "ESRI.style", ""
    pEnumStyleGallery.Reset
    'Create a new array
    Set pArray = New esriCore.Array
    'Clear out the list box
    pListBox.Clear
    'Get IStyleGalleryItem interface
    Set pStyleItem = pEnumStyleGallery.Next
    'Loop through the style gallery items
    Do While Not pStyleItem Is Nothing
        'Add the style to the array
        pArray.Add pStyleItem.Item
        'Add the style name to the list box
        pListBox.AddItem pStyleItem.Name
        Set pStyleItem = pEnumStyleGallery.Next
    Loop

    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControl1_Click()

```

```
UserForm1.Show
End Sub
```

1.8.10. 如何设置打印纸张的大小和方向

本例主要运用 IPageLayout 这个接口来对打印的纸张进行设置，包括用纸的大小和横竖方向。

● 要点

本例中运用 IPage 中的 FormID 属性对纸张的大小进行设置，运用 Orientation 属性对纸张的打印方向（水平或垂直）进行设置。

● 程序说明

将所有的纸张大小和方向分别插入到两个 ComboBox 中，然后根据用户不同的选择对打印纸张进行不同的设置。

● 代码

初始化窗口菜单

```
Private Sub UserForm_Initialize()
    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    Set pPageLayout = pMxDocument.PageLayout
    ' 设置纸张大小, 将所有的类型插入到 ComboBox 中
    Me.cmbSize.AddItem "Letter - 8.5in x 11in"
    Me.cmbSize.AddItem "Legal - 8.5in x 14in"
    Me.cmbSize.AddItem "Tabloid - 11in x 17in"
    Me.cmbSize.AddItem "C - 17in x 22in"
    Me.cmbSize.AddItem "D - 22in x 34in"
    Me.cmbSize.AddItem "E - 34in x 44in"
    Me.cmbSize.AddItem "Metric A5 - 148mm x 210mm"
    Me.cmbSize.AddItem "Metric A4 - 210mm x 297mm"
    Me.cmbSize.AddItem "Metric A3 - 297mm x 420mm"
    Me.cmbSize.AddItem "Metric A2 - 420mm x 594mm"
    Me.cmbSize.AddItem "Metric A1 - 594mm x 841mm"
    Me.cmbSize.AddItem "Metric A0 - 841mm x 1189mm"
    Me.cmbSize.AddItem "Custom Page Size"
    Me.cmbSize.AddItem "Page Form same as Printer Form"

    ' 设置纸张方向, 并将其类型插入到 ComboBox 中
    Me.cmbOri.AddItem "Portrait"
    Me.cmbOri.AddItem "Landscape"
    ' 将当前纸张的打印大小和方向在两个 ComboBox 中显示
    Me.cmbSize.ListIndex = pPageLayout.Page.FormID
    Me.cmbOri.ListIndex = pPageLayout.Page.Orientation - 1
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
```

```

End Sub

' 点击 Set 按钮所触发的事件
Private Sub cmdSet_Click()
    On Error GoTo ErrorHandler

    pPageLayout.Page.FormID = Me.cmbSize.ListIndex
    pPageLayout.Page.Orientation = Me.cmbOri.ListIndex + 1
    Unload Me
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.9. 坐标系统

1.9.1. 如何在 ArcMap 中设置地理坐标系和投影坐标系

本例要实现的功能是给 Map 设置地理坐标系或者投影坐标系。如果改变 Map 中的地理坐标系或投影坐标系，Map 中图形的形状也会随之改变。

● 要点

在 ArcGis 中，坐标系分为地理坐标系和投影坐标系，创建一个坐标系通常用接口 ISpatialReferenceFactory2 实现。在接口 ISpatialReferenceFactory2 中有两个常用的方法：CreateGeographicCoordinateSystem 和 CreateProjectedCoordinateSystem，分别用来创建地理坐标系和投影坐标系。并且，ArcGis 中预定义了大多数国际上公认的坐标系统的参数，可以直接引用。此外，也可以读入 prj 文件进行设置。

● 程序说明

本例在点击按钮事件中设置 Map 的坐标系。由于 Map 中不能同时设置地理坐标系和投影坐标系，所以例子代码中注释了设置地理坐标系的代码。如果读者想给 Map 设置地理坐标系，只要去掉相关地理坐标系的注释，并且注释掉相关投影坐标系的代码即可。

● 代码

```

Private Sub UIButtonControll_Click()
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pSpatialReferenceF As ISpatialReferenceFactory2
    Dim pProjectedCoordinateS As IProjectedCoordinateSystem
    'Dim pGeographicCoordinateS As esriCore.IGeographicCoordinateSystem

    On Error GoTo ErrorHandler

```

```

Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap

' 实例化 ISpatialReferenceFactory2 的变量
Set pSpatialReferenceF = New SpatialReferenceEnvironment
' 创建空间投影坐标系，类型为 TokyoJapan10
Set pProjectedCoordinateS = pSpatialReferenceF. _
    CreateProjectedCoordinateSystem(esriSRProjCS_TokyoJapan10)
' 创建地理坐标系，类型为 GCS_Tokyo
' Set pGeographicCoordinateS = pSpatialReferenceF. _
    CreateGeographicCoordinateSystem(esriSRGeoCS_Tokyo)
' 给 Map 设置投影坐标系
Set pMap.SpatialReference = pProjectedCoordinateS
' 给 Map 设置地理坐标系
' Set pMap.SpatialReference = pGeographicCoordinateS
' 弹出消息框显示当前 Map 的坐标系名字
MsgBox pMap.SpatialReference.Name
pMxDocument.ActiveView.Refresh
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.9.2. 如何修改层的坐标系统

当 ArcMap 加载一个 Shape 文件的时候，会检查有没有跟 Shape 文件相关的 prj 文件，如果有就根据该 prj 文件内的参数设置层的坐标系统，反之就会提示该 Shape 文件没有相关的空间坐标系，并且把层的坐标系统设置为“Unknown”。本例要实现的功能是当 ArcMap 已经运行时，改变一个已加载进 ArcMap 的 Feature Layer 的坐标系统。

● 要点

1. 本例使用接口 IGeoDatasetSchemaEdit 来改变层的坐标系统。
2. 本例使用的数据包括两个 Shape 文件，在路径..\data\1.9Projection 下。

File Name	Description	ShapeType	Projected Coord System
europeEquidistant	Eastern Europe	Polygon	Two Point Equidistant
westeuutm33	Western Europe	Polygon	UTM Grid Zone 33N

3. 本例的操作过程为：首先在以上提到的目录下重命名文件 europeEquidistant.prj 为 europeEquidistant.prjxxx；接着打开工程文件，

你会发现两个层的图形不在一起,层 europeEquidistant 在层 westeuutm33 的东南边;你也可以观察两个层的坐标系统,层 westeuutm33 为 UTM Grid Zone 33N,而层 europeEquidistant 为 Unknown;现在运行宏里的程序,你会发现层 europeEquidistant 经过坐标投影后跟层 westeuutm33 排在一起了。这是回到数据保存的路径下,会发现多了一个 europeEquidistant.prj 文件。

● 程序说明

在过程运行时,必须保证层 europeEquidistant 为第一个层。程序首先检查层 europeEquidistant 的坐标系统名字是否为“Unknown”,如果是就把它变换为 Two Point Equidistant 投影坐标系。

● 代码

```
Sub AlterSpatialReference()
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pFeatureLayer As IFeatureLayer
    Dim pGeoDataset As IGeoDataset
    Dim pSpatialReference As ISpatialReference
    Dim pFeatureClass As IFeatureClass
    Dim pGeoDatasetEdit As IGeoDatasetSchemaEdit
    Dim pSpatialReferenceF As ISpatialReferenceFactory2
    Dim pProjectedCoordinateS As IProjectedCoordinateSystem

    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    Set pMap = pMxDocument.FocusMap
    'Assume that europeEquidistant is added last and that it is at the top of the map.
    Set pFeatureLayer = pMap.Layer(0)
    'This is how we get the current spatial reference for a layer
    'QI for the geodataset from the layer
    Set pGeoDataset = pFeatureLayer
    Set pSpatialReference = pGeoDataset.SpatialReference

    'Note that ArcMap sets the SR as "Unknown"
    MsgBox pFeatureLayer.Name + " SpatialReference is " + pSpatialReference.Name
    If (pSpatialReference.Name = "Unknown") Then
        'Get the FeatureClass from the Layer
        Set pFeatureClass = pFeatureLayer.FeatureClass
        'QI for the Geodataset from the FeatureClass
        Set pGeoDataset = pFeatureClass
        'QI for GeoDatasetSchemaEdit from the Geodataset
        Set pGeoDatasetEdit = pGeoDataset
        'Test if we can alter the spatialreference, if we can then we create a factory
        'and use that to create a projected coordinate system.
        If (pGeoDatasetEdit.CanAlterSpatialReference = True) Then
            Set pSpatialReferenceF = New SpatialReferenceEnvironment
            'Use a SpatialReferenceFactory to create the Projected Coordinate System.
            'Here we are using a Factory Code for the Two Point Equidistant
            'coordinate system.
```

```

        Set pProjectedCoordinateS = pSpatialReferenceF._
        CreateProjectedCoordinateSystem(esriSRProjCS_World_TwoPointEquivalent)
        'Now alter the layers spatial reference
        pGeoDatasetEdit.AlterSpatialReference pProjectedCoordinateS
        'Now get the updated SpatialReference and its name
        Set pSpatialReference = pGeoDataset.SpatialReference
        MsgBox pFeatureLayer.Name + " SpatialReference is " + pSpatialReference.Name
        'Force a full refresh
        pMxDocument.ActiveView.Refresh
    End If
End If
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.9.3. 如何把 Polygon 的顶点从经纬度坐标转换到平面直角坐标

本例要实现的功能是对 Polygon 的顶点作坐标投影。所谓坐标投影就是把在地理坐标系下的经纬度坐标按照指定的投影系转换成平面坐标。比如东经 139.382 度，北纬 35.727 度的一个点，按照东京测地系第 9 系的投影坐标进行投影后的平面坐标为 (-40799.457, -30168.456)。

● 要点

坐标的投影主要是对一个几何图形进行。对于一个已经设置了地理坐标系的图形，比如一个 polygon，可以利用 IPolygon 的 project 方法对其进行投影，使其顶点的坐标投影为平面直角坐标。

● 程序说明

本例在鼠标点击事件中得到唯一被选择的 Feature，并得到该 Feature 的图形上在鼠标点击位置的顶点。过程 VertexProject 对图形进行投影，并且弹出消息框显示同一个顶点在投影前和投影后的坐标值。

● 代码

```

Private Sub UIToolControl1_MouseDown(ByVal button As Long, ByVal shift As Long, _
                                     ByVal x As Long, ByVal y As Long)
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pEnumFeature As IEnumFeature
    Dim pFeature As IFeature
    Dim pPoint As IPoint
    Dim pHitPoint As IPoint
    Dim pHitTest As IHitTest
    Dim dSearchRadius As Double
    Dim dHitDistance As Double

```



```

Dim lHitPartIndex As Long
Dim lHitSegmentIndex As Long

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pPoint = pMxDocument.ActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(x, y)
' 计算查找半径
dSearchRadius = pMxDocument.ActiveView.Extent.Width / 100
' 只在一个 Feature 被选择的情况下进行
If pMap.SelectionCount = 1 Then
    Set pEnumFeature = pMap.FeatureSelection
    If pEnumFeature Is Nothing Then Exit Sub
    pEnumFeature.Reset
    ' 得到被选择的 Feature
    Set pFeature = pEnumFeature.Next
    If Not pFeature Is Nothing Then
        ' 根据鼠标点得到图形相应的顶点
        Set pHitTest = pFeature.Shape
        If pHitTest.HitTest(pPoint, dSearchRadius, esriGeometryPartVertex,
            pHitPoint, dHitDistance, lHitPartIndex, lHitSegmentIndex, False) Then
            ' 投影变换
            VertexProject pFeature.Shape, lHitPartIndex, lHitSegmentIndex
        End If
    End If
End If
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

Private Sub VertexProject(ByRef pGeometry As IGeometry, ByVal lPartIndex As Long, _
    ByVal lVertexIndex As Long)
    Dim pPointCollection As IPointCollection
    Dim pPoint As IPoint
    Dim pSpatialReferenceF As ISpatialReferenceFactory2
    Dim pSpatialReference As ISpatialReference

    On Error GoTo ErrorHandler
    If pGeometry Is Nothing Then Exit Sub
    Set pPointCollection = pGeometry
    ' 得到被选中的顶点
    Set pPoint = pPointCollection.Point(lVertexIndex)
    ' 弹出投影变换前的 X、Y 坐标值
    MsgBox "Before projected X: " & pPoint.x & " Y: " & pPoint.y
    Set pSpatialReferenceF = New SpatialReferenceEnvironment
    Set pSpatialReference = pSpatialReferenceF. _
        CreateProjectedCoordinateSystem(esriSRProjCS_Tokyo, apn9)
    ' 给图形设置投影坐标系
    pGeometry.Project pSpatialReference
    Set pPoint = pPointCollection.Point(lVertexIndex)
    ' 弹出投影变换后的 X、Y 坐标值

```

```

        MsgBox "After projected X: " & pPoint.x & " Y: " & pPoint.y
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1. 10. ArcGis 相关文件

1. 10. 1. 如何夹载 grf 文件

本例要实现的是如何在把一个 grf 文件加载到当前工程中。grf 文件即 Grapher(GolderSoftware 公司)图形文件,可以在 ArcMap 中由 Tools->Graphs->Create... 来创建一个当前图层的 grf 文件。利用 IDataGraph, IDataGraphs, IDataGraphWindow 三个接口来实现加载 grf 文件。

● 程序说明

在过程 addGrfFile 中实现加载 grf 文件的过程。首先利用当前的 MxDocument 生成一个 IDataGraphs 接口的对象,然后用该对象创建一个 IDataGraph 对象用来控制 grf 图形文件。IDataGraphWindow 接口的对象用来显示图形。

● 代码

```

Private Sub UIButtonControll1_Click()
    'Call addGrfFile
    Dim pVBProject As VBProject
    Dim nIndex As Integer
    Set pVBProject = ThisDocument.VBProject
    nIndex = InStrRev(pVBProject.fileName, "\")
    addGrfFile Left(pVBProject.fileName, nIndex) & "..\..\data\tmpIdoukui.g f"
End Sub

Private Sub addGrfFile(ByVal sFileName As String)
    On Error GoTo ErrorHandler:
        Dim pMxDocument As IMxDocument
        Dim pDataGraphW As IDataGraphWindow
        Dim pDataGraph As IDataGraph
        Dim pDataGraphs As IDataGraphs
        Dim sGrfFile As String

        If Dir(sFileName) = "" Then
            MsgBox sFileName & " File is not Exist"
            Exit Sub
        End If
        sGrfFile = sFileName
        Set pMxDocument = ThisDocument
        Set pDataGraphs = pMxDocument
        Set pDataGraph = pDataGraphs.Create
        pDataGraph.LoadFromFile sGrfFile

```

```

Set pDataGraphW = New DataGraphWindow
Set pDataGraphW.DataGraph = pDataGraph
Set pDataGraphW.Application = Application

Exit Sub
ErrorHandler:
MsgBox "There are some error!"
End Sub

```

1.10.2. 如何新建指向 Shape 文件的 lyr 文件

本例实现的是如何新建一个指向 Shape 文件的 lyr 文件。利用 IGxLayer, IGxFile 和 IFeatureLayer 接口来实现该功能。通过 GxLayer 类实现 IGxLayer 接口对象。通过 GxFile 类实现 IGxFile 接口对象

●程序说明

函数 OpenFeatureClass 打开用来新建 lyr 文件的 shape 文件。参数 sPath, sName 指定 shape 文件的位置。过程 CreateLyrFileFromShape 创建 lyr 文件。

●代码

```

Option Explicit

Private Sub CreateLyrFileFromShape(sLyrFilePath As String, sShpFilePath As String, _
sShpFileName As String)
On Error GoTo ErrorHandler:
Dim pGxLayer As IGxLayer
Dim pGxFile As IGxFile
Dim pFeatureLayer As IFeatureLayer
Dim pMxDocument As IMxDocument

Set pGxLayer = New GxLayer
Set pGxFile = pGxLayer
pGxFile.Path = sLyrFilePath

Set pFeatureLayer = New FeatureLayer
Set pFeatureLayer.FeatureClass = OpenFeatureClass(sShpFilePath, sShpFileName)
If pFeatureLayer.FeatureClass Is Nothing Then
GoTo ErrorHandler:
End If

pFeatureLayer.Name = "myCountryLyr"
Set pGxLayer.Layer = pFeatureLayer

Set pMxDocument = ThisDocument
pMxDocument.FocusMap.AddLayer pGxLayer.Layer

```

```

        Set pGxLayer = Nothing
        Set pGxFile = Nothing
        Exit Sub
ErrorHandler:
        MsgBox Err.Description
End Sub

Function OpenFeatureClass(ByVal sPath As String, ByVal sName As String) As IFeatureClass
On Error GoTo ErrorHandler:
        Dim pWorkspaceFactory As IWorkspaceFactory
        Set pWorkspaceFactory = New ShapefileWorkspaceFactory

        Dim pFeatureWorkspace As IFeatureWorkspace
        Set pFeatureWorkspace = pWorkspaceFactory.OpenFromFile(sPath, 0)
        Set OpenFeatureClass = pFeatureWorkspace.OpenFeatureClass(sName)
        Exit Function
ErrorHandler:
        Set OpenFeatureClass = Nothing
End Function

Private Sub UIButtonControll1_Click()
        Dim sDataFilePath As String
        Dim pVBProject As VBProject
        Dim nIndex As Integer
        Set pVBProject = ThisDocument.VBProject
        nIndex = InStrRev(pVBProject.FileName, "\")
        sDataFilePath = Left(pVBProject.FileName, nIndex) & "..\..\..\data\"
        'call AddLayerFileToMap
        CreateLyrFileFromShape sDataFilePath & "myCountry.lyr", sDataFilePath,
"WorldCountries"
End Sub

```

1.10.3. 如何新建指向 GeoDataBase 文件的 lyr 文件

本例实现的是如何新建指向 GeoDataBase 文件的 lyr 文件。本实例的过程和新建指向 Shape 文件的 lyr 文件相似。

方法：使用 IGxLayer, IGxFile, IFeatureLayer, IPropertySet 等接口实现该实例的要求。

●要点

使用 IPropertySet 接口设置 lyr 文件的数据源。语句 pPropertySet.SetProperty "DATABASE", GdbPath + GdbName 用来设置数据源。

●程序说明

函数 OpenFeatureClass 根据 GeoDatabase 文件的所在位置和表名来设置 lyr 文件的数据源，过程 CreateLyrFileFromGDB 依据传入的参数 sLyrFilePath 创建 lyr

文件。在 UIButtonControl1 按钮的 click 事件中调用 CreateLyrFileFromGDB 过程。

● 代码

```
Option Explicit

Private Sub CreateLyrFileFromGDB(sLyrFilePath As String, sMDBFileName As String,
sTableName As String)
On Error GoTo ErrorHandler:
    Dim pGxLayer As IGxLayer
    Dim pGxFile As IGxFile
    Dim pFeatureLayer As IFeatureLayer
    Dim pMxDocument As IMxDocument

    If Dir(sLyrFilePath) <> "" Then
        MsgBox "File already Exist"
        Exit Sub
    End If
    Set pGxLayer = New GxLayer
    Set pGxFile = pGxLayer
    pGxFile.Path = sLyrFilePath

    Set pFeatureLayer = New FeatureLayer
    Set pFeatureLayer.FeatureClass = OpenFeatureClass(sMDBFileName, sTableName)
    If pFeatureLayer.FeatureClass Is Nothing Then
        GoTo ErrorHandler:
    End If

    pFeatureLayer.Name = "myCountryLyr"
    Set pGxLayer.Layer = pFeatureLayer
    Set pMxDocument = ThisDocument
    pMxDocument.FocusMap.AddLayer pGxLayer.Layer

    Set pGxLayer = Nothing
    Set pGxFile = Nothing
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Function OpenFeatureClass(ByVal GdbName As String, ByVal tableName As String) As
IFeatureClass
On Error GoTo ErrorHandler:
    Dim pPropertySet As IPropertySet
    Dim pFeatureWorkspace As IFeatureWorkspace
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pWorkspace As IWorkspace
    Dim pFeatureClass As IFeatureClass

    Set pPropertySet = New PropertySet
    'Set data source
    pPropertySet.SetProperty "DATABASE", GdbName
    Set pWorkspaceFactory = New AccessWorkspaceFactory
```

```

Set pWorkspace = pWorkspaceFactory.Open(pPropertySet, 0)
Set pFeatureWorkspace = pWorkspace
Set pFeatureClass = pFeatureWorkspace.OpenFeatureClass(tableName)

If Not pFeatureClass Is Nothing Then
    Set OpenFeatureClass = pFeatureClass
End If
Exit Function
ErrorHandler:
Set OpenFeatureClass = Nothing
Dim Msg As String
Msg = "Error # " & Str(Err.Number) & " was generated by " _
    & Err.Source & Chr(13) & Err.Description
MsgBox Msg, "Error", Err.HelpFile, Err.HelpContext
End Function

Private Sub UIButtonControl1_Click()
Dim sDataFilePath As String
Dim pVBProject As VBProject
Dim nIndex As Integer
Set pVBProject = ThisDocument.VBProject
nIndex = InStrRev(pVBProject.FileName, "\")
sDataFilePath = Left(pVBProject.FileName, nIndex) & "..\..\..\data\" & "US "
'call AddLayerFileToMap
CreatelyrFileFromGDB Left(sDataFilePath, Len(sDataFilePath) - 3) & "myCount: 1.lyr",
sDataFilePath, "states"
End Sub

```

1.10.4. 如何加载 mxd 文件

mxd 文件是 ArcMap 生成的工程文件。本例将告诉你如何在程序中加载 mxd 文件。使用 MapControl 控件来实现 mxd 文件的加载。

● 要点

首先在 VBA 中添加一个 Form，然后在 Toolbox 中单击鼠标右键，在弹出的窗口中选择 Additional Controls... 将会出现 Additional Controls 对话框，从中选择 ESRI MapControl 8.1，单击 OK。在 Toolbox 中就会出现 MapControl 控件。

● 程序说明

在 UserForm1 中添加一个 MapControl 控件和两个按钮。一个 name 为 cmdAddMxd, caption 为 Add MxdFile, 另一个 name 为 cmdClose, caption 为 Close。

● 代码

```

Private Sub cmdAddMxd_Click()
Dim pVbproject As New VBProject
Set pVbproject = ThisDocument.VBProject
Dim pMxfilePath As String

```

```

pMxfilePath = pVbproject.FileName & "..\" & "data\addLyrFile.mxd"

'Load a MxFile according the pMxfilePath
UserForm1.MapControl1.LoadMxFile pMxfilePath

'show in full extent
MapControl1.Extent = MapControl1.FullExtent
End Sub

Private Sub cmdClose_Click()
    Unload Me
End Sub

```

1.10.5. 如何加载 Apr 文件(ArcView32)

本例要实现的是如何在 ArcGis 中加载 Apr(ArcView32 工程)文件。首先使用 ArcView32 创建一个 Apr 文件。

方法：使用 IAVObject, IAVObjectConverter, IMap 三个接口来实现该实例。

●要点

定义 IAVObjectConverter 接口对象，并用 AVObjectConverter 类来实现，利用该对象读取 Apr 文件。然后将 IAVObjectConverter 接口对象中的 View 对象赋值给 IAVObject 接口的一个对象。使用 IAVObjectConverter 接口的 ConnectToView 方法返回一个和 AV View 对象相联系的 IMap 对象。

●程序说明

本实例在过程 ImportAprFile 中实现加载 Apr 文件的过程，然后在 UIButtonControl 按钮的 Click 事件中调用 ImportAprFile 过程。

●代码

```

Public Sub ImportAprFile()
    Dim pAVObject          As IAVObject
    Dim pAVObjectC          As IAVObjectConverter
    Dim pLayer              As ILayer
    Dim pAprDialog          As New CommonDialog
    Dim sAprFile            As String
    Dim pMap                As IMap
    Dim pMxDocument         As IMxDocument

    On Error GoTo ErrorHandler
    pAprDialog.DialogTitle = "Select ArcView 32 Project file"
    pAprDialog.Filter = "ArcView32 Project file(*.apr)|*.apr"
    pAprDialog.FilterIndex = 0
    pAprDialog.CancelError = True
    pAprDialog.ShowOpen
    sAprFile = pAprDialog.FileName
    If sAprFile = "" Then

```

```

        MsgBox "you have not select a Apr file"
    End If

    Set pAVObjectC = New AVObjectConverter
    'Read Apr file
    pAVObjectC.ReadObjects sAprFile
    pAVObjectC.Reset
    Set pAVObject = pAVObjectC.NextObject
    'Make sure search for a View object
    Do While pAVObject.Type <> "View"
        Set pAVObject = pAVObjectC.NextObject
        If pAVObject Is Nothing Then Exit Do
    Loop
    'Returns the map associated with an AV view object
    Set pMap = pAVObjectC.ConnectToView(pAVObject)

    Set pMxDocument = ThisDocument
    Dim LayerCount As Integer
    LayerCount = 0
    'Show the View Object in ThisDocument
    Do While LayerCount < pMap.LayerCount
        pMxDocument.AddLayer pMap.Layer(LayerCount)
        LayerCount = LayerCount + 1
    Loop
    Exit Sub
ErrorHandler:
    Msg = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description
    MsgBox Msg, , "Error", Err.HelpFile, Err.HelpContext
End Sub

Private Sub UIButtonControll1_Click()
    ImportAprFile
End Sub

```

1.10.6. 如何加载 lyr 文件

后缀为 lyr 的文件即 layer file，存放关于 layer 的相关信息。本实例实现的功能是将一 layer file 加载到当前打开的 Map document 中。

方法：在实现的过程中主要用到了 IGxLayer，IGxFile 两个接口。

●要点

利用 IGxLayer 对象的 Path 属性得到 layer file。将 IGxLayer 对象的 Layer 属性赋值给一个 IFeatureLayer 对象。然后利用 AddLayer 方法将该 layer file 加载到当前的 Map document 中。

●程序说明

首先从 Tools->Customize... 的对话框中选择 Commands 标签，选中 UIControls 然后单击 New UIControls 按钮新建一个 Project.UIButtonControll

按钮，并且将其拖放到 ArcMap 的工具栏上。在 Project.UIButtonControll 按钮的 click 事件中调用过程 AddLayerfileToMap 实现加载 lyr 文件。

● 代码

```
Option Explicit

Private Sub AddLayerFileToMap(sLyrFilePath As String)
On Error GoTo ErrorHandler:
    Dim pGxLayer As IGxLayer
    Dim pGxFile As IGxFile
    Dim pMxDoc As IMxDocument
    Dim pFeatureLayer As IFeatureLayer

    If Dir(sLyrFilePath) = "" Then
        MsgBox sLyrFilePath & " File is not Exist"
        Exit Sub
    End If
    Set pGxLayer = New GxLayer
    Set pGxFile = pGxLayer
    pGxFile.Path = sLyrFilePath
    Set pFeatureLayer = pGxLayer.Layer

    Set pMxDoc = ThisDocument
    pMxDoc.FocusMap.AddLayer pGxLayer.Layer
    pMxDoc.ActiveView.Refresh
    Set pGxLayer = Nothing
    Set pGxFile = Nothing
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub UIButtonControll_Click()
    Dim sLyrFilePath As String
    Dim nIndex As Integer
    Dim pVBProject As VBProject
    Set pVBProject = ThisDocument.VBProject
    nIndex = InStrRev(pVBProject.FileName, "\")
    sLyrFilePath = Left(pVBProject.FileName, nIndex) & "..\..\..\data\" & 'standb5
polygon.lyr'
    'call AddLayerFileToMap
    AddLayerFileToMap sLyrFilePath
End Sub
```

1.10.7. lyr 文件的属性的设置

本例要实现的是如何对 lyr 文件的属性进行设置。在此例中是对 lyr 文件的 name 属性和 scale 属性以及 visiable 属性的一些设置操作。

方法：使用 IGxLayer, IGxFile 等接口的方法和属性对 lyr 文件进行操作

● 要点

首先在 VBA 中插入一个类模块取名为 clsLyrFile, 在该模块中实现对 lyr 文件的属性的设置。然后插入一个 Form 模块取名为 SetPropertyFrm 用来输入设置参数。在 Module 模块中放公共变量。

● 程序说明

在 clsLyrFile 中 OpenFile 过程用来加载 lyr 文件, SetScale 过程设置 lyr 文件的显示比例。SetName 过程设置 lyr 文件的 name 属性。SetVisible 过程决定 lyr 文件是否可见。SaveFile 过程保存 lyr 文件。

● 代码

```
' clsLyrFile 模块中的代码
Option Explicit

Private pGxLayer As IGxLayer
Private pGxFile As IGxFile
Private PFeatureLayer As IFeatureLayer

Public Sub OpenFile(ByVal lyrFileName As String)
On Error GoTo ErrorHandler
    Set pGxLayer = New GxLayer
    Set pGxFile = pGxLayer
    pGxFile.Path = lyrFileName
    Set PFeatureLayer = pGxLayer.Layer
    mMxDocument.FocusMap.AddLayer PFeatureLayer
    Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Public Sub SetScale(ByVal maximumScale As Double, ByVal minimumScale As Double)
    If Not PFeatureLayer Is Nothing Then
        'MsgBox PFeatureLayer.maximumScale & ":" & PFeatureLayer.minimumScale
        PFeatureLayer.maximumScale = maximumScale
        PFeatureLayer.minimumScale = minimumScale
    End If
End Sub

Public Sub SetName(ByVal Name As String)
    If Not pFeatureLayer Is Nothing Then
        pFeatureLayer.Name = Name
    End If
End Sub

'Decide the lyr file is visiable or not.
'if visiable is ture then lyr file is visiable.
Public Sub SetVisible(ByVal visiable As Boolean)
    pGxLayer.Layer.Visible = visiable
End Sub

Public Sub SaveFile()
```

```

On Error GoTo ErrorHandler
    If Not pGxFile Is Nothing Then
        If Not PFeatureLayer Is Nothing Then
            Set pGxLayer.Layer = PFeatureLayer
        End If
        pGxFile.Save
        pGxFile.Close True
    End If
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

' SetPropertyFrm 模块中的代码
Option Explicit

Private Sub cmdCancel_Click()
On Error GoTo ErrorHandler
    If MsgBox("Are you want to save the set!", vbOKCancel) = vbOK Then
        mCurrentLyr.SaveFile
    End If
    Unload Me
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub cmdOk_Click()
On Error GoTo ErrorHandler
    If txtName.Text <> "" Then
        mCurrentLyr.SetName txtName.Text
        mMxDocument.ContentView(0).Refresh (0)
        mMxDocument.ActiveView.Refresh
    End If
    mCurrentLyr.SetVisible (cbxVisiable.Value)
    mMxDocument.UpdateContents
    If optNotShow.Value Then
        If IsNumeric(txtMaximum.Text) And IsNumeric(txtMinimum.Text) Then
            mCurrentLyr.SetScale CDBl(txtMaximum.Text), CDBl(txtMinimum.Text)
            mMxDocument.ActiveView.Refresh
        Else
            MsgBox "you must input some numeric!"
        End If
    End If
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub

Private Sub optNotShow_Click()
    txtMaximum.Enabled = True
    txtMinimum.Enabled = True
    txtMaximum.BackColor = &H80000005

```

```

        txtMinimum.BackColor = &H80000005
End Sub

Private Sub optShow_Click()
    txtMaximum.Enabled = False
    txtMinimum.Enabled = False
    txtMaximum.BackColor = &H80000004
    txtMinimum.BackColor = &H80000004
    mCurrentLyr.SetScale 0, 0
    mCurrentLyr.SetVisible (True)
End Sub

' Module 模块中的代码
Option Explicit
'some common variant
Public mCurrentLyr As clsLyrFile
Public mMxDocument As IMxDocument

```

1. 11. 其他

1. 11. 1. 如何创建简单的 Column Chart

本例实现的功能是根据 FeatureLayer 被选择 Features 和属性表中指定的字段，生成一个 Chart。主要用到 IDataGraphProperties 接口。

● 要点

要实现本例的功能，首先需要创建一个 IDataGraph，然后通过 IDataGraphProperties 接口来设置 DataGraph 的属性，接着创建 IDataGraphWindow，把 pDataGraph 赋给 pGraphWindow。DataGraph 就可以实现

● 程序说明

先把选择的数据赋给 pDataGraph.Table，再用 pDataGraph.Field1 设置要显示的 X 轴的字段，把 pDataGraph 赋给 pDataGraphP，再设置 pDataGraphP 的属性，然后再把 DataGraphP 赋给 pGraphWindow.DataGraph，最后把 pGraphWindow 显示出来就完成了。

● 代码

```

Private Sub UIButtonControll_Click()
    ' call sub CreateNewChart
    CreateNewChart
End Sub

Public Sub CreateNewChart()
    Dim pMxDocument As IMxDocument
    Dim pDataGraph As IDataGraph

```

```

Dim pDataGraphP          As IDataGraphProperties
Dim pGraphWindow         As IDataGraphWindow
Dim pDataGraphs          As IDataGraphs

On Error GoTo ErrorHandler
Set pMxDocument = ThisDocument
If pMxDocument.SelectedLayer Is Nothing Then Exit Sub
If Not TypeOf pMxDoc.SelectedLayer Is IFeatureLayer Then Exit Sub
'Create a new graph
Set pDataGraph = New DataGraph
'Set the default Table, DataGraph will select a default graph type and some fields
Set pDataGraph.Table = pMxDocument.SelectedLayer
'Specifically give the chart a name and title
pDataGraph.Name = pMxDocument.SelectedLayer.Name & " Chart"
Set pDataGraphP = pDataGraph
pDataGraphP.Title = "My Chart"
pDataGraphP.ShowXAxisLabels = True
pDataGraphP.ShowLegend = True
pDataGraphP.SubTitle = pDataGraph.FieldSet1 & " vs. " & pDataGraphP.XaxisLabelField
'Associate the data graph with a data graph window
Set pGraphWindow = New DataGraphWindow
Set pGraphWindow.DataGraph = pDataGraph
Set pGraphWindow.Application = Application
'Add the graph to the system
Set pDataGraphs = pMxDocument
pDataGraphs.Add pDataGraph
Exit Sub

ErrorHandler:
MsgBox Err.Description
End Sub

```

1. 11. 2. 如何将数据输出到 Excel

本例要实现的是将指定文件里的 DBF 数据用导出到 Excel 中。

● 程序说明

先将数据读入到 Recordset 中, 然后再将 Recordset 中的数据按照 Excel 中单元格的位置填写到 Sheet 中。

● 代码

```

Private Sub UIButtonControl1_Click()
Dim pDocument          As IDocument
Dim pVBProject          As VBProject
Dim strPath            As String

Set pDocument = ThisDocument
Set pVBProject = pDocument.VBProject
strPath = pVBProject.FileName
strPath = Mid(strPath, 1, InStrRev(strPath, "\"))
Call ExportToExcel(strPath & "\..\..\data")

```

```

End Sub

Public Function ExportToExcel(ByVal strPath As String) As Boolean
    Dim adoConn          As New ADODB.Connection
    Dim Rs_Data1         As New ADODB.Recordset
    Dim xlApp            As New Excel.Application
    Dim xlBook           As New Excel.Workbook
    Dim xlSheet1         As New Excel.Worksheet
    Dim strSQL           As String
    Dim lngRow           As Long
    Dim lngCol           As Long
    Dim lngTag           As Long

    On Error GoTo ErrorHandler

    ' 打开一个数据库连接
    adoConn.Open "[PROVIDER=MSDASQL.1];DRIVER=Microsoft Visual FoxPro
Driver;UID=;Deleted=yes;" & _

        "Null=no;Collate=Machine;BackgroundFetch=no;Exclusive=No;SourceType=Default;Source
eDB=" & _ strPath
    strSQL = "SELECT * FROM spcs27;"
    ' 打开一个记录集
    Rs_Data1.Open strSQL, adoConn, adOpenStatic, adLockReadOnly
    xlApp.Visible = False
    ' 新建一个 Excel 对象
    Set xlApp = CreateObject("Excel.Application")
    ' 设置 Sheet 的个数
    xlApp.SheetsInNewWorkbook = 1
    Set xlBook = xlApp.Workbooks().Add
    Set xlSheet1 = xlBook.Worksheets("sheet1")
    ' 重命名 Sheet1
    xlSheet1.Name = "spcs27"
    ' Excel 不可见
    xlApp.Visible = False
    If Not IsNull(Rs_Data1) Then
        lngTag = 1
        lngRow = 2
        While Not Rs_Data1.EOF
            If lngTag = 1 Then
                For lngCol = 0 To Rs_Data1.Fields.Count - 1
                    ' 将 Recordset 的字段名称写入 Sheet 的单元格中
                    xlSheet1.Cells(1, lngCol + 1) = Rs_Data1.Fields(lngCol).Name
                Next lngCol
                lngTag = lngTag + 1
            End If
            If lngTag > 1 Then
                For lngCol = 0 To Rs_Data1.Fields.Count - 1
                    ' 将 Recordset 的字段值写入 Sheet 的单元格中
                    xlSheet1.Cells(lngRow, lngCol + 1) = Rs_Data1.Fields(lngCol).Value
                Next lngCol
            End If
            Rs_Data1.MoveNext
            lngRow = lngRow + 1
        End While
    End If
End Function

```

```

        Wend
    End If
    'Excel 可见
    xlApp.Visible = True
    '清除对象变量
    Set xlApp = Nothing
    Set xlBook = Nothing
    Set xlSheet1 = Nothing

    Rs_Data1.Close
    Set Rs_Data1 = Nothing
    ExportToExcel = True
    Exit Function

ErrorHandler:
    '出错处理
    ExportToExcel = False
    MsgBox Err.Description
End Function

```

1.11.3. 如何把 Labels 转换为 Annotation

本例要完成的功能是根据指定 FeatureLayer 的 Labels 生成一个新的 Annotation 类型的 FeatureLayer。

● 要点

用 IFeatureWorkspaceAnno 的 CreateAnnotationClass 方法可以创建 Annotation 的 FeatureClass, 在创建之前, 先要设定好 Annotation 层的名字, 相关属性表的字段, GraphicsLayerScale, SymbolCollection 等。

● 程序说明

首先根据指定 FeatureLayer 中的 Label 创建一个 Annotation 层的 FeatureClass, 然后根据新生成的 FeatureClass 在 GeoDatabase 中新建一个属性层, 并添加到当前的 Map 中。

● 代码

```

Option Explicit
Const ANNO_FC_NAME = "Test_01"

Private Sub UIButtonControl1_Click()
    ConvertLabels2Anno
End Sub

Sub ConvertLabels2Anno()
    On Error GoTo ErrorHandler

    Dim pMxDocument As IMxDocument

```

```

Dim pMap As IMap
Dim pLayer As ILayer
Dim pGeoFeatureL As IGeoFeatureLayer
Dim pAnnoClass As IFeatureClass
Dim pFDODGraphicsLayerF As IFDODGraphicsLayerFactory
Dim pAnnoDataset As IDataset
Dim pAView As IActiveView
Dim pFDODGraphicsLayer As IFDODGraphicsLayer '

Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pLayer = pMap.Layer(0)
Set pGeoFeatureL = pLayer

If pMap Is Nothing Then
    MsgBox "No active map"
    Exit Sub
End If

If pGeoFeatureL Is Nothing Then
    MsgBox "First layer in map must be feature layer"
    Exit Sub
End If

Set pAnnoClass = CreateRelatedAnnoClass(pMap, pGeoFeatureL)
Set pFDODGraphicsLayerF = New FDODGraphicsLayerFactory
Set pAnnoDataset = pAnnoClass
Set pFDODGraphicsLayer = pFDODGraphicsLayerF.OpenGraphicsLayer(pAnnoDataset.Workspace, pAnnoClass.FeatureDataset, pAnnoDataset.Name)

pMap.AddLayer pFDODGraphicsLayer
AddAnnoElements pMap, pGeoFeatureL, pFDODGraphicsLayer

' make sure labels are off for source layer
pGeoFeatureL.DisplayAnnotation = False

' refresh map
Set pAView = pMap
pAView.Refresh
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Private Function CreateRelatedAnnoClass(ByVal pMap As IMap, & _
    ByVal pGeoFeatureLayer As IGeoFeatureLayer) & _
    As IFeatureClass

On Error GoTo ErrorHandler

    Dim pFeatureClass As IFeatureClass
    Dim pDataset As IDataset

```



```

Dim pFeatureWorkspaceA      As IFeatureWorkspaceAnno
Dim pAnnotateLayerPC        As IAnnotateLayerPropertiesCollection
Dim pAnnotateLayerP         As IAnnotateLayerProperties
Dim pLabelEngineLLP         As ILabelEngineLayerProperties
Dim pAnnotateLayerTP        As IAnnotateLayerTransformationProperties
Dim pGeoDataset             As IGeoDataset
Dim pSymbolCollection       As ISymbolCollection
Dim pGraphicsLayerS         As IGraphicsLayerScale
Dim strAnnoName             As String
Dim pAnnoFeatureCD          As IFeatureClassDescription
Dim pAnnoObjectCD           As IObjectClassDescription
Dim pFields                 As IFields
Dim pFieldsEdit             As IFieldsEdit
Dim pField                  As IField
Dim pFieldEdit              As IFieldEdit
Dim pGeomDefE               As IgeometryDefEdit

Set pFeatureClass = pGeoFeatureLayer.FeatureClass
Set pDataset = pFeatureClass
Set pFeatureWorkspaceA = pDataset.Workspace

'Get Annotation Properties from Layer
Set pAnnotateLayerPC = pGeoFeatureLayer.AnnotationProperties

'Get First AnnotateLayerProperties and QI to ILabelEngineLayerProperties
pAnnotateLayerPC.QueryItem 0, pAnnotateLayerP
Set pLabelEngineLLP = pAnnotateLayerP

'Update IAnnotateLayerTransformationProperties
Set pAnnotateLayerTP = pAnnotateLayerP

Set pGeoDataset = pGeoFeatureLayer.FeatureClass
pAnnotateLayerTP.ReferenceScale = pMap.MapScale
pAnnotateLayerTP.Units = pMap.MapUnits
pAnnotateLayerTP.scaleratio = 1
pAnnotateLayerTP.Bounds = pGeoDataset.Extent

'Add Symbol from ILabelEngineLayerProperties to new symbolcollection
Set pSymbolCollection = New SymbolCollection
Set pSymbolCollection.Symbol(0) = pLabelEngineLLP.Symbol

'Create GraphicsLayerScale object and populate
Set pGraphicsLayerS = New GraphicsLayerScale
pGraphicsLayerS.ReferenceScale = pMap.MapScale
pGraphicsLayerS.Units = pMap.MapUnits
strAnnoName = ANNO_FC_NAME
Set pAnnoFeatureCD = New AnnotationFeatureClassDescription
Set pAnnoObjectCD = pAnnoFeatureCD

Set pFields = pAnnoObjectCD.RequiredFields
Set pFieldsEdit = pFields
Set pField = pFields.Field(pFields.FindField(pAnnoFeatureCD.ShapeFieldName))
Set pFieldEdit = pField

```

```

Set pGeomDefE = pField.GeometryDef
Set pGeomDefE.SpatialReference = pGeoDataset.SpatialReference
Set pFieldEdit.GeometryDef = pGeomDefE
Set pFieldsEdit.Field(pFields.FindField(pAnnoFeatureCD.ShapeFieldName) =
pFieldEdit

Set pFeatureClass = pFeatureWorkspaceA.CreateAnnotationClass(strAnnotationName,
pFieldsEdit, _
pAnnoObjectCD.InstanceCLSID, pAnnoObjectCD.ClassExtensionCLSID, _
pAnnoFeatureCD.ShapeFieldName, "", pFeatureClass.FeatureDataset, pFeatureClass, _
pGeoFeatureLayer.AnnotationProperties, pGraphicsLayerS, pSymbolCollection, True)

Set CreateRelatedAnnoClass = pFeatureClass
Exit Function

ErrorHandler:
Set CreateRelatedAnnoClass = Nothing
End Function

Private Sub AddAnnoElements(pMap As IMap, &_
pGeoFeatLayer As IGeoFeatureLayer, & _
pAnnoLayer As IFDOGraphicsLayer)
On Error GoTo ErrorHandler

Dim pActiveView As IActiveView
Dim pScreenDisplay As IScreenDisplay
Dim pGraphicsLayer As IGraphicsLayer
Dim AnnotateMapP As IAnnotateMapProperties
Dim pAnnoLayerPC As IAnnotateLayerPropertiesCollection
Dim pAnnoLayerP As IAnnotateLayerProperties
Dim pEnv As IEnvelope
Dim pDisplayTransformation As IDisplayTransformation
Dim dblscaleratio As Double
Dim dblrefscale As Double
Dim pBounds As IEnvelope
Dim dunits As esriUnits
Dim pAnnotateLayerTP As IAnnotateLayerTransformationProperties
Dim AnnotateMap As IAnnotateMap
Dim pTrackCancel As ItrackCancel

Set pActiveView = pMap
Set pScreenDisplay = pActiveView.ScreenDisplay
Set pGraphicsLayer = pAnnoLayer
pGraphicsLayer.Activate pScreenDisplay

' get AnnotateMapProperties from source layer
Set AnnotateMapP = New AnnotateMapProperties
' get AnnotateLayerPropertiesCollection from source layer
Set pAnnoLayerPC = pGeoFeatLayer.AnnotationProperties
Set AnnotateMapP.AnnotationLayerPropertiesCollection = pAnnoLayerPC

' get the (first) property set in the collection

```

```

pAnnoLayerPC.QueryItem 0, pAnnoLayerP, Nothing, Nothing

' direct labels to target anno feature class
Set pAnnoLayerP.GraphicsContainer = pAnnoLayer

With pAnnoLayerP
    Set .FeatureLayer = pGeoFeatLayer
    .AddUnplacedToGraphicsContainer = False
    .AnnotationMaximumScale = 0
    .AnnotationMinimumScale = 0
    .CreateUnplacedElements = False
    .DisplayAnnotation = True
    .Extent = pGeoFeatLayer.AreaOfInterest 'source layer
    .FeatureLinked = False
    .LabelWhichFeatures = esriAllFeatures
    .UseOutput = True
    .WhereClause = ""
End With

Set pDisplayTransformation = pScreenDisplay.DisplayTransformation

' dblscaleratio = pDisplayTransformation.sdblscaleratio
dblscaleratio = pDisplayTransformation.scaleratio
dblrefscale = pDisplayTransformation.ReferenceScale
Set pBounds = pDisplayTransformation.Bounds
dunits = pDisplayTransformation.Units

Set pAnnotateLayerTP = pAnnoLayerP
With pAnnotateLayerTP
    .Units = dunits
    .Bounds = pBounds
    .scaleratio = dblscaleratio
    .ReferenceScale = dblrefscale
End With

' Label features
Set AnnotateMap = New AnnotateMap
Set pTrackCancel = New CancelTracker
AnnotateMap.Label AnnotateMapP, pMap, pTrackCancel
Set pAnnoLayerP.FeatureLayer = Nothing
Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1.11.4. 如何把 Annotation 转换为 Polygon Features

本例要完成的功能是将指定的 GeoDatabase 中的 Annotation Features 转换为 Polygon Features。每一个 Annotation Feature 都被转换为多 Ring 的 Polygon

Feature，并且通过创建新的 Feature 保存在一个图形类型为 Polygon 的 FeatureLayer 中。

- 要点

IFDOGraphicsLayerRead. NextGraphic: 得到 FDOGraphicsLayer 层中每一个 Annotation Feature 的 Element 属性 IQueryGeometry.GetGeometry: 得到一个 Element 的 Geometry。

- 程序说明

首先将一个 Annotation 层加入到当前 Map 中，并置为第 0 层；再加一个 polygon 层，并置为第 1 层，并让这些层处于编辑状态。

- 代码

```
Private Sub UIButtonControl1_Click()  
    AnnoPolyCon_Click  
End Sub  
  
Private Sub AnnoPolyCon_Click()  
On Error GoTo ErrorHandler  
    Dim lngFlayerNum As Long  
    Dim lngFDOLayerNum As Long  
    Dim dblReferenceScale As Double  
    Dim dblMapScale As Double  
    Dim dblOutputDPI As Double  
    Dim dblOptimumScale As Double  
    Dim dblScreenResolution As Double  
    Dim dblAnnoScaleFactor As Double  
    Dim dblTempTextSize As Double  
    Dim dblFinalOutputScale As Double  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pActiveView As IActiveView  
    Dim pScreenDisplay As IScreenDisplay  
    Dim pDisplayTransform As IDisplayTransformation  
    Dim pFLayer As IFeatureLayer  
    Dim IFeatureClass As IFeatureClass  
    Dim pClass As IClass  
    Dim pAnnoClass As IAnnoClass  
    Dim pAnnoFeature As IFeature  
    Dim pAnnoElement As IElement  
    Dim pFDOGraphicsLayer As IFDOGraphicsLayer  
    Dim pFDOGraphicsRead As IFDOGraphicsLayerRead  
    Dim pPolygon As IPolygon  
    Dim pTextElement As ITextElement  
    Dim pTextSymbol As ITextSymbol  
    Dim pTextQuery As IQueryGeometry  
    Dim pTextPointGeo As IGeometry  
    Dim pTopoOperator2 As ITopologicalOperator2
```

```

Dim pFeature As IFeature
'SET these variables for your individual case
lngFDOLayerNum = 0
'Set annotation layer here (zero-based: 0 is first layer in TOC)
lngFlayerNum = 1
'Set empty feature layer here (zero-based: 1 is second layer in TOC)
dblOutputDPI = 1200
'Highest DPI of your final output device(s)
dblScreenResolution = 96
'Resolution of your monitor
dblFinalOutputScale = 24000 'Final scale that your map will be printed with
Set pMxDocument = ThisDocument
Set pMap = pMxDocument.FocusMap
Set pActiveView = pMap
Set pScreenDisplay = pActiveView.ScreenDisplay
Set pDisplayTransform = pScreenDisplay.DisplayTransformation
Set pFLayer = pMap.Layer(lngFlayerNum)
Set IFeatureClass = pFLayer.FeatureClass
Set pClass = pMap.Layer(lngFDOLayerNum)
Set pAnnoClass = pClass.Extension

dblReferenceScale = pMap.ReferenceScale
dblMapScale = pMap.MapScale
dblOptimumScale = (dblScreenResolution / dblOutputDPI) * (dblFinalOutputScale / 2)
dblAnnoScaleFactor = pAnnoClass.ReferenceScale / dblOptimumScale

Set pFDOLayer = pMap.Layer(lngFDOLayerNum)
Set pFDOLayer.Read = pFDOLayer
pMap.ReferenceScale = 0
pMap.MapScale = dblOptimumScale

'Generate graphics for
pFDOLayer.Read.StartGeneratingGraphics Nothing, pScreenDisplay, True, True, False
Set pAnnoElement = pFDOLayer.Read.NextGraphic
Do Until pAnnoElement Is Nothing
    Set pPolygon = New Polygon
    Set pTextElement = pAnnoElement
    Set pTextSymbol = pTextElement.Symbol
    'Temporarily change textsymbol's size
    dblTempTextSize = pTextSymbol.Size
    pTextSymbol.Size = dblTempTextSize * dblAnnoScaleFactor
    Set pTextQuery = pTextSymbol
    Set pTextPointGeo = pAnnoElement.Geometry
    'Setup screen for drawing
    pScreenDisplay.StartDrawing pScreenDisplay.WindowDC, pScreenDisplay.ActiveCache
    'Get ESRI geometry from Text
    Set pPolygon = pTextQuery.GetGeometry(pScreenDisplay.WindowDC, & _
        pDisplayTransform, pTextPointGeo)
    'Ensure geometry is suitable for a feature (sorts inner/outer rings)
    Set pTopoOperator2 = pPolygon
    pTopoOperator2.IsKnownSimple = False
    pPolygon.SimplifyPreserveFromTo
    pScreenDisplay.FinishDrawing

```

```

        'Restore textsymbol size
        pTextSymbol.Size = dblTempTextSize
        'Store geometry in a feature
        Set pFeature = IFeatureClass.CreateFeature
        Set pFeature.Shape = pPolygon
        pFeature.Store
        'Move to next piece of anno and loop
        Set pAnnoElement = pFD0GraphicsRead.NextGraphic
    Loop
    'Restore dataframe's previous extent
    pMap.ReferenceScale = dblReferenceScale
    pMap.MapScale = dblMapScale
    pActiveView.Refresh
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1. 11. 5. 如何设置 Featurelayer 的 Label

本例要完成的功能是将指定层上的某个字段所有值或根据某个表达式选择部分值作为 label 显示出来。

● 要点

IGeoFeatureLayer.AnnotationProperties.Add: 将设置好的 LabelEngineLayerProperties 添加到 IGeoFeatureLayer 中。

ILabelEngineLayerProperties.IsExpressionSimple: 设置显示 Label 的表达式是简单表达式还是复杂表达式。

ILabelEngineLayerProperties.Expression: 设置要显示 Label 的表达式。

IGeoFeatureLayer.DisplayAnnotation: 设置是否要显示 Label。

● 程序说明

先把一个指定的层赋给一个 IGeoFeatureLayer 变量, 生成一个 ILineLabelPosition 的实例, 并设置与它相关的属性、方法; 再生成一个 ILabelEngineLayerProperties 的实例, 设置相关的属性、方法; 最后将生成好的 ILabelEngineLayerProperties 赋给要设定的层。

● 代码

```

Private Sub UIButtonControl1_Click()
    ShowLabel
End Sub

```

```

Public Sub ShowLabel()
    Dim pMxDocument      As IMxDocument
    Dim pGeoFeatureL      As IGeoFeatureLayer
    Dim pLineLabelP       As ILineLabelPosition
    Dim pLabelEngineLP     As ILabelEngineLayerProperties
    Dim pAnnotateLayerP    As IAnnotateLayerProperties

    On Error GoTo ErrorHandler
    Set pMxDocument = ThisDocument
    Set pGeoFeatureL = pMxDocument.FocusMap.Layer(0)
    pGeoFeatureL.AnnotationProperties.Clear
    Set pLineLabelP = New LineLabelPosition
    With pLineLabelP
        .Above = False
        .AtEnd = False
        .Below = False
        .Horizontal = False
        .Inline = True
        .OnTop = True
        .Parallel = True
        .ProduceCurvedLabels = True
    End With
    Set pLabelEngineLP = New LabelEngineLayerProperties
    With pLabelEngineLP
        Set .Symbol = New TextSymbol
        ' 方法一
        .IsExpressionSimple = False
        .Expression = "Function FindLabel ( [ZONENAME27] , [FID]) " _
            & "FindLabel = [ZONENAME27] & [FID]" _
            & " End Function"

        ' 方法二
        .IsExpressionSimple = True
        .Expression = "[ZONENAME27]"
        .BasicOverposterLayerProperties.LineLabelPosition = pLineLabelP
    End With

    Set pAnnotateLayerP = pLabelEngineLP
    With pAnnotateLayerP
        .DisplayAnnotation = True
        Set .FeatureLayer = pGeoFeatureL
        .LabelWhichFeatures = esriVisibleFeatures
        .WhereClause = ""
    End With
    pGeoFeatureL.AnnotationProperties.Add pLabelEngineLP
    pGeoFeatureL.DisplayAnnotation = True
    pMxDocument.ActiveView.Refresh
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

```

1. 11. 6. 如何设置图层显示的透明度

本例要完成的功能是设置指定层的透明度, 使其下面的层也能够被看见。

- 要点

接口 ILayerEffects 中有个属性 Transparency 可以用来设置层的透过率。该属性的值是从 0-100 的, 数值越大, 透过率也越大。

- 程序说明

先将要设置透明度的层的引用赋给一个 ILayerEffects 的实例, 然后再用 ILayerEffects.Transparency 进行透过率的设置。

- 代码

```
Private Sub UIButtonControl1_Click()  
    Dim pMxDocument As IMxDocument  
    Dim pMap As IMap  
    Dim pFeatureLayer As IFeatureLayer  
    Dim pLayerEffects As ILayerEffects  
  
    Set pMxDocument = ThisDocument  
    Set pMap = pMxDocument.FocusMap  
    Set pFeatureLayer = pMap.Layer(0)  
  
    Set pLayerEffects = pFeatureLayer  
    If pLayerEffects.SupportsTransparency Then  
        ' 设置层的透过率  
        pLayerEffects.Transparency = 75  
    EndIf  
    pMxDocument.ActivatedView.Refresh  
End Sub
```

1. 11. 7. 如何过滤层中要显示的 Features

本例要完成的功能是过滤层中要显示的 Features, 即根据指定的条件显示层中的 Features。

- 要点

实现本例的功能用到接口 IFeatureLayerDefinition 中的属性 DefinitionExpression: 设置查询条件来选择出要显示的 Features。

- 程序说明

先得到要过滤层的 FeatureLayer, 再用 IFeatureLayerDefinition.DefinitionExpression 设置显示条件。

- 代码


```

Private Sub UIButtonControl1_Click()
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap
    Dim pActiveView As IActiveView
    Dim pFeatLayer As IFeatureLayer
    Dim pFeatureLayerD As IFeatureLayerDefinition

    Set pMxDocument = Application.Document
    Set pMap = pMxDocument.FocusMap
    Set pActiveView = pMap
    Set pFeatLayer = pMap.Layer(0)
    Set pFeatureLayerD = pFeatLayer
    pFeatureLayerD.DefinitionExpression = "FID <=10"
    pMxDoc.UpdateContents
    pActiveView.Refresh
End Sub

```

1.11.8. 如何在 MapControl 中新建一个 Document 并且保存

本例要完成的功能是把一个选择的 Shape 文件添加到 Mapcontrol 控件中，并在 Mapcontrol 调用 ArcMap 把它保存在一个工程文件(mxd)里。

● 要点

IDocument. Parent: 从已打开的文档中返回 IApplication。

IApplication.NewDocument: 根据设定的模版创建一个新的文档。

IApplication.SaveDocument: 根据指定的路径保存文档。

● 程序说明

先选择一个 Shape 文件加入到 MapControl 控件中 (AddShapeFile)，然后再把它保存到一个工程文件 (CreateNewDoc) 中。

● 代码

```

Private Sub cmdAddLayer_Click()
    On Error GoTo ErrorHandler
    CommonDialog1.ShowOpen
    AddShapeFile (CommonDialog1.FileName)
    Exit Sub

ErrorHandler:
    MsgBox Err.Description
End Sub

Public Sub AddShapeFile(ByVal strFileName As String)
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pFeatureWorkspace As IFeatureWorkspace
    Dim pFeatureLayer As IFeatureLayer
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap

```

```

Dim strPath As String
Dim strName As String

strPath = Mid(strFileName, 1, InStrRev(strFileName, "\"))
strName = Mid(strFileName, InStrRev(strFileName, "\" ) + 1)
strName = Mid(strName, 1, Len(strName) - 4)
'Create a new ShapefileWorkspaceFactory object and open a shapefile folder
Set pWorkspaceFactory = New ShapefileWorkspaceFactory
Set pFeatureWorkspace = pWorkspaceFactory.OpenFromFile(strPath, 0)
'Create a new FeatureLayer and assign a shapefile to it
Set pFeatureLayer = New FeatureLayer
Set pFeatureLayer.FeatureClass = pFeatureWorkspace.OpenFeatureClass(strName)
pFeatureLayer.Name = pFeatureLayer.FeatureClass.AliasName
'Add the FeatureLayer to the focus map
MapControl1.AddLayer pFeatureLayer
MapControl1.Refresh
End Sub

Private Sub cmdSaveMxd_Click()
    If CreateNewDoc() = True Then
        MsgBox "Finish--OK"
    Else
        MsgBox "Finish--Err"
    End If
End Sub

Public Function CreateNewDoc() As Boolean
    On Error GoTo ErrorHandler

    Dim index                As Integer
    Dim pMapControl          As esriMapControl.MapControl
    Dim pApplication         As esriCore.IApplication
    Dim pDocument            As esriCore.IDocument
    Dim pTemplates           As esriCore.ITemplates
    Dim pMxDocument          As esriCore.IMxDocument
    Dim pMap                 As esriCore.IMap
    Dim pActiveView          As esriCore.IActiveView
    Dim pWorkspaceFactory    As esriCore.IWorkspaceFactory
    Dim pObjFactory          As esriCore.IObjectFactory
    Dim pFeatureWorkspace    As esriCore.IFeatureWorkspace
    Dim pMDataSet            As esriCore.IDataset
    Dim pMLayer              As esriCore.ILayer
    Dim pMFeatureLayer       As esriCore.IFeatureLayer
    Dim pFeatureLayer        As esriCore.IFeatureLayer
    Dim pMGeoLayer           As esriCore.IGeoFeatureLayer
    Dim pGeoLayer            As esriCore.IGeoFeatureLayer
    Dim fso                  As New FileSystemObject

    CreateNewDoc = False
    CommonDialog1.ShowSave
    If Not fso.FileExists(CommonDialog1.FileName) Then
        Set pMapControl = MapControl1

```

```

Set pDocument = New esriCore.MxDocument
Set pApplication = pDocument.Parent
Set pTemplates = pApplication.Templates
pApplication.NewDocument False, pTemplates.Item(0)

Set pMxDocument = pDocument
Set pMap = pMxDocument.FocusMap
Set pMap.SpatialReference = pMapControl.SpatialReference
Set pObjFactory = pApplication
For index = pMapControl.LayerCount - 1 To 0 Step -1
    Set pMLayer = pMapControl.Layer(index)
    If Not pMLayer Is Nothing Then
        If TypeOf pMLayer Is IFeatureLayer Then
            Dim l As Long
            Dim str As String

            Set pMFeatureLayer = pMLayer
            Set pMDataSet = pMFeatureLayer
            Set pMGeoLayer = pMFeatureLayer ''
            strFileName = pMDataSet.Workspace.PathName
            l = InStrRev(strFileName, ".")
            str = Right(strFileName, Len(strFileName) - l + 1)
            If UCase(Trim(str)) = ".MDB" Then
                Set pWorkspaceFactory = New esriCore.AccessWorkspaceFactory
                Set pFeatureWorkspace = _
                    pWorkspaceFactory.OpenFromFile(strFileName, 0)
                Set pFeatureLayer = _
                    pObjFactory.Create("esriCore.FeatureLayer")
                Set pFeatureLayer.FeatureClass = pFeatureWorkspace._
                    OpenFeatureClass & (pMFeatureLayer.FeatureClass.AliasName)
                pFeatureLayer.Name = pFeatureLayer.FeatureClass.AliasName
                Set pGeoLayer = pFeatureLayer
                Set pGeoLayer.Renderer = pMGeoLayer.Renderer
            Else
                Set pWorkspaceFactory = New esriCore.ShapefileWorkspaceFactory
                Set pFeatureWorkspace = _
                    pWorkspaceFactory.OpenFromFile(strFileName, 0)
                Set pFeatureLayer = _
                    pObjFactory.Create("esriCore.FeatureLayer")
                Set pFeatureLayer.FeatureClass = pFeatureWorkspace._
                    OpenFeatureClass & (pMFeatureLayer.FeatureClass.AliasName)
                pFeatureLayer.Name = pFeatureLayer.FeatureClass.AliasName
                Set pGeoLayer = pFeatureLayer
                Set pGeoLayer.Renderer = pMGeoLayer.Renderer
            End If
            pMap.AddLayer pFeatureLayer
        End If
    End If
Next
Set pActiveView = pMap
pActiveView.Extent = pMapControl.Extent

pApplication.SaveDocument CommonDialog1.FileName

```

```

        pApplication.Shutdown
    End If
    CreateNewDoc = True
    Exit Function

ErrorHandler:
    MsgBox Err.Description
End Function

```

2. 提高篇

2.1. 缩略图的实现

2.2. FeatureLayer 显示 Symbol 的定制

2.3. 空间查询的综合应用

2.4. 图形编辑的综合应用

2.5. グラフの重ね合わせ表示と印刷

ArcMap 提供了种类丰富, 功能强大的 GRAPH 作成机能, 并可以将作成的 GRAPH 保存为 GRF 文件, 但并不能同时显示多张 GRAPH, 例如同时显示一个地区的地下水位折线图 and 降雨量柱状图, 这时候我们需要自己开发这样的功能。

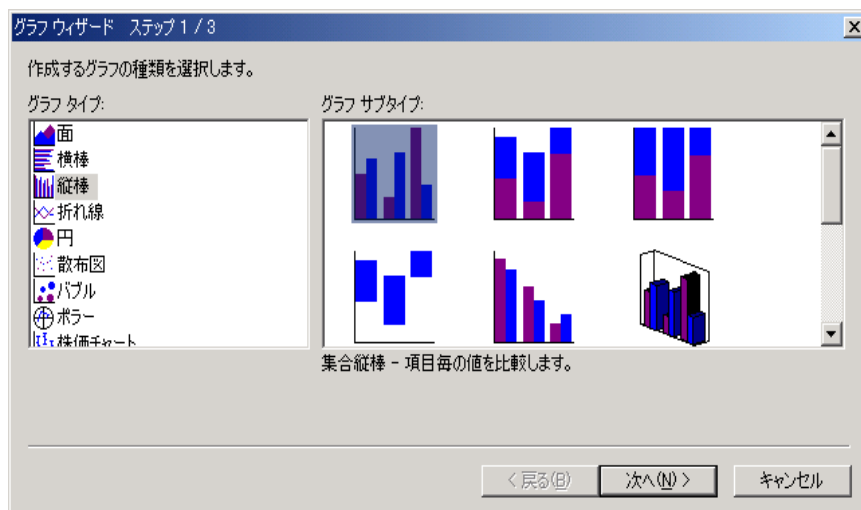


图 16 ArcMap 的 GRAPH 作成机能

本机能的主要内容：

グラフ的打开, 显示, 编辑, 保存.

重ね合わせて印刷, 打印纸张大小和方向的设定, 背景色的设定, GRID 的设定, TITLE 的设定和编辑. Border 的设定. 画面的扩大表示, 移动表示, 全范围表示, 文件的导出等.

画面一覧：

1. グラフの重ね合わせて表示画面的初始表示：



图 17

2. GRF 文件的打开和显示

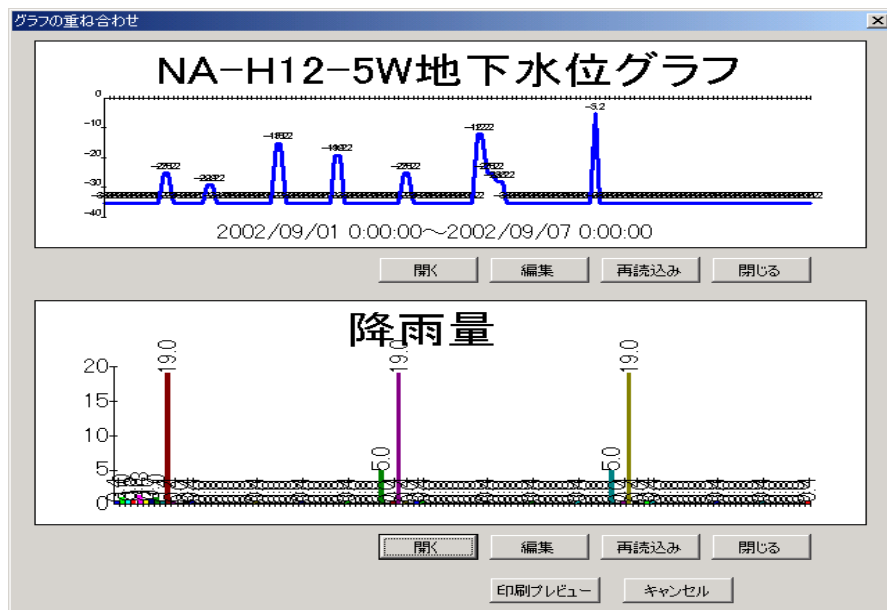


图 18

3. GRAPH 的编辑

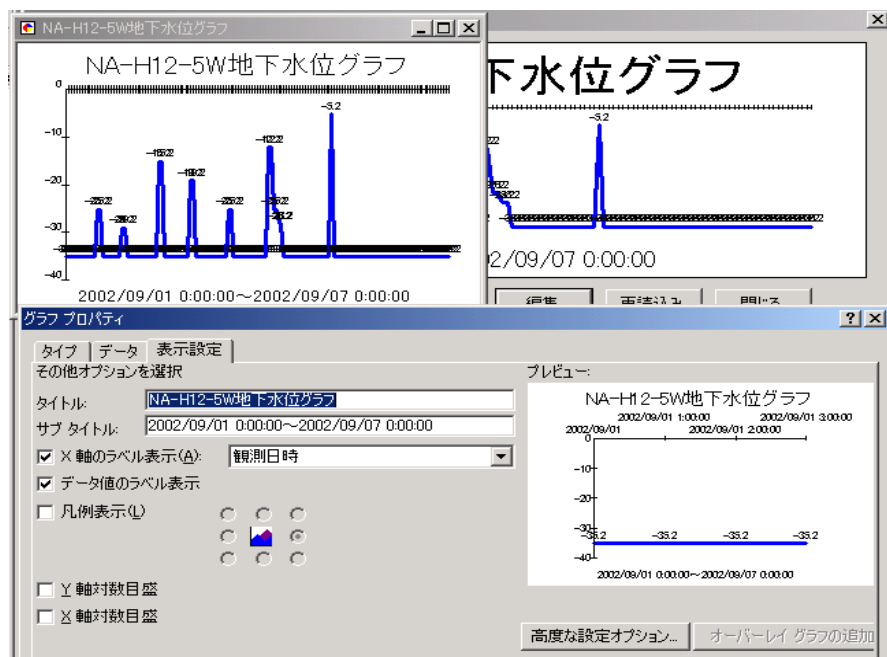


图 19

4. 印刷预览

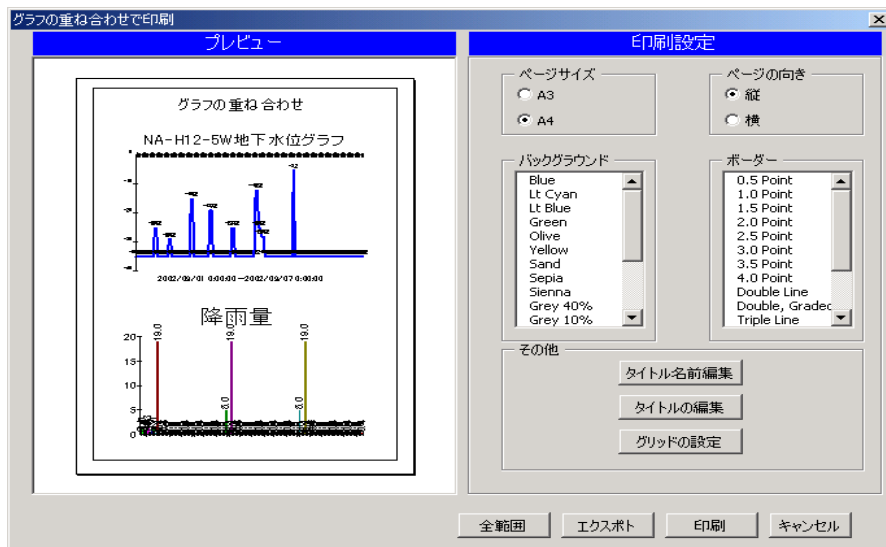


图 20

具体功能请参照实例程序.

以下为部分重要代码:

1. 文件的打开

ShowGraph 関数

```
Private Sub cmdOpen1_Click()  
On Error GoTo EH  
    comFileDialog.CancelError = False  
    With comFileDialog  
        .DialogTitle = "Open File"  
        .DefaultExt = ".grf"  
        .Filter = "Graph Files (*.grf)|*.grf|"  
        .ShowOpen  
    End With  
    g_grfFilePathName1 = comFileDialog.filename  
  
    If g_grfFilePathName1 = "" Then  
        m_IsGraphShow1 = False  
        Exit Sub  
    End If  
  
    m_IsComeStandalone = True  
    ShowGraph  
  
Exit Sub  
EH:  
End Sub
```

```

EH:
    MsgBox m_strFormName & "に 関数「cmdOpen1_Click」 エラー: " & vbCrLf & Err.Description, _
        vbCritical + vbOKCancel, g_strErrTitle
End Sub

Public Sub ShowGraph()
On Error GoTo EH
    Dim pDoc As IMxDocument
    Dim pMap As IMap
    Set pDoc = ThisDocument
    Set pMap = pDoc.FocusMap

    ' Create Graph
    ' Dim pDataGraph As IDataGraph
    Dim pDataGraphProperties As IDataGraphProperties
    Dim pGraphWindow As IDataGraphWindow
    Dim pDataGraphs As IDataGraphs

    Set g_pDataGraph1 = New DataGraph
    g_pDataGraph1.LoadFromFile g_grfFilePathName1

    g_pDataGraph1.ReloadAlways = True

    g_pDataGraph1.Attach MapControl1.hwnd

    g_pDataGraph1.Draw
    m_IsGraphShow1 = True

Exit Sub
EH:
    MsgBox m_strFormName & "に 関数「ShowGraph」 エラー: " & vbCrLf & Err.Description, _
        vbCritical + vbOKCancel, g_strErrTitle
End Sub

```

2. グラフの編集

```

Private Sub cmdEdit1_Click()
On Error GoTo EH
    Dim pDataGraphProperties As IDataGraphProperties
    Dim pGraphWindow As IDataGraphWindow
    Dim pDataGraphs As IDataGraphs

    If m_IsGraphShow1 = False Then Exit Sub
    Dim pDoc As IMxDocument
    Dim pMap As IMap
    Set pDoc = ThisDocument
    Set pMap = pDoc.FocusMap

    Set pGraphWindow = New DataGraphWindow
    Set pGraphWindow.DataGraph = g_pDataGraph1
    Set pGraphWindow.Application = Application

```



```

'Add the graph to the system
Set pDataGraphs = pDoc 'QI
pDataGraphs.Reset

pDataGraphs.Add g_pDataGraph1
ShowGraph
Set pDataGraphProperties = g_pDataGraph1 'QI

pGraphWindow.Show True

Set pGraphWindow = Nothing
Set pDataGraphs = Nothing
Exit Sub
EH:
MsgBox m_strFormName & "に 関数「cmdEdit1_Click」 エラー: " & vbCrLf & Err.Description, _
vbCritical + vbOKCancel, g_strErrTitle
End Sub

```

3. 印刷プレビュー

```

Private Sub cmdPrintPreview_Click()
On Error GoTo EH
If m_IsGraphShow1 = False And m_IsGraphShow2 = False Then
Exit Sub
End If
If m_IsGraphShow1 = True And m_IsGraphShow2 = True Then
Set g_PrintGraph1 = New DataGraph
Set g_PrintGraph1 = g_pDataGraph1

Set g_PrintGraph2 = New DataGraph
Set g_PrintGraph2 = g_pDataGraph2

frmTwoGraphPrint.Show
End If
Exit Sub
EH:
MsgBox m_strFormName & "に 関数「cmdPrintPreview_Click」 エラー: " & vbCrLf & _
& Err.Description, vbCritical + vbOKCancel, g_strErrTitle
End Sub

```

4. グラフ合わせて印刷画面の初期化

```

Private Sub UserForm_Initialize()
On Error GoTo EH
Dim pMapFrame As esriCore.IMapFrame
Dim pElement As esriCore.IElement
Dim pEnvelope As esriCore.IEnvelope

PageLayoutControll1.Page.Orientation = 1 'Portrait:1 ;Landscape:2
PageLayoutControll1.PageLayout.Page.Orientation = 1
PageLayoutControll1.PageLayout.Page.FormID = esriPageFormA4

```

```

If Not PageLayoutControl1.Printer Is Nothing Then
    PageLayoutControl1.Printer.Paper.Orientation = 1
End If

Set pMapFrame = PageLayoutControl1.GraphicsContainer._
    FindFrame(PageLayoutControl1.ActiveView.FocusMap)
Set pElement = pMapFrame

Set pEnvelope = New esriCore.Envelope
pEnvelope.PutCoords 1, 1, 20, 29
pElement.Geometry = pEnvelope
PageLayoutControl1.Refresh

Dim pDGElem As IDataGraphElement
Set pDGElem = New DataGraphElement
Set pDGElem.DataGraph = g_PrintGraph1
Dim pElem As IElement
Set pElem = pDGElem
Dim pGc As IGraphicsContainer
Set pGc = PageLayoutControl1.GraphicsContainer

Dim pEnv As IEnvelope
Set pEnv = New Envelope
pEnv.xMin = 2
pEnv.xMax = 19
pEnv.yMin = 14
pEnv.yMax = 26
pElem.Geometry = pEnv

pGc.AddElement pElem, 0

Dim pDGElem2 As IDataGraphElement
Set pDGElem2 = New DataGraphElement
Set pDGElem2.DataGraph = g_PrintGraph2
Dim pElem2 As IElement
Set pElem2 = pDGElem2

Dim pEnv2 As IEnvelope
Set pEnv2 = New Envelope

pEnv2.xMin = 2
pEnv2.xMax = 19
pEnv2.yMin = 2
pEnv2.yMax = 13
pElem2.Geometry = pEnv2

pGc.AddElement pElem2, 0

PageLayoutControl1.Refresh
optA4.Value = True
optPortrait.Value = True
UpdateArrayAndListBoxFromStyleGallery "Backgrounds", _
    m_pStylesArray(1), lstBackground

```

```

UpdateArrayAndListBoxFromStyleGallery "Borders", m_pStylesArray(2), lstBorder

g_strTitle = "グラフの重ね合わせ"
'Create an element and grab hold of the IElement interface
Set m_pTitleElement = New TextElement
'Grab hold of the ITextElement interface through the element
Set m_pTextElement = m_pTitleElement
'Create a text symbol and grab hold of the ITextSymbol interface
Set m_pTextSymbol = New TextSymbol
'Set text symbol properties
m_pTextSymbol.Text = g_strTitle
m_pTextSymbol.Size = 30
'Set geometry property
m_pTitleElement.Geometry = CreateEnvelope(140, 64, 180, 30)
'Set text element properties
m_pTextElement.Symbol = m_pTextSymbol
m_pTextElement.Text = g_strTitle
'Add the element
PageLayoutControl1.AddElement m_pTitleElement
'Refresh the new graphic element
PageLayoutControl1.Refresh esriViewGraphics, m_pTitleElement
'Remove All Parameters
Set pMapFrame = Nothing
Set pElement = Nothing
Set pEnvelope = Nothing
Set pDGElem = Nothing
Set pElem = Nothing
Set pGc = Nothing
Set pEnv = Nothing
Set pDGElem2 = Nothing
Set pElem2 = Nothing
Set pEnv2 = Nothing

Exit Sub
EH:
MsgBox m_strFormName & "に関数「UserForm_Initialize」エラー: " & vbCrLf_
    & Err.Description, vbCritical + vbOKCancel, g_strErrTitle
End Sub

```

5. エクスポート

```

Private Sub cmdExport_Click()
On Error GoTo EH
comFileDialog.CancelError = False
With comFileDialog
.DialogTitle = "Open File"
.DefaultExt = ".jpg"
.Filter = "Jpg Files(*.jpg)|*.jpg|PDF Files(*.pdf)|_
    |*.pdf|BMP Files(*.bmp)|*.bmp|TIFF Files(*.tif)|*.tif"
.ShowOpen
End With

```

```

Dim ExportName As String
ExportName = comFileDialog.filename
Exit Sub
End If

Dim pActiveView As IActiveView
Dim pExporter As IExporter
Dim pEnv As IEnvelope
Dim exportFrame As tagRECT
Dim dpi As Integer
Dim xMin As Double
Dim yMin As Double
Dim xMax As Double
Dim yMax As Double
Dim hdc As Long

Set pActiveView = PageLayoutControll.ActiveView
Set pEnv = New Envelope
If basPubFileUtil.GetFileExtention(ExportName) = ".jpg" Then
    Set pExporter = New JpegExporter
    exportFrame = pActiveView.exportFrame
    pEnv.PutCoords exportFrame.Left, exportFrame.Top, _
        exportFrame.Right, exportFrame.bottom
    dpi = 300 'Set a high resolution

    With pExporter
        .PixelBounds = pEnv
        .ExportFileName = ExportName
        .Resolution = dpi
    End With
ElseIf basPubFileUtil.GetFileExtention(ExportName) = ".pdf" Then
    Set pExporter = New PDFExporter
    exportFrame = pActiveView.exportFrame
    pEnv.PutCoords exportFrame.Left, exportFrame.Top, _
        exportFrame.Right, exportFrame.bottom
    dpi = 300 'Set a high resolution

    With pExporter
        .PixelBounds = pEnv
        .ExportFileName = ExportName
        .Resolution = dpi
    End With
ElseIf basPubFileUtil.GetFileExtention(ExportName) = ".bmp" Then
    Set pExporter = New DibExporter
    exportFrame = pActiveView.exportFrame
    pEnv.PutCoords exportFrame.Left, exportFrame.Top, _
        exportFrame.Right, exportFrame.bottom
    dpi = 100

    With pExporter
        .PixelBounds = pEnv
        .ExportFileName = ExportName
    End With

```

```

ElseIf basPubFileUtil.GetFileExtention(ExportName) = "tif" Then
    Set pExporter = New TiffExporter
    exportFrame = pActiveView.exportFrame
    pEnv.PutCoords exportFrame.Left, exportFrame.Top, _
        exportFrame.Right, exportFrame.bottom
    dpi = 100

    With pExporter
        .PixelBounds = pEnv
        .ExportFileName = ExportName
    End With
End If

Set pEnv = pExporter.PixelBounds
pEnv.QueryCoords xmin, ymin, xmax, ymax
exportFrame.Left = xmin
exportFrame.Top = ymin
exportFrame.Right = xmax
exportFrame.bottom = ymax

'Do the export
hdc = pExporter.StartExporting
pActiveView.Output hdc, dpi, exportFrame, Nothing, Nothing
pExporter.FinishExporting
Exit Sub
EH:
    MsgBox m_strFormName & "に 関数「cmdExport_Click」 エラー: " & vbCrLf & Err.Description, _
        vbCritical + vbOKCancel, g_strErrTitle
End Sub

```

2.6. バッファ処理

ArcMap 自身具有一个功能叫” バッファ処理”, 画面如下:

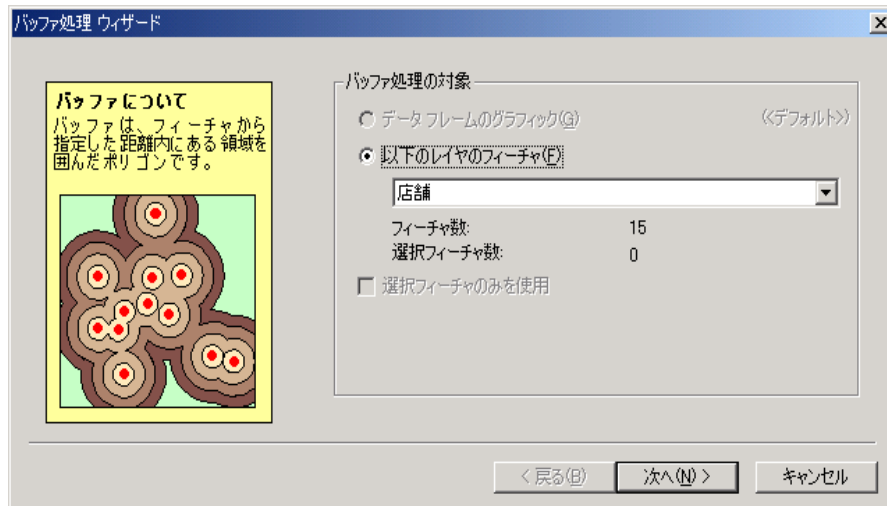


图 21

这个处理的功能和使用 ItopologicalOperator 接口的 Buffer 方法有一定的相似之处.

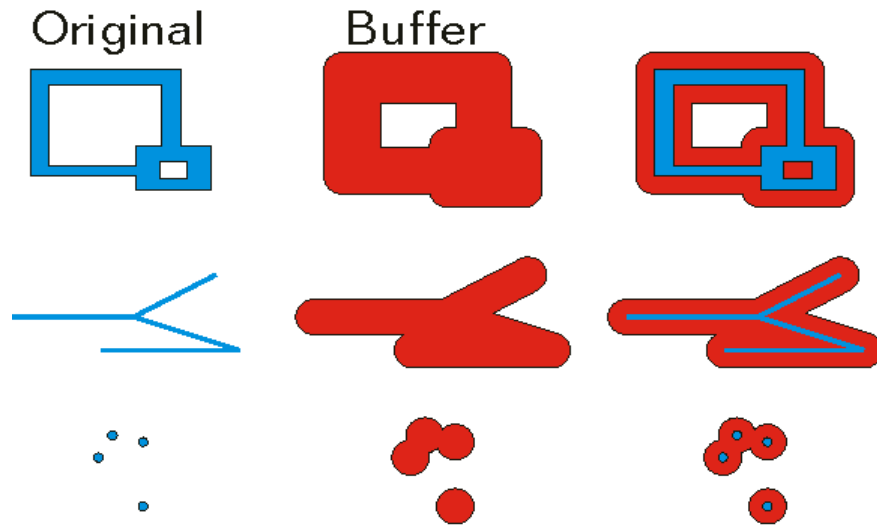


图 22

但是, 当处理的 Shape 文件是带有经纬度坐标系的文件时, 2 者处理的结果是不一样的.

例如, 一个点的 Shape 文件是带有投影坐标系的文件时, 对这个文件做バッファ处理后得到的结果为正圆:

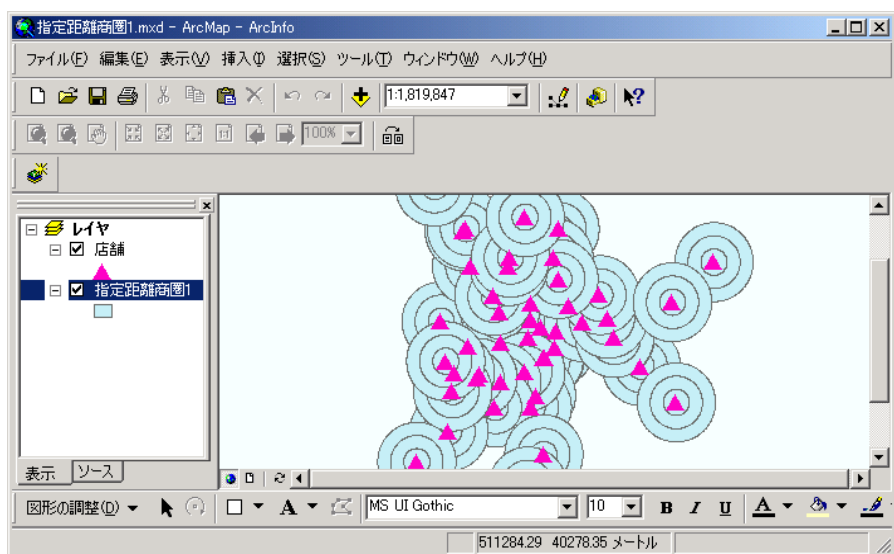


图 23

而这个点的 Shape 文件是带有经纬度坐标系的文件时,对这个文件做バッファ处理后得到的结果为椭圆:

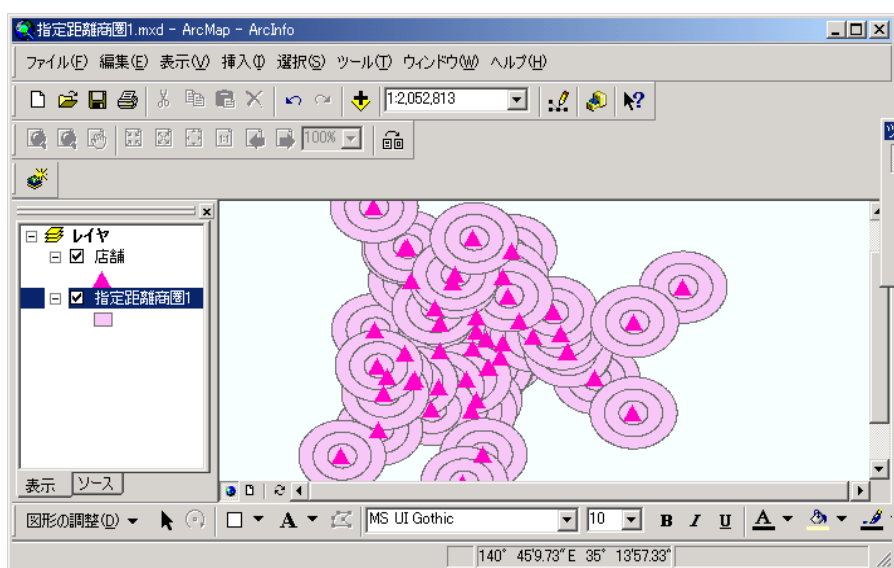


图 24

下面我们以实例来介绍如何实现一个类似的功能:

首先,新建一个工程,添加一个点的 Shape 文件,并使用宏添加一个按钮.

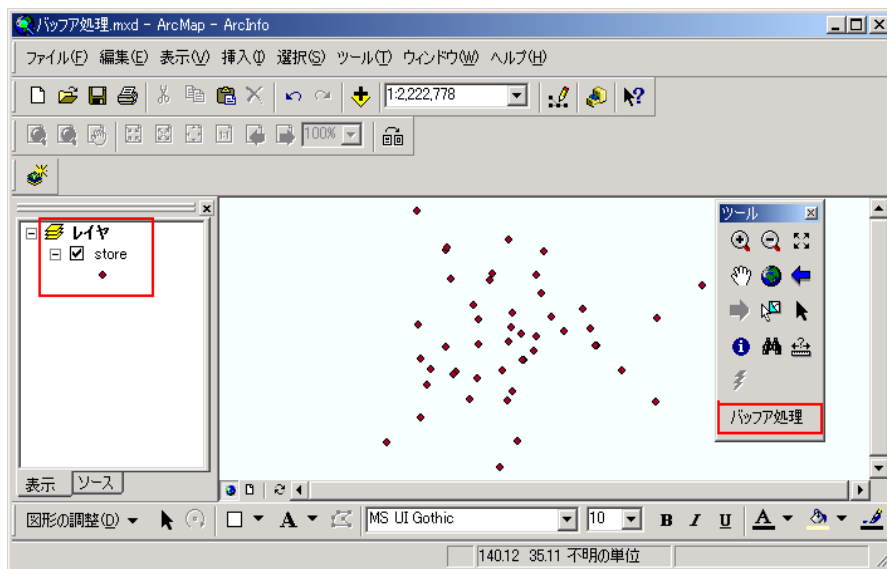


图 25

然后, 添加如下处理代码:

```
Private Sub UIButtonControll_Click()
    Dim pMXDocument As IMxDocument
    Dim pMap As IMap
    Dim pActiveView As IActiveView
    Dim pFeatureClass As IFeatureClass
    Dim pFeatureLayer As IFeatureLayer
    Dim pBufferFeatureLayer As IFeatureLayer
    Set pMXDocument = ThisDocument
    Set pMap = pMXDocument.FocusMap
    Set pFeatureLayer = pMap.Layer(0)

    Set pBufferFeatureLayer = New FeatureLayer
    If MakeBuffer(pFeatureLayer, pBufferFeatureLayer) = False Then
        Exit Sub
    Else
        pMap.AddLayer pBufferFeatureLayer
    End If
End Sub
```

バッファ処理関数:

```
Public Function MakeBuffer(ByVal pPointFeatureLayer As IFeatureLayer, ByRef
pOutFeatureLayer As IFeatureLayer) As Boolean
On Error GoTo ErrorHandler
    Dim pPointFeatureClass As IFeatureClass
    Dim pPointGeoDataset As IGeoDataset
    Dim pPointSpReference As ISpatialReference
    Dim PolyBufferType As esriBufferType
```



```

Dim BufferUnits As esriUnits
Dim MapUnits As esriUnits
Dim pSourceSpatialReference As ISpatialReference
Dim pBufferSpatialReference As ISpatialReference
Dim pOutputSpatialReference As ISpatialReference

Dim BufferSpatialReferenceType As esriBufferSpatialReferenceType
Dim TargetSpatialReferenceType As esriBufferSpatialReferenceType

Dim pFeatureCursorBuffer As IFeatureCursorBuffer
Dim pFeatureCursor As IFeatureCursor
Dim pFeatureClass As IFeatureClass
Dim pFeatureClassName As IFeatureClassName
Dim pWorkspaceName As IWorkspaceName
Dim pDatasetName As IDatasetName
Dim DissolveBuffers As Boolean
Dim pBufferProcessingParameter As IBufferProcessingParameter
Dim pFeatureCursorBuffer2 As IFeatureCursorBuffer2

Dim pTrackCancel As ITrackCancel

Dim pWorkspaceFactory As IWorkspaceFactory
Dim pFeatureWorkspace As IFeatureWorkspace
Dim pOutFCClass As IFeatureClass

'Check to see whether file exists, If Exist, Then delete them.
If FileExists(OUTFILEPATH & "\" & OUTFILENAME & ".shp") Then _
    Delete_File (OUTFILEPATH & "\" & OUTFILENAME & ".shp")
If FileExists(OUTFILEPATH & "\" & OUTFILENAME & ".shx") Then _
    Delete_File (OUTFILEPATH & "\" & OUTFILENAME & ".shx")
If FileExists(OUTFILEPATH & "\" & OUTFILENAME & ".dbf") Then _
    Delete_File (OUTFILEPATH & "\" & OUTFILENAME & ".dbf")
If FileExists(OUTFILEPATH & "\" & OUTFILENAME & ".prj") Then _
    Delete_File (OUTFILEPATH & "\" & OUTFILENAME & ".prj")

Set pPointFeatureClass = pPointFeatureLayer.FeatureClass
Set pPointGeoDataset = pPointFeatureClass
Set pPointSpReference = pPointGeoDataset.SpatialReference

Set pFeatureCursor = pPointFeatureClass.Search(Nothing, False)

Set pSourceSpatialReference = pPointSpReference
Set pOutputSpatialReference = pPointSpReference

Set pFeatureClassName = New FeatureClassName
Set pDatasetName = pFeatureClassName

pDatasetName.Name = OUTFILENAME & ".shp"
Set pWorkspaceName = New WorkspaceName

pWorkspaceName.PathName = OUTFILEPATH
pWorkspaceName.WorkspaceFactoryProgID = "esriCore.shapefileworkspacefactor.1"

```

```

Set pDatasetName.WorkspaceName = pWorkspaceName

pFeatureClassName.FeatureType = esriFTSimple
pFeatureClassName.ShapeType = esriGeometryPolygon
pFeatureClassName.ShapeFieldName = "Shape"
Set pFeatureCursorBuffer = New FeatureCursorBuffer

PolyBufferType = esriBufferOutsideIncludeInside
DissolveBuffers = False
BufferUnits = esriKilometers
MapUnits = esriDecimalDegrees
BufferSpatialReferenceType = esriFeatureSetOptimizedSpatialReference
TargetSpatialReferenceType = esriMapSpatialReference

Set pBufferProcessingParameter = pFeatureCursorBuffer
Set pFeatureCursorBuffer2 = pFeatureCursorBuffer
Set pBufferProcessingParameter.FeatureClass = pPointFeatureClass
Set pFeatureCursorBuffer.FeatureCursor = pFeatureCursor

pFeatureCursorBuffer.PolygonBufferType = PolyBufferType
'バッファ間の境界線を削除しますか?
pFeatureCursorBuffer.Dissolve = False
'バッファ距離の単位
pFeatureCursorBuffer.Units(MapUnits) = BufferUnits
'バッファを作成する方法
pFeatureCursorBuffer.ValueDistance = 15
pBufferProcessingParameter.AdjustCirclesForProjection = True
pBufferProcessingParameter.SimplifyShapes = False
pBufferProcessingParameter.GenerateRings = True
pBufferProcessingParameter.InputHasPolygons = False

Set pFeatureCursorBuffer2.SourceSpatialReference= pSourceSpatialReference
pBufferProcessingParameter.BufferSpatialReference= BufferSpatialReferenceT pe
Set pFeatureCursorBuffer2.BufferSpatialReference = Nothing

pBufferProcessingParameter.TargetSpatialReference= TargetSpatialReferenceT pe
Set pFeatureCursorBuffer2.TargetSpatialReference = pOutputSpatialReference
Set pFeatureCursorBuffer2.DataFrameSpatialReference = pOutputSpatialRefere ce
Set pTrackCancel = New CancelTracker
Set pFeatureCursorBuffer.CancelTrack = pTrackCancel
pBufferProcessingParameter.SaveAsGraphics = False
pFeatureCursorBuffer.Buffer pFeatureClassName

Set pFeatureClassName = Nothing
Set pDatasetName = Nothing
Set pWorkspaceName = Nothing
Set pFeatureCursorBuffer = Nothing
Set pBufferProcessingParameter = Nothing
Set pFeatureCursorBuffer2 = Nothing
Set pTrackCancel = Nothing
Set pFeatureCursor = Nothing
Set pFeatureClass = Nothing
Set pSourceSpatialReference = Nothing

```

```

Set pOutputSpatialReference = Nothing

Set pWorkspaceFactory = New ShapefileWorkspaceFactory
Set pFeatureWorkspace = pWorkspaceFactory.OpenFromFile(OUTFILEPATH, 0)
Set pOutFCClass = pFeatureWorkspace.OpenFeatureClass(OUTFILENAME)

Set pOutFeatureLayer.FeatureClass = pOutFCClass
pOutFeatureLayer.Name = "バッファ処理結果"
MakeBuffer = True
Exit Function
ErrorHandler:
MsgBox Err.Description
MakeBuffer = False
End Function

```

运行结果为:

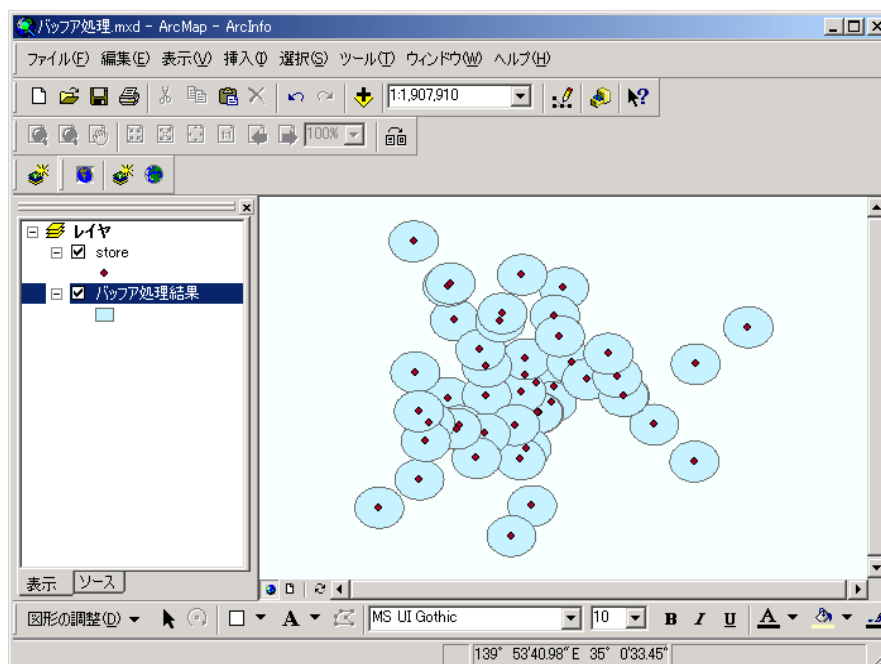


图 26

2.7. Voronio 作成

2.8. 数据处理加速—地图分块处理

2. 9. MapControl 的使用

2. 10. 运用 PageLayout 控件打印图形

附录 ArcGIS 的 GUID 一览表