

MapX应用讲义

一、加载地图数据

1、TAB 的数据分为两种数据：地图数据（Layers）、属性数据（Datasets）。关系：不可分割的一个数据集的两部分。

2、数据加载：GST 文件由 GeosetManager40.exe 程序生成。在程序使用 gsT 文件：Map1.Geoset=Filepath+FileName

3、问题：GST 文件加载后，只是默认将地图数据加载，属性数据另外需要使用单独的命令进行加载，否则对属性数据的操作全部非法。加载：Map1.Datasets.ADD 属性数据集名称

4、另一种加载方式：使用 LayerInfo 对象，这种方式下加载地图数据源的地图集和属性集均可直接使用。示例：

```
dim LayerInfo as MapXLib.LayerInfo
dim Lyr as Mpxlib.layer
```

```
LayerInfo.Type = miLayerInfoTypeTab      ‘加载表的类型
LayerInfo.AddParameter "FileSpec", FilePath + LayerName + ".TAB"      ‘加载表的全路径名
LayerInfo.AddParameter "NAME", LayerName      ‘地图集的别名
LayerInfo.AddParameter "AutoCreateDataset", 1      ‘是否加载属性数据集
LayerInfo.AddParameter "datasetname", LayerName      ‘属性数据集别名
```

```
MainMap.Layers.Add LayerInfo      ‘加载到指定的 MapX 对象中，立即可直接使用
```

5、第三种加载数据方式：GST 文件+ LayerInfo 方式。示例：

```
使用两个 MapX 对象：MainMap、TempMap
TempMap.Geoset=GST 文件
MainMap.geoset=""
TempMap.Refresh
```

```
For I=1 to TempMap.Layers.Count
FileName=TempMap.Layers.Item(I).Filespec
‘直接引用 LayerInfo 方式加载地图数据到 MainMap
Next
```

二、创建地图对象

必要：创建地图对象，必须使用 FeatureFactory 对象

1、创建一个点对象

点对象有一个坐标点（X，Y），点对象变量是 Point 类型，点对象的样式（Style）是符号样式。

```
Dim Pnt AS MapXLib.Point
```

```

Dim FeaFac AS MapXLib.FeatureFactory
Dim Lyr AS MapXLib.Layer
Dim Ftr AS MapXLib.Feature
Dim NewStyle AS MapXLib.Style

‘绑定
SET Lyr=MainMap.Layers.Item(LayerName)
SET FeaFac=mainmap.featurefactory
‘设置点对象样式
With NewStyle
    .SymbolType = miSymbolTypeBitmap
    .SymbolBitmapSize = 24
    .SymbolBitmapTransparent = False
    .SymbolBitmapName = "YIEL2-32.BMP"
End With
Mainmap.AutoRedraw=False ‘禁止自动刷新
Lyr.Editable=True ‘置当前图层为可写状态
‘创建点对象
pnt.set X1,Y1
‘添加进当前图层
Set Ftr=FeaFac.CreateSymbol (Pnt,Newstyle) ‘创建符号
‘Set Ftr=FeaFac.CreateSymbol (Pnt,MainMap.DefaultStyle)
‘添加
Lyr.AddFeature Ftr
Lyr.Refresh
Mainmap.AutoRedraw=True
Lyr.Editable=False
‘释放
SET Pnt = Nothing
SET FeaFac = Nothing
SET Lyr = Nothing
SET Ftr = Nothing
‘以上代码放在 MapX 的 ToolUsed 事件下

```

单独修改某个图元的样式：SET Ftr.Style=NewStyle，再用 Update 即可

2、 创建一个线矩形

```

Dim Pnts AS MapXLib.Points

With NewStyle
    .LineColor=Rgb(0, 0,255)
End With
‘第一个点
Pnt.Set X1,Y1
Pnts.add Pnt

```

```

‘第二个点
Pnt.Set X2, Y1
Pnts.add Pnt
‘第三个点
Pnt.Set X2, Y2
Pnts.add Pnt
‘第四个点
Pnt.Set X1, Y2
Pnts.add Pnt
‘第五个点
Pnt.Set X1, Y1
Pnts.add Pnt

```

```

‘创建线矩形
SET Ftr=FeaFac.CreateLine(Pnts, NewStyle)
Lyr.AddFeature Ftr
Lyr.Refresh

```

3、上面创建对象中存在的问题：并未对其数据数据进行赋值
创建对象的同时创建其数据集合

```

Dim Pnt AS MapXLib.Point
Dim FeaFac AS MapXLib.FeatureFactory
Dim Lyr AS MapXLib.Layer
Dim Ftr AS MapXLib.Feature
Dim NewStyle AS MapXLib.Style
Dim ds AS MapXLib.Dataset
Dim Flds AS MapXLib.Fields

```

```

‘绑定
SET Lyr=MainMap.Layers.Item(LayerName)
SET ds=Lyr.Datasets.Item(1)
Set Flds=ds.Fields
SET FeaFac=mainmap.featurefactory
‘设置点对象样式
With NewStyle
    .SymbolType = miSymbolTypeBitmap
    .SymbolBitmapSize = 24
    .SymbolBitmapTransparent = False
    .SymbolBitmapName = "YIEL2-32.BMP"
End With
Mainmap.AutoRedraw=False ‘禁止自动刷新
Lyr.Editable=True ‘置当前图层为可写状态
‘创建点对象
pnt.set X1, Y1

```

```

‘创建图形
Set Ftr=FeaFac. CreateSymbol (Pnt,Newstyle) ‘创建符号
‘Set Ftr=FeaFac. CreateSymbol (Pnt,MainMap.DefaultStyle)
‘设置属性
For I=1 to Flds.Count
Lyr.KeyFields=Flds.Item(i).Name
Ftr.KeyValue=ValueStr(I) ‘这里并没有对字段类型进行判断
Next
‘另外一种方法：使用 RowValues 和 RowValue 对象
‘添加
Lyr.AddFeature Ftr
Lyr.Refresh
Mainmap.AutoRedraw=True
Lyr.Editable=False
‘释放
SET Pnt = Nothing
SET FeaFac = Nothing
SET Lyr = Nothing
SET Ftr = Nothing

```

```

SET ds = Nothing
SET Flds = Nothing

```

4、创建表

(1) 临时表：

- A、用 MainMap.Layers.CreateLayer 方法创建临时表。但这个临时表只有一个字段：GeoName(Char 24)。程序运行过程中该表存放位置为系统临时文件夹下
- B、使用 LayerInfo 对象创建临时表，可以指定字段。示例：

```

Dim Lyr As MapXLib.Layer
Dim LayerInfo As New MapXLib.LayerInfo
Dim Flds As New MapXLib.Fields

```

‘字段定义

```

Flds.AddStringField "ID", 12
Flds.AddStringField "Name", 50
Flds.AddNumericField "Deptch", 12, 2
Flds.AddIntegerField "Length"

```

```

&acute;

```

```

LayerInfo.Type = miLayerInfoTypeTemp
LayerInfo.AddParameter "FileSpec", FileName
LayerInfo.AddParameter "NAME", LayerName
LayerInfo.AddParameter "Fields", Flds

```

```
Set Lyr = MainMap.Layers.Add(LayerInfo, 1)
```

```
Set Lyr = Nothing
```

```
Set LayerInfo = Nothing
```

(2) 创建永久表

```
Dim Lyr As MapXLib.Layer
```

```
Dim LayerInfo As New MapXLib.LayerInfo
```

```
Dim Flds As New MapXLib.Fields
```

```
Flds.AddStringField "ID", 12
```

```
Flds.AddStringField "Name", 50
```

```
Flds.AddNumericField "Deptch", 12, 2
```

```
Flds.AddIntegerField "Length"
```

```
&acute;
```

```
LayerInfo.Type = miLayerInfoTypeNewTable
```

```
LayerInfo.AddParameter "FileSpec", FilePath + "" + FileName
```

```
LayerInfo.AddParameter "NAME", LayerName
```

```
LayerInfo.AddParameter "Fields", Flds
```

```
Set Lyr = MainMap.Layers.Add(LayerInfo, 1)
```

```
Set Lyr = Nothing
```

```
Set LayerInfo = Nothing
```

5、 创建工具句柄

系统已经定义工具句柄都以整数（包括 16 进制）常数存在，句柄号大于 1000 和小于 12 基本

都为系统使用。

A. 定义常数：必须为全局变量

```
Global Const CreateSymbolTool = 13 &acute;创建节点
```

```
Global Const CreateLineTool = 15 &acute;创建管线
```

```
Global Const InfoTipTool = 16 &acute;信息工具
```

```
Global Const MoveFeature = 17 &acute;移动地图
```

```
Global Const ScaleDistanceTool = 18 &acute;测量两点间的距离
```

B. 使用 CreateCustomTool 创建新的工具句柄：

```
MainMap.CreateCustomTool CreateSymbolTool, miToolTypePoint, miSymbolCursor
```

```
MainMap.CreateCustomTool CreateLineTool, miToolTypeLine, miCrossCursor
```

```
MainMap.CreateCustomTool InfoTipTool, miToolTypePoint, miCrossCursor
```

```
MainMap.CreateCustomTool MoveFeature, miToolTypeLine, miPanCursor
```

```
MainMap.CreateCustomTool ScaleDistanceTool, miToolTypeLine, miPanCursor
```

C. 如何使用?

在 Map 对象的 ToolUsed 事件的 ToolNum 参数为当前所激活的工具使当

前操作指向某行为: MainMap.CurrentTool=工具句柄号, 如放大: MainMap.CurrentTool=miZoomInTool, 移动图元: MainMap.CurrentTool=MoveFeature

操作具体的工具句柄时, 执行该捕捉到的工具句柄的代码:

在 ToolUsed 事件中:

```
Select Case ToolNum
  Case MoveFeature
    '执行代码
End Select
```

删除图元: Lyr.DeleteFeature Ftr

三、查询

1、 属性查找。Find、Search 方法: 注意的是 Find 方法只支持 TAB 表文件, 不支持空间数据表。

Find : 与 FoxPro 中 Locate 定位命令想类似。

Search: 支持 SQL 语句。写法: 仅指 SQL 语句的 WHERE 部分, 且 From 语句中只能有一个表

— 一仅对单表进行操作: Select * from LayerName WHERE ID LIKE “%北京%”

示例:

A、查找

Dim Ftrs AS MapXlib.Features ‘图元集合

```
SET Ftrs=Lyr.Search(“ID LIKE “ “%北京%” ” ” )
```

```
For I=1 to Ftrs.Count
```

```
    ‘执行语句
```

```
Next
```

B、高亮显示

Lyr.Selection.Replace Ftrs ‘将当前查询所得的结果集全部高亮显示 (隐含执行: Lyr.ClearSelection 语句) ——加入 selection 集合

闪烁: 不能用 Selection, 否则会对整个屏幕进行整个刷新 (抖动)。使用更新 Style 的方法进行选定图元的闪烁。

记载图元的老样式: Set Oldstyle=Ftr.Style

Lyr.Selection.Add Ftrs ‘将当前查询所得的结果集添加到已有的结果集中, 再全部高亮显示

C、对查询的结果集进行属性修改

示例程序: 完成的是 Professional 中信息工具功能

```

Dim ds AS MapXlib.Dataset
Dim Flds AS MapXlib.Fields
Dim Ftr AS MapXlib.Feature

Set Lyr=MainMap.Layers.Item(LayerName)
Set ds=Lyr.Datasets.item(1)
Set Flds=ds.Fields
    ‘查找
SET Ftrs=Lyr.Search(“ID LIKE “ “%北京%” ” ” )
If Ftrs.count=0 then exit sub

    ‘读取属性值
For I=1 to Ftrs.Count
    Set Ftr=Ftrs.Item(I)
    For j=1 to Flds.count
FldsName(J)=Flds.Item(J).Name    ‘字段列表
Lyr.KeyField=FldsName(J)
    ValueStr(I, J)=Ftr.KeyValue    ‘值列表
    Next
Next

    ‘修改属性
MainMap.AuyoRedraw=False
Lyr.Editable=True

For j=1 to Flds.count
Lyr.KeyField= Flds.Item(J).Name
Ftrs.Item(j).KeyValue =ValueStr(J)    ‘更新值列表
Ftrs.Item(j).Update True
Next

Lyr.Refresh
Lyr.Editable=False
MainMap.AuyoRedraw=True

    ‘修改样式
Dim NewStyle AS MapXlib.Style

With NewStyle
    ‘设置样式
End With

MainMap.AuyoRedraw=False
Lyr.Editable=True

```

```

For i=1 to Ftrs.count
Set Ftr =Ftrs.Item(I)
SET Ftr.Style=NewStyle      ‘更新样式
Ftr.Update True
Next

```

```

Lyr.Refresh
Lyr.Editable=False
MainMap.AuyoRedraw=True

```

2、空间查找

² 点查找: SearchAtPoint, 结果集为 Features 类型
Dim Pnt AS MapXlib.Point

```

Pnt.Set X,Y
Set Ftrs=Lyr.SearchAtPoint(Pnt,miSearchResultAll)
For I=1 to Ftrs.Count
    ‘执行语句
Next

```

注意: 点查找时, 一般情况下结果集在一个以上的图层都存在。所以取值时应分别提取

² 园查找: 在临时图层上画一个不保存的圆, 然后查找被这个圆所包含的所有图层的图元对象。

```

Dim Pnt AS MapXlib.Point
Dim TempCir AS MapXlib.Feature
Dim FeaFac AS MapXLIB.featurefactory

```

```

Pnt.Set X,Y
Set tempcir=FeaFac.CreateCircularRegion(miCircleTypeMap ,Pnt,1, MainMap.MapUnit,,)

```

‘miSearchTypeCentroidWithin : 中心点包含
‘miSearchTypePartiallyWithin : 部分包含
‘miSearchTypeEntirelyWithin : 全部包含

```

Set Ftrs=Lyr.SearchWithinFeature (TempCir, miSearchTypePartiallyWithin)

```

```

For I=1 to Ftrs.Count
    ‘执行语句
Next

```

```

SET Pnt =Nothing

```

```
set TempCir =Nothing
set FeaFac =Nothing
```

3、 相交

判断两个图元是否有交点以及交点坐标信息。

(1) 判断是否相交

```
IF Lyr.IntersectionTest( ftr1, ftr2, miIntersectFeature ) THEN
    ‘交点
END IF
```

(2) 获取相交点坐标信息

‘交点

```
Dim Ftr AS MapXlib.Feature
```

```
SET Ftr=MainMap.FeatureFactory. IntersectFeatures(Ftr1,Ftr2)
```

‘交点坐标信息

```
For J=1 to Ftr.parts.item(1).count
    X1= Ftr.parts.item(1).Item(J).X
    Y1= Ftr.parts.item(1).Item(J).Y
Next
```

4、 测距

使用 Map 对象的 Distance 方法。如何测量任意多边形的周长？

使用累加的方法，还要使用图元节点集合。

DistanceValue=0

‘第一个点

```
Pnt.Set Ftr.Parts.Item(1).Item(1).X, Ftr.Parts.Item(1).Item(1).Y
```

```
For j=2 TO Ftr.Parts.Item(1).Count
```

‘累加

```
X1= Ftr.Parts.Item(1).Item(j-1).X
```

```
Y1= Ftr.Parts.Item(1).Item(j-1).Y
```

```
X2= Ftr.Parts.Item(1).Item(j).X
```

```
Y2= Ftr.Parts.Item(1).Item(j).Y
```

```
DistanceValue = DistanceValue +MainMap.Distance(X1, Y1, X2, Y2)
```

```
Next
```

‘多边形周长

```
Msgbox DistanceValue+” ” +MainMap.MapUnit
```

四、对象编辑

(1)、对属性的编辑

主要使用 Fields 对象。示例：

```
Dim Flds AS MapXLIB.Fields
```

```

‘修改当前图层的每一个字段
For J=1 to Flds.Count
Lyr.KeyField= Flds.Item(j).Name ‘使当前图层指向 J 字段
‘更新当前图元的 J 字段值
Ftr.KeyValue=NewValueStr(J)
Ftr.Update True ‘并未写入硬盘
Next
Lyr.Refresh ‘保存修改到硬盘

```

(2)、移动地图

首先创建一个移动工具句柄

```
MainMap.CreateCustomTool MoveFeature, miToolTypeLine, miPanCursor
```

在 Map 对象的 ToolUsed 事件的 ToolNum 参数为当前所激活的工具

捕捉 MoveFeature 工具句柄

```
‘传过来的参数: X1, Y1, X2, Y2
```

```
Select case ToolNum
```

```
.....
```

```
Case MoveFeature
```

```
Dim Lyr AS MapXlib.Layer
```

```
Dim Ftr AS MapXlib.Feature
```

```
Dim Ftrs AS MapXlib.Features
```

```
Dim Xe,Ye AS Double ‘坐标偏移量
```

```
Xe=X2-X1
```

```
Ye=Y2-Y1
```

```
Set Lyr=Mainmap.Layers.Item(LayerName)
```

```
Set Ftrs=Lyr.Selection.Clone ‘将当前图层中选定的集合复制到 Ftrs 变量中
```

```
MainMap.AutoRedraw=False
```

```
Lyr.Editable=True
```

```
For J=1 to Ftrs.Count
```

```
Set Ftr=Ftrs.Item(J)
```

```
Ftr.Offset Xe,Ye
```

```
Ftr.Update True
```

```
Next
```

```
Lyr.Refresh
```

```
Lyr.Editable=False
```

```
MainMap.AutoRedraw=True
```

```
SET lyr=Nothing
```

```
SET Ftr=Nothing
```

```
End Select
```

(3)、样式更新

```
Dim NewStyle AS MapXLib.Style
```

‘初始赋值

```
Set Lyr=MainMap.Layers.Item(LayerName)
```

```
Set Ftrs=Lyr.AllFeatures
```

```
Set NewStyle=Ftrs.Item(1).Style
```

‘设置样式

```
With NewStyle
```

```
    .SymbolType = miSymbolTypeBitmap
```

```
    .SymbolBitmapSize = 24
```

```
    .SymbolBitmapTransparent = False
```

```
    .SymbolBitmapName = "YIEL2-32.BMP"
```

```
End With
```

‘更新

```
MainMap.AutoRedraw=False
```

```
Lyr.Editable=True
```

```
SET Ftr.Style=NewStyle
```

```
Ftr.Update True
```

```
Lyr.Refresh
```

```
Lyr.Editable=False
```

```
MainMap.AutoRedraw=True
```

五、输出

1、属性的输出 输出到 EXCEL 表:

```
For I=1 to Flds.Count
```

```
Lyr.KeyFields=Flds.Item(i).Name
```

```
Excel(1, I).Cell=Ftr.KeyValue
```

```
Next
```

2、复制、粘贴

```
Global CopyFtrs AS MapXLib.Features
```

```
Set lyr=mainmap.Layers.item(LayerName)
```

```
Set Ftrs=Lyr.Selection.Clone ‘复制选中集合
```

‘复制

```
For I=1 to Ftrs.Count
```

```
CopyFtrs.add Ftrs.Item(I)
```

```
Next
```

‘粘贴 (图形)

```
Set lyr_1=mainmap.Layers.item(LayerName_1)
```

```
Mainmap.AutoRedraw=False
```

```
Lyr_1.Editable=True
For J=1 to CopyFtrs.Count
    Lyr_1.AddFeature CopyFtrs.Item(J)
Next
Lyr_1.Refresh
Mainmap.AutoRedraw=True
Lyr_1.Editable=False
```

3、地图的打印

```
Dim iScaleMode As Integer

iScaleMode = MainMap.Container.ScaleMode
MainMap.Container.ScaleMode = 6

On Error GoTo PrinterError

Printer.Print " "
Printer.CurrentX = 0
Printer.CurrentY = 0
MainMap.PrintMap Printer.hDC, 0, 0, MainMap.Width * 100, MainMap.Height * 10
0
Printer.NewPage
Printer.EndDoc
MainMap.Container.ScaleMode = iScaleMode
Exit Sub
```

PrinterError:

```
If Err.Number = 482 Then
    On Error Resume Next
    CommonDialog1.Flags = &H40
    CommonDialog1.ShowPrinter
Else
    MsgBox "    打印机存在错误，请更正后重试。错误号：" + (Str(Err.Number)),
, "失败"
End If
```

4、另存为图片文件

MainMap.ExportMap(App.Path+" Images", miFormatJPEG) ‘输出当前地图窗口
参数设置：MainMap.ExportSelection=True ‘将选中部分以不同于其他未选中地图部分
形式输出

六、专题图

6 种专题图：除独立值专题图绑定的字段类型可以是字符的以外，都必须是数字类型。与其他数据源绑定时，使用 ODBC

调用 ThemeDlg 对话框可以让用户自己定义专题图。示例：

```
For Each ftr In lyr.Selection
  ' The children of the layer are the individual
  ' features

  Set ftrNode = QueryTree.Nodes.Add(lyrNode, twwChild, lyr.Name _
    & ftr.Name & Str$(ftr.FeatureID), ftr.Name)

  For Each fld In ds.Fields

    ' Each feature has data attached to it; add this data as a child of the feature

    lyr.KeyField = fld.Name

    QueryTree.Nodes.Add ftrNode, twwChild, , lyr.KeyField _
      & ": " & ftr.KeyValue

  Next

Next
```

七、在 MapX 下紧缩表

在 Professional 里面，紧缩表用 Pack Table 语句完成。而在 MapX 中则需要使用临时图层，并用复制技术来完成。示例：

‘紧缩当前 Map 对象中的所有图层

```
Dim LayerInfo As New MapXLib.LayerInfo
Dim Lyr As MapXLib.Layer
Dim LyrTemp As MapXLib.Layer
Dim Flds As MapXLib.Fields
Dim Ds As MapXLib.Dataset
```

```
Dim I As Integer
Dim LayerName, FilePath As String
```

```
On Error Resume Next
```

```
For I = MainMap.Layers.Count To 1 Step -1
  &acute;复制源表数据到临时表
  Set Lyr = MainMap.Layers.Item(I)
  Set Ds = Lyr.Datasets.Item(1)
```

```

Set Flds = Ds.Fields

LayerName = Lyr.Name

LayerInfo.Type = miLayerInfoTypeTemp
LayerInfo.AddParameter "FileSpec", LayerName
LayerInfo.AddParameter "NAME", LayerName
LayerInfo.AddParameter "Features", Lyr.AllFeatures '复制所有有效图元
LayerInfo.AddParameter "Fields", Flds '复制字段列表

LayerInfo.AddParameter "AutoCreateDataset", 1
LayerInfo.AddParameter "datasetname", LayerName
Set LyrTemp = MapTemp.Layers.Add(LayerInfo, 1) '复制到另外 Map 对象

&acute;删除源表
Set Lyr = Nothing
FilePath = MainMap.Layers.Item(I).Filespec
LayerName = Mid(FilePath, InStr(1, FilePath, "Maps") + 6, Len(FilePath)
- InStr(1, FilePath, "Maps"))
FilePath = Mid(FilePath, 1, InStr(1, FilePath, "Maps") + 5)
LayerName = Mid(LayerName, 1, Len(LayerName) - 4)

MainMap.Layers.Remove (I)
MainMap.Refresh

Kill FilePath + LayerName + ".TAB"

&acute;复制临时表数据到源表
Set LyrTemp = MapTemp.Layers.Item(LayerName)

LayerInfo.Type = miLayerInfoTypeNewTable
LayerInfo.AddParameter "FileSpec", FilePath + LayerName + ".TAB"
LayerInfo.AddParameter "NAME", LayerName
LayerInfo.AddParameter "Features", LyrTemp.AllFeatures
LayerInfo.AddParameter "Fields", Flds

LayerInfo.AddParameter "AutoCreateDataset", 1
LayerInfo.AddParameter "datasetname", LayerName
Set Lyr = MainMap.Layers.Add(LayerInfo, 1)

&acute;删除临时表
MapTemp.Layers.Remove (MapTemp.Layers.Count)
MapTemp.Refresh
Next

```

```
Set Lyr = Nothing
Set Ds = Nothing
Set Flds = Nothing
Set LayerInfo = Nothing
```

八、如何和大型数据库关联

系统是混合结构：对地图的访问使用的是文件访问方式，对属性数据的访问使用的是大型数据库形式使用 ID 关联：在 TAB 表和数据库中有 ID 字段，两个字段作为唯一值关联字段。

（一对一的关系）。示例：

```
Lyr.KeyField=" ID"
Set Ftr=Ftrs.Item(J)
```

‘SQL 语句查找对应的属性信息

```
SampleAdo.RecordSource =" Select a.id as id,b.name as name from Table1 a, Table2 b WHERE (a.id=b.id) AND ( a.id LIKE ‘% “ +Ftr.KeyValue+ “%’ )”
```

九、空间数据库

’ 空间数据库中的索引技术用的是 R_Tree 技术，而不是原来一般意义上的 B_Tree 索引技术。

’ 空间数据在 Spatial 中以地理属性信息形式存放。

’ 在 Oracle 的版本中有如下需要注意事项：

a) Oracle 8.1.5 这个版本中，首次引入 Spatial 组件。使用上非常不好：图形的样式非常单一，且为黑白的而非彩色，上载地图数据时，数据丢失非常严重。存取数据时非常慢。

在 Spatial 中自动创建 prinx 字段作为地图索引主关键字段

b) Oracle 8.1.6 版本：图形为彩色的，增加了样式的支持，但不支持同一图层下的多样式（主要指点对象）。创建地图数据时，地图的坐标常发生偏移。属性数据更新时，需要两次刷新才能完整提交；地图数据提交时，其坐标发生偏移：向原点偏移，需要认为地

单独将其移动到其初始创建位置。存取速度上还是比较慢，离实用尚有一定距离。在 Spatial 中自动创建 mi_prinx 字段作为地图索引主关键字段

² Oracle 8.1.7 版本：支持多样式，数据上载丢失非常小（只有文本对象存在丢失的记录）。在地图数据存取不是很大的时候，速度上可以被用户接受。

² 如何将 MapInfo 的 TAB 表内容上传到 Oracle 中？

a) 免费工具：easyloader6.7 (Oracle 8.1.7)，下载地址：www.mapinfo.com.cn

b) 如何上载？注意：上载之前对 Tab 表进行紧缩。

c) 在程序中如何使用空间数据库中的地图数据？

d) 添加地图时，必须指定字段 mi_prinx 的明确值，且该值不能为表中已存在的值。写入地图数据时，应将其全部 NOT NULL 字段值赋给，否则保存失败。

e) 示例：Tab 表+空间数据表 的数据分布形式。

十、GIS 应用的分发

- 1、制作你自己的系统的安装盘：可执行文件、必要系统文件、运行库文件、其他数据文件。
- 2、单独的 MapX 安装盘：MapInfo MapX Runtime 安装程序，实际上是 MapX 控件安装程序（存在于 MapX sdk 包）
- 3、注册：安装完成以后，运行 GeosetManager40.exe 程序获得硬件 ID 号，然后通过 EMAIL 的形式将该 ID 号发送到 MapX 产品供应商申请正式的许可文件（mapx40.lic）。获得后覆盖原 mapx40.lic 文件即可。注意：硬盘格式化后该 ID 号失效。

十一、构造一个 GIS 应用系统

需求：鹰眼功能、拓扑关系、不同图形不同颜色表现、数据绑定、系统性能。

结构分析：做基于 TAB 文件的 GIS 系统

1、鹰眼功能

- (1) 建立两个 Form 对象，将两个 Map 对象分别放在这两个窗口对象中。
- (2) 一个小窗口作为鹰眼窗口，大窗口作为主地图窗口。鹰眼窗口中的 Map 对象的视野应很大，而主地图窗口的视野根据需要设置。
- (3) 两个窗口中加载不同的 GST 文件。需要的是主地图窗口的显示范围应为鹰眼窗口中某个矩形所包含的地图对象范围。
- (4) A、在鹰眼窗口中画一个矩形（Rect 为矩形对象），B、主地图窗口执行 `Set MainMap.Bounds = Rect`
- (5) 需要注意的是：鹰眼窗口与主地图窗口两者间的坐标投影系统应完全一致。

2、拓扑关系

实际上就是图元与图元的空间关系。说历史：原来建立拓扑关系使用的是属性关联。点查询、圆查询、矩形查询、多边形查询等这些是属于简单空间关系的对比。而对拓扑关系的查询多数情况下使用 Parts 对象来解决。

Ftr1 与另一个 Ftr2 的空间关联：先找到 Ftr1 的起止节点，然后以这个节点为中心画一个非常小的圆，在这个圆范围内的某个设备可认为与该 Ftr1 相连。

判断某个图元在指定图层上的相交对象集合：

Set Ftrs=Lyr.SearchWithinFeature (SearchFtr, miSearchTypePartiallyWithin)

3、不同图形不同颜色表现

(1) 更新样式 再结合临时图层就可以很好解决（使用图层刷新）。而且刷新时不会引起整个 Map 对象的刷新（屏幕抖动）

(2) 专题图 存在更新属性值后不能实时刷新专题图的问题。

4、数据绑定

对未绑定的属性集合使用 `MainMap.Datasets.ADD`

地图集合和属性集合为一个数据集合的不可分割的两个部分

5、系统性能

速度是否为用户所接受、系统是否稳定（界面要求）、修改数据数据的时候是否存在数据一致性维护问题、造价是否合理。

（1）速度：使用数据分布可以较好地解决：地图数据：地形图数据以文件形式存放，业务地图数据存放在空间数据库中；属性数据：全部存放在大型数据库中。

（2）系统是否稳定（界面要求）：A、字段全部用英文，B、地图拓扑关系在进入系统以前进行必要的验证，C、文件地图数据中字段数减少到最小，可使访问属性频率大大减少，可有效保证其他用户访问时不会大量出现访问拒绝的现象；D、数据提交时尽可能采用事务机制：事务开始：Lyr.BeginAccess，事务结束：Lyr.EndAccess；E、数据修改提交时尽可能采用批量提交。至于界面要求，应主要满足从地图对象获取相关的要求。

（3）数据一致维护：遵循图元优先的原则。图形对象必须首先存在，其他相关信息在此基础上建立。注意：

² 修改时应先修改地图对象，后提交属性信息

² 删除时应先删除其他信息，最后删除图形信息

（4）造价 这里主要指平台费用。A、现有数据格式，B、功能要求：空间分析是否复杂、地图图层是否分散、系统中地图输出的质量要求、系统中统计分析复杂程度是否较繁琐、直接的平台使用用户数（并发数）。