

地理信息系统

——组件开发篇

刘光 编著



中国电力出版社

www.cpe.com.cn

735

p283.7

471

地理信息系统

——组件开发篇

刘光 编著

本书附盘可从本馆主页 <http://lib.szu.edu.cn/>
上由“馆藏检索”该书详细信息后下载，
也可到视听部复制



中国电力出版社

内 容 提 要

地理信息系统 (GIS) 组件的代表作应首推 MapX 和 MapObjects。其中 MapX 由著名的桌面 GIS 厂商——美国的 MapInfo 公司推出; MapObjects 由全球最大的 GIS 厂商——美国环境研究所 (ESRI) 推出。

本书主要分两大部分, 分别介绍了如何利用 MapX 和 MapObjects 在 Visual Basic、Visual C++ 与 C++ Builder 中进行 GIS 的二次开发。在介绍每个组件时, 都从基本概念开始, 然后由浅入深地介绍如何实现地图显示、符号输出、专题制图和空间查询与分析等内容。本书提供了大量的实例供读者参考。

本书适用于在 GIS、CAD 和 MIS 等领域从事软件开发的广大技术人员以及大专院校的师生。

中国电力出版社出版、发行

(北京三里河路 6 号 100044 <http://www.infopower.com.cn>)

汇鑫印务有限公司印刷

各地新华书店经售

*

ISBN 7-900109-39-0/TP·40

2003 年 1 月第一版 2003 年 1 月北京第一次印刷

787 毫米×1092 毫米 16 开本 22.5 印张 538 千字

定价 39.00 元

版 权 所 有 翻 印 必 究

(本书如有印装质量问题, 我社发行部负责退换)

前 言

在 Windows 系统中, GIS 软件的二次开发方法主要有两大类。

一是利用 GIS 支撑软件自身所带的开发工具, 如 MapInfo 的 MapBasic。这样开发出的程序必须在该 GIS 支撑软件环境里运行。这属于早期的开发方法, 优点是可以利用 GIS 支撑软件的现有功能, 开发周期短; 缺点是系统资源开销大。由于 GIS 支撑软件功能都大而全, 故其程序庞大, 而定制程序又需要在该环境里运行, 每次运行都启动一个庞大的 GIS 支撑软件, 而它的功能又不都是需要的, 其对系统资源的浪费不言而喻; 其次由于开发 GIS 支撑软件的公司一般都熟悉 GIS, 而开发工具非其所长, 故其附带的开发工具一般说来开发效率不高, 功能也不强, 难以开发出界面友好而又功能强大的定制程序; 另外由于与 GIS 支撑软件紧密结合, 受其限制也大, 在汉化方面受约束多, 更主要的是移植性差, 对软件的未来发展不利, 且这种方法至今没有什么新的突破, 只是随着所使用的 GIS 支撑软件的发展而发展。

二是利用优秀的专业开发工具, 如 Visual Basic、Visual C++、C++ Builder、Delphi 和 Power Builder 等, 而后台对地图的处理依然采用整个 GIS 支撑软件。与第一种相比, 它的系统资源开销依然很大, 但由于前台采用专业开发工具, 在开发效率、开发能力、汉化、移植性等方面都大大优于第一种。而且这种方法能随着整个软件开发行业的发展而不断发展。此种方法关键是前台定制程序与后台 GIS 支撑软件之间的结合方式, 它不仅影响到系统的资源占用也影响到编程效率和移植性等。最初, 前台与后台之间采用动态数据交换 (DDE) 方式进行通信, 效率不高, 且开发较麻烦, 故而与第一种方法并存了一段时间。后来, 很多专业开发工具和 GIS 支撑软件支持的对象链接与嵌入 (OLE) 方式出现了, 它大大提高了前台客户程序和后台 GIS 支撑软件两者结合的紧密程度, 提高了编程效率。而后是现今流行的 OLE / COM 编程, 它极大地提高了定制程序和 GIS 支撑软件之间的紧密程度, 可以说此时已不存在以前所谓的前台和后台之分了, 这是面向对象的技术充分运用的结果。说得形象点, 就是将 GIS 支撑软件拆成一个个小对象控件 (动态链接库), 需要时调入系统使用, 不用时由操作系统自动清理出内存。实际上这与定制程序自身所带的动态链接库已没有什么区别。这样显然大大降低了系统资源的占用率, 同时又保留了专业开发工具所带来的编程高效性, 而且定制程序与 GIS 支撑软件既能紧密结合, 又能相互独立。更重要的是这种真正面向对象的技术为将来软件的维护、升级换代或跨平台移植奠定了坚实的技术基础。

本书分为两大部分

第一部分介绍了利用 MapX 组件进行 GIS 的二次开发, 从第 1 章~第 7 章。MapX 由著名的桌面 GIS 厂商美国的 MapInfo 公司推出。MapInfo 吸取了传统 GIS 系统的精华, 并借助于计算机技术的进展, 及时地将 GIS 的概念从中大型计算机的专用工作站引入到普通 PC 机器上, 开创了一种崭新的 GIS 模式, 即桌面 GIS。MapInfo 产品自 20 世纪 90 年代初进入中国后, 在各行各业得到了普遍的关注。在统计、信息中心、测绘、邮电、水利、环保、

油田、林业、军事和工商等部门得到了广泛的应用。

第 1 章主要介绍 MapX 的特征、MapX 支持的外部数据及其空间数据结构,还介绍了 MapX 的一些基础知识。例如,地图对象、图层集以及地图化概念等,最后通过实例演示了如何在 Visual Basic、Visual C++与 C++ Builder 中开始使用 MapX 组件。

第 2 章主要介绍图层集合及其属性和方法、图层对象、图层顺序、检查图层、图层标注、笔记以及活动图层。

第 3 章主要介绍数据绑定、在地图中加入数据、数据集对象、数据集集合、字段集合对象、绑定图层与数据源的不同类型等内容。

第 4 章主要介绍地图特征、特征集、选择集与编辑特征等内容。

第 5 章主要介绍如何通过 MapX 创建并控制专题地图及其图例。MapX 提供强大的专题地图功能,可以创建单变量专题地图,如范围、独立值、点密度和等级符号等,或多变量专题地图,如饼图和直方图等。

第 6 章主要介绍如何使用查找对象的方法在地图中查找特征,以及如何使用查找特征对象。

第 7 章主要介绍如何使用坐标系统来改变地图的显示,以及改变地图坐标的单位等内容。在各类 GIS 的建立过程中,选择适当的地图投影系统是首先要考虑的问题。没有合适的投影或坐标系的空间数据不是一个好的空间数据,甚至是没有意义的空间数据,因为这种数据不含实际地理意义。

第二部分介绍了利用 MapObjects 组件进行 GIS 的二次开发从第 8 章~第 14 章。MapObjects 由全球最大的 GIS 厂商美国环境研究所(ESRI)推出。美国环境系统研究所是世界上 GIS 的拓荒者,同时也是当今 GIS 技术的领先者。他们在 20 世纪 80 年代就推出了 Arc/Info 软件,之后又陆续推出多种 GIS 软件产品满足不同的需求,如 SDE、ArcView GIS、ArcCAD 等,这些产品已经建立起制图与 GIS 的工业标准。

第 8 章主要介绍 MapObjects 的功能、特点与结构,以及如何在当前最流行使用的开发语言环境 Visual Basic、Visual C++与 C++ Builder 中开始使用 MapObjects。

第 9 章首先介绍利用地图控件的属性与方法来控制地图的显示,例如放大、缩小、漫游与显示全图等;然后介绍如何在地图中加入各种类型的数据,例如 Shape 文件、ArcInfo Coverage、ArcSDE 图层等;最后介绍地图输出与利用 Windows API 扩展 MapObjects 应用程序。

第 10 章主要介绍如何利用符号对象与着色对象表现地理特征的属性数据。

第 11 章主要介绍通过查询与数据集有关的表从数据中获取信息,以及通过空间和逻辑的查询方法从数据中获取信息。同时介绍如何查询、修改和更新记录集中的地理数据集。

第 12 章主要介绍如何利用几何对象实现创建与编辑特征,进行地图数字化,以及如何合并与拆分数据集。

第 13 章主要介绍如何利用 MapObjects 开发地理编码应用程序,进行地址匹配。

第 14 章主要介绍了坐标系与投影问题。

由于作者理论水平和实践经验有限,书中肯定存在一些缺点甚至错误,希望能得到广大读者和专家不吝批评与指正。

作 者

2002 年 11 月

目 录

前 言

第 1 章 MapX 开发概述	1
1.1 MapX 概述	1
1.2 使用 MapX	5
1.3 MapX 基础	12
1.4 数据组织	17
第 2 章 图层与图层集合对象	19
2.1 图层集合	19
2.2 图层对象	29
2.3 缩放图层	34
2.4 图层标注	35
2.5 注记	38
2.6 活动图层	44
2.7 用户绘制图层	49
第 3 章 数据绑定	58
3.1 数据绑定概述	58
3.2 数据集对象与数据集集合	59
3.3 字段集合	62
3.4 绑定图层	64
第 4 章 特征与选择集合	75
4.1 特征集合	75
4.2 选择集合	85
4.3 特征编辑	89
第 5 章 专题图与分析	104
5.1 专题图的概念	104
5.2 设计专题图	104
5.3 专题图类型	108
5.4 控制专题图	119
5.5 自定义专题图的图例	120
第 6 章 查找特征	121
6.1 查找对象	121

6.2 查找特征对象	125
第 7 章 坐标系	134
7.1 坐标系的基本概念	134
7.2 坐标系对象	135
第 8 章 MapObjects 开发概述	143
8.1 MapObjects 概述	143
8.2 MapObjects 的结构	145
8.3 在 Visual Basic 中使用 MapObjects	148
8.4 在 C++ Builder 中使用 MapObjects	150
8.5 在 Visual C++ 中使用 MapObjects	153
第 9 章 地图图层化	157
9.1 地图控件	157
9.2 图层控制	164
9.3 地图图层	179
9.4 动态跟踪图层	183
9.5 输出地图	191
9.6 利用 Windows API 函数	193
第 10 章 显示特征与符号化属性	199
10.1 符号对象	199
10.2 着色对象	208
第 11 章 特征选择	248
11.1 记录集对象	248
11.2 选择特征	254
第 12 章 创建与编辑地理特征	264
12.1 几何对象	264
12.2 创建与编辑特征	282
第 13 章 开发地理编码应用程序	309
13.1 地理编码概述	309
13.2 用于地址匹配的对象	311
13.3 实现地理编码	314
13.4 地名查找	327
第 14 章 坐标系与投影	333
14.1 如何在 MapObjects 中使用坐标系	333
14.2 使用坐标系应注意的问题	335
14.3 地图投影对象的使用	335
14.4 坐标系统转换	340

第 1 章 MapX 开发概述

MapX 是 MapInfo 公司向用户提供的具有强大地图分析功能的 ActiveX 组件产品。由于它是一种基于 Windows 操作系统的标准组件,因而能支持绝大多数标准的可视化开发环境,如 Visual C++、Visual Basic、Delphi 和 PowerBuilder 等。编程人员在开发过程中可以选用自己最熟悉的开发语言,轻松地将地图功能嵌入到应用中,并且可以脱离 MapInfo 的软件平台运行。利用 MapX,能够简单快速地在企业应用中嵌入地图化功能,增强企业应用的空间分析能力,实现企业应用的增值。MapX 采用基于 MapInfo Professional 的相同的地图化技术,可以实现 MapInfo Professional 具有的绝大部分地图编辑和空间分析功能。而且,MapX 提供了各种工具、属性和方法,实现这些功能是非常容易的。同时由于 MapX 与其他 MapInfo 产品基于同样的地图化技术,例如 MapInfo Professional 与 Microsoft Map,因此在 MapX 中能使用在 MapInfo Professional 或 Microsoft Map 中创建的 MapInfo 地图数据(表)。

本章将主要介绍 MapX 的主要特征、MapX 支持的外部数据及其空间数据结构;还介绍 MapX 的一些基础知识,例如地图对象、图层集以及地图化概念等;最后通过实例演示了如何在 Visual Basic、Visual C++ 与 C++ Builder 中使用 MapX 组件。

1.1 MapX 概述

本节将介绍 MapX 的关键特征、MapX 的空间数据结构、MapX 组件的模型结构以及 MapX 所支持的外部数据。

1.1.1 MapX 的关键特征

虽然 MapInfo 是一个面向中小型用户的桌面地理信息系统(它的价格远远低于 ARC/INFO 和 MGE),但是 MapX 不仅仅是一个地图显示器。通过 MapX,可以分析和可视化企业数据、创建与编辑地图特征以及地图化的显示结果

MapX 主要包含以下一些关键特征:

1. 专题地图(Thematic mapping)

专题地图是 MapInfo 产品有别于其他 GIS (Geographical Information System, 地理信息系统) 产品的功能。虽然 GIS 产品一般都提供专题制图功能,但是远不如 MapInfo 的功能强大。通过 MapX 可将数据库表中的特定值赋给地图对象的颜色、图案或符号,从而创建不同的专题地图。通过 MapX 可创建范围值、等级符号、点密度、独立值、直方图和饼图

6 种方式的专题地图。

2. 可深入的地图 (Drill-down mapping)

通过简单的单击方式可浏览与地图对象相连的分成多个等级的数据信息。例如，一 Drilldown 地图包括两个销售大区，每个销售大区包含几个销售小区，每个销售小区又包含几个省，每个省又包含多个县，这时用户可以在地图中单击鼠标便可以层层深入，也可以通过单击鼠标逐级返回。

3. 数据绑定 (Data binding)

地图可以从组件嵌入的容器中获取数据，也可以从 ODBC (Open Data Connectivity, 开放数据库连接性) 或 DAO (Database Access Object, 数据库访问对象) 数据源 (例如 Microsoft Access) 获取数据。MapX 提供了多种绑定数据的方法。例如，可绑定包含 X/Y 字段的数据，也可对包含邮政编码的数据进行地理编码。

4. 注记 (Annotations)

通过 MapX 可控制指定数据的方向、加亮显示，还可加入文本、符号、表格使地图信息更加丰富直观。

5. 图层化 (Layering)

可控制图层只有当地图在一定缩放范围才显示。还可以通过使用或创建无缝地图图层 (seamless map layer)，就如同控制一张表一样控制一组基础表。通过一个无缝图层可以为一整组表改变显示、实施或改变标注或使用图层控制对话框。还可支持一些特殊的应用，比如用于实时跟踪的活动图层和可绘制特殊图形的用户自定义图层 (如 logo 图案)。

6. 栅格图像 (Raster Images)

采用栅格图像作为地图的基础图层，可使其他图层有一个更细致的背景。只要将地图图层覆盖在栅格图像上，就可以将栅格图像作为编辑地图图层的参考，这种基于屏幕图像来进行图形编辑的过程称为“屏幕数字化”。这也是 MapInfo 所特有的功能。

7. 自动标注 (Automatic Labeling)

可控制自动在地图上加入标注，同时也可以控制标注的属性和显示。

8. 选择 (Selections)

可在地图上拖动鼠标绘制一虚拟的圆、矩形或特定的点，然后用它们选择一个或多个对象或记录以供分析。

9. 特征库 (Feature Factory)

通过 FeatureFactory 对象可创建、合并或删除点、线、区域特征，也可以为这些特征创建缓冲区。

10. 工具 (Tools)

MapX 中内建了许多工具。例如，放大、缩小、漫游、标注、选择和圆选等，根据需将这些工具集成到我们的应用程序中，这样用户可以通过简单的单击或拖动鼠标便可以与地图进行交互。同时还可以创建自定义的工具。例如，在创建 Drilldown 地图时，需要创

建一 Drilldown 工具和一 Rollup 工具。

11. 地图编辑 (Map Editing)

通过 MapX 可为用户提供添加、修改和删除地图上的特征功能。

12. 投影与坐标系统 (Projections and Coordinate Systems)

MapX 完全支持坐标系统和地图投影。通过 MapX 可允许用户调整地图的显示, 也可用本地坐标系统处理 X-Y 坐标数据。

13. 连接远程空间数据服务器 (Remote Spatial Server Connectivity)

可以访问存储在 Oracle8i 和 MapInfo SpatialWare 中的远程地图数据。空间数据服务器 (如 SpatialWare 和 Oracle8i 等) 都提供了先进的查询处理能力, 提高了空间数据组织的性能。将空间数据存储到关系型数据库中, 可以增加应用程序的灵活性, 同时也要求在地图编辑和大数据集方面做更多的工作。

1.1.2 MapX 的空间数据结构

空间数据结构是对 GIS 空间数据元素之间结构关系的描述, 它是 GIS 的基石。空间数据按其存储格式分为两大类, 即矢量数据和栅格数据。MapX (甚至 MapInfo) 中虽然同时支持矢量数据和栅格数据, 但是对栅格数据的支持仅仅只达到显示程度, 并不支持进一步的处理与分析, 也就是说只作为矢量数据显示的背景而已。

从横向分析, MapX 采取的空间数据结构是基于空间实体和空间索引相结合的一种结构。空间实体是地理图形的抽象模型, 主要包括点、线、面三种类型。任何点、线、面实体都可以用直角坐标点 x 、 y 来表示。点可以表示成一组坐标 (x, y) , 对于线和面, 则均被表示成多组坐标 $(x_1, y_1; x_2, y_2; \dots; x_n, y_n)$ 。空间索引是查询空间实体的一种机制, 通过空间索引, 就能够以尽量快的速度查询到给定坐标范围内的空间实体及其所对应的数据。

从纵向分析, MapX 的空间数据结构是一种分层存放的结构, 如图 1.1 所示。用户可以通过图形分层技术, 根据自己的需求或一定的标准对各种空间实体进行分层组合, 将一张地图分成不同图层。采用这种分层存放的结构, 可以提高图形的搜索速度, 便于各种不同数据的灵活调用、更新和管理。

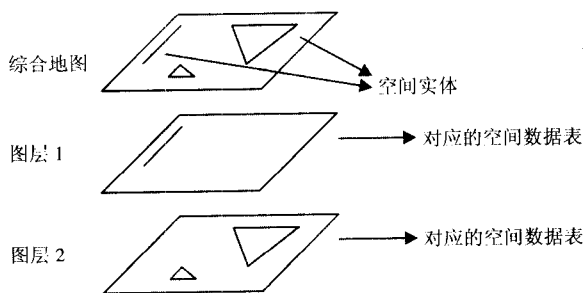


图 1.1 MapX 的空间数据结构

1.1.3 MapX 组件的模型结构

MapX 组件的基本组成单元是单个 Object（对象）和 Collection（集合）。其中集合包括对象，是多个对象的组合。每种对象和集合负责处理地图某一方面的功能。

MapX 组件的模型结构如图 1.2 所示。位于顶层的是 Map 对象本身，其他均由 Map 对象继承。Layers、DataSets、Annotations 是 Map 对象下面的三个重要的分支。其中 Layer 主要用于操作地图的图层，DataSet 用于访问空间数据表，Annotation 用于在地图上增加文本或者符号。

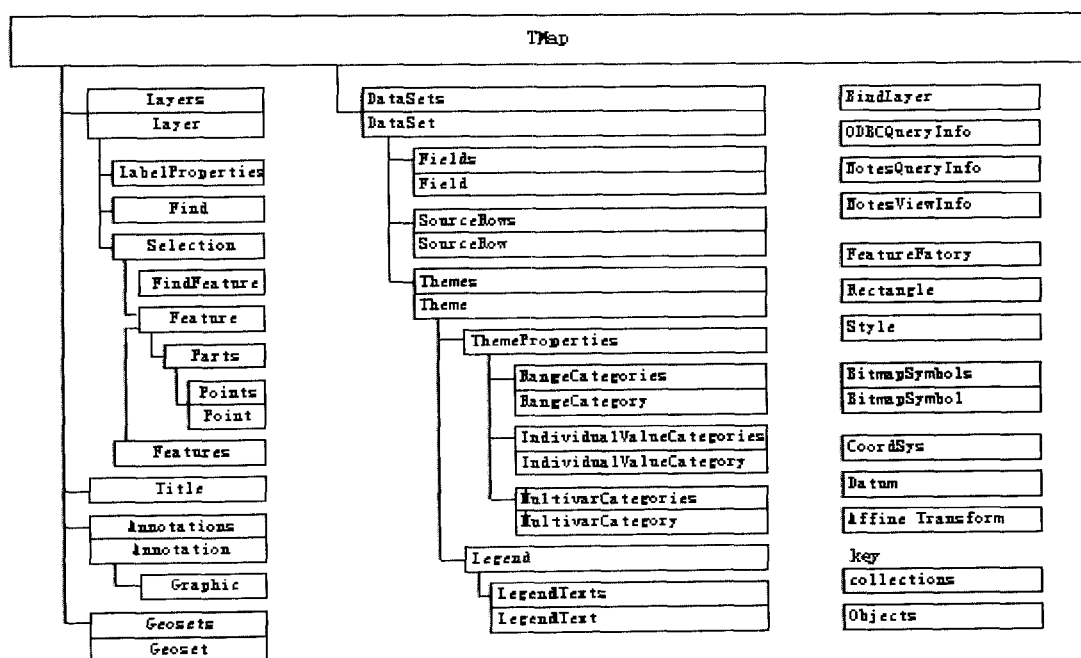


图 1.2 MapX 组件的模型结构

1.1.4 MapX 支持的外部数据

在 MapX 中可以引用多种类型的外部数据。

(1) 地图数据：如果用户已经购买了或是利用 MapInfo 创建了 MapInfo 地图，可以直接将它们在应用中打开。

(2) 远程空间数据库：利用 MapX 可以访问保存在 Oracle8i 及 MapInfo SpatialWare 中的地图数据。其中，对 Oracle8i 的支持是 MapX 4.0 的新特性。通过 Oracle8 Call Interface (OCI)，MapX 可以将存储在 Oracle8i 数据库服务器中的 MapInfo 空间数据和属性数据同时下载到本地。

(3) 其他远程数据：MapX 支持多种对外远程数据的访问方式，如 ADO（Active Data Objects，活动数据对象）、DAO、及 RDO（Remote Data Objects，远程数据对象）等，更可以通过 ODBC 使用更广泛范围内的数据。

1.2 使用 MapX

本节将通过三个实例分别演示如何在 Visual Basic、C++ Builder 和 Visual C++ 中使用 MapX 组件。读者将会发现，虽然同样是使用 MapX 组件，但是由于开发工具不同，使用的具体过程也会有很大的区别。

1.2.1 在 Visual Basic 中使用 MapX

在 Visual Basic 中使用 MapX 最简单，根本不需要编写一行代码就可以在应用程序中显示一张地图。下面将通过实例 1.1 演示如何在 Visual Basic 中加入 MapX 组件中的 Map 控件并显示的一张地图。

【实例 1.1】 启动 Visual Basic，从对话框中选择 New Standard EXE，创建一新的应用程序。在工具箱上单击右键，然后选择弹出式菜单中的 Components 命令，打开如图 1.3 所示的控件对话框。当然也可以通过选择 Project 菜单中的 Components 命令打开该对话框，还可以直接按 Ctrl+T 快捷键来实现。

在 Controls 对话框的控件列表中找到 MapInfo MapX V4，并在右边的复选框中打勾，表示选择该控件。然后选择确定按钮关闭对话框。这时在 Visual Basic 的工具箱上出现一新的控件，这个新的控件就是 Map 控件。如图 1.4 所示。

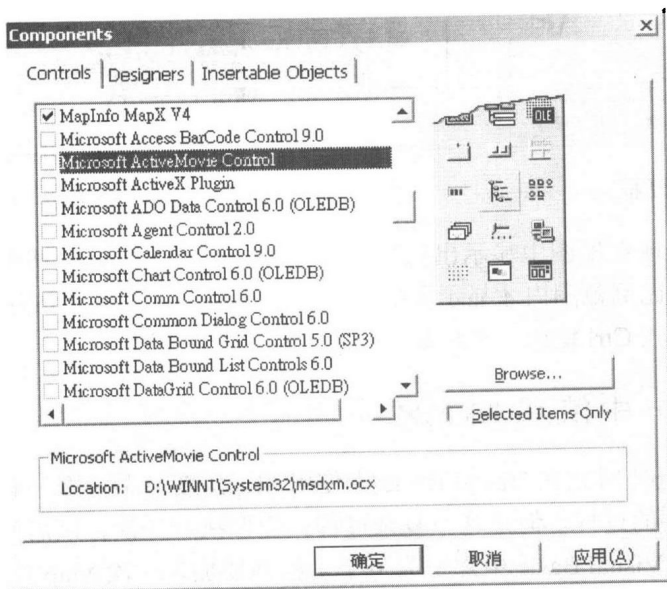


图 1.3 加入 MapX 控件

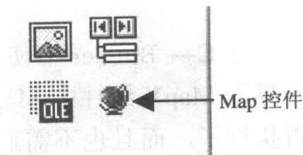


图 1.4 Map 控件

地图控件用于显示地图。因此要显示地图，首先需要在窗体中加入一地图控件。可以通过在工具箱上双击地图控件，在窗体中加入地图控件，并根据需要再适当调整地图控件的大小和位置。也可以先选择该地图控件，然后在窗体中按住鼠标左键并拖动鼠标，指定地图控件的位置和大小。

当在窗体中加入地图控件时,默认显示一美国地图,即地图控件 Map 的 GeoSet 默认属性为 United States。此外,也可以通过该属性在地图控件中显示其他地图。

不过有个更好的工具用于在程序设计期间加入地图数据,那就是属性对话框。在地图控件上单击鼠标右键,然后选择“Properties”命令,打开如图 1.5 所示的地图属性页(Property Pages)对话框。

在对话框中 General 选项卡的底部,找到 Current 选项。该选项用来设置地图中的当前工具,例如选择工具、放大工具、缩小工具以及漫游工具等。通过下拉框将当前工具设置为 1003-ZoomIn,即放大工具。

选择“确定”按钮关闭对话框。现在就可以编译并运行程序。可以观察到当光标在地图范围内时,光标显示成一个带加号的放大镜。用鼠标单击地图,便可以看到 MapX 在单击位置处放大地图。当然也可以通过画一个矩形来放大地图。图 1.6 所示为放大显示后的地图。

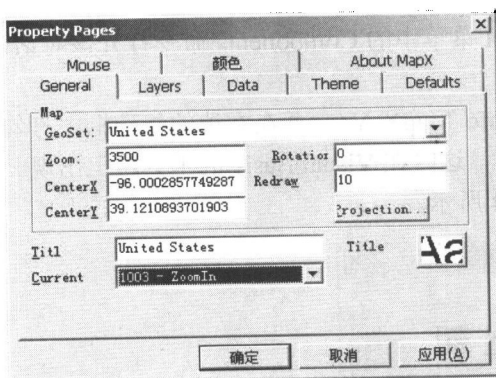


图 1.5 地图属性页对话框

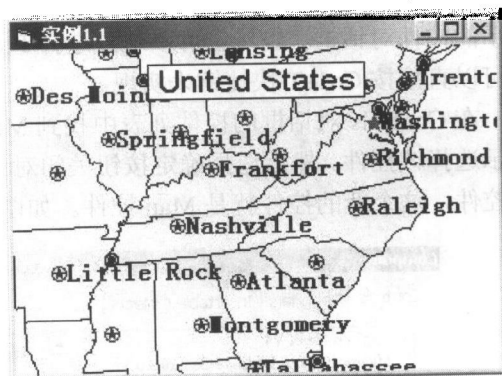


图 1.6 放大显示后的地图

随着地图的放大,就会发现在地图中显示出更多的地理特征。这是因为一些图层设置成了缩放图层(在一定的缩放比例范围内才显示图层)。要缩小地图,只需要按住 Ctrl 键不放,并在地图中单击。当按住 Ctrl 键时,光标显示成一个带减号的放大镜。

1.2.2 在 C++ Builder 中使用 MapX

在 C++ Builder 中使用 MapX 时比在 Visual Basic 中使用 MapX 要复杂得多,它多了一个导入 MapX 组件中 Map 控件的过程。但是这个复杂过程只需要执行一次,以后便不需要再执行了,而且也不需要像在 Visual Basic 中那样为每个工程都要加入一次 Map 控件。

在 C++ Builder 集成开发环境中加入 Map 控件,操作步骤如下:

- (1) 选择 Component 菜单中的 Import ActiveX Control 命令。
- (2) 如果原来没有注册过 MapX,必须选择 Add 按钮,打开 Register OLE Control 对话框来注册该控件。找到 MAPX40.OCX 并将它加入。
- (3) 拖动滚动条并从列表框中选择 MapInfo MapX V4,如图 1.7 所示。

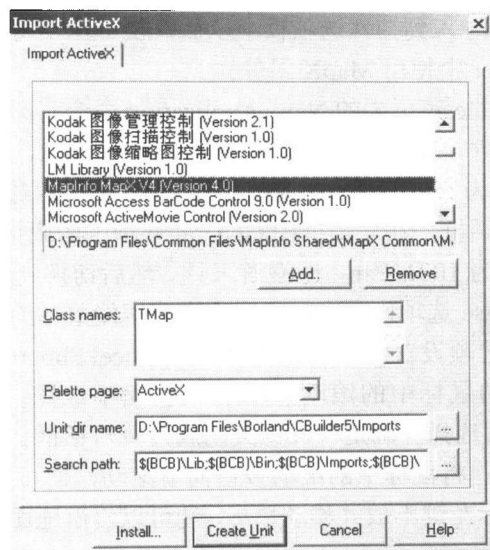


图 1.7 Import ActiveX 对话框

(4) 选择 Install 命令，将 MapX 安装到控件选项板上。选择 Install 命令将产生 MapX_TLB 与 MapX_OCX 两个单元文件，然后弹出如图 1.8 所示的 Install 对话框。在该对话框中用户可以指定将 MapX 组件安装在哪个程序包中。在 C++ Builder 中用程序包组织控件。Dclusr50.bpk 是 C++ Builder 专门为用户安装控件准备的，用户也可以将控件安装在其他程序包中，甚至可以创建一新包来包含该控件。

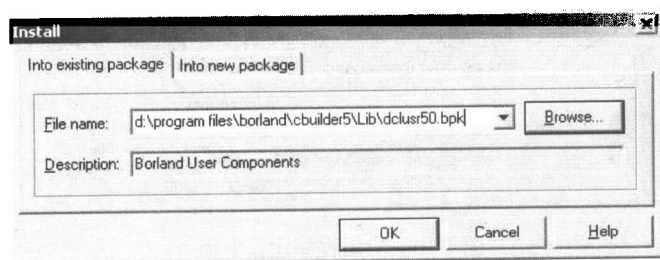


图 1.8 Install 对话框

(5) 接受默认设置，选择 OK 按钮。C++ Builder 就会重新编译 dclusr50.bpl 包文件。编译完毕之后，MapX 就被嵌入到用户组件程序包中了，并且 C++ Builder 会弹出一消息框显示 TMap 控件已被注册。选择 OK 按钮完成安装。并选择 File 菜单中的 Save All 命令保存所做的改变。这时可以切换到控件选项板的 ActiveX，就会看到新增加的 Map 控件。如图 1.9 所示。

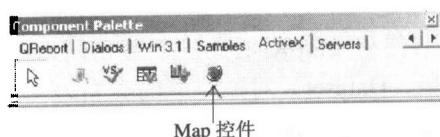


图 1.9 Map 控件在控件选项板中的位置

将 MapX 的地图控件导入到控件选项板上以后, 便可以使用该控件了。下面通过实例 1.2 演示如何在 C++ Builder 中使用 MapX 组件。

【实例 1.2】 选择 File 菜单中的 New Application 命令, 新建一工程。在窗体 Form1 上加入 MapX 的 Tmap 控件, 将它的 Align 属性设置为 alClient, 以便地图控件充满整个窗口, 并将 Geoset 属性设置为 Europe 1。在地图控件上单击鼠标右键, 然后选择 “Properties” 命令, 打开地图属性页对话框。在对话框中 General 选项卡的底部, 找到 Current 选项。通过下拉框将当前工具设置为 1001-Pan, 即漫游工具。然后切换到 Mouse 选项卡中, 在该选项卡的底部有一 Intellimouse 选项, 该选项是用来设置智能鼠标的。如果你的鼠标左右键之间有滚动按钮的话, 将该选项设置为 “2-Partial Mousewheel Support” 或 “3-Full Mousewheel Support”, 便能通过滚动鼠标中的滚动按钮放大和缩小地图。在本实例中选择 3-Full Mousewheel Support。然后选择 “确定” 按钮关闭属性页对话框。

编译并运行应用程序。将鼠标中的滚动按钮向前滚动, 便可放大地图, 向后滚动便可缩小地图。还可以按住鼠标左键不放并移动鼠标, 便可以漫游地图。应用程序界面如图 1.10 所示。



图 1.10 应用程序运行界面

从实例 1.1 和实例 1.2 可以看到我们并没有编写一行代码, 就实现了在应用程序中显示一张地图; 并可以缩放地图, 而且根据缩放比例控制各个图层的显示, 这些充分显示了 MapX 组件的方便、快捷和强大的支持能力。

1.2.3 在 Visual C++ 中使用 MapX

在 Visual C++ 中使用 MapX 的方法不同于在 Visual Basic 和 C++ Builder 中使用。在 Visual C++ 中, 每个 MapX 对象是一个 C++ 类, 这些类在 MapX.h 文件中声明, 在 MapX.cpp 文件中实现。这些 C++ 类的名称与 MapX 对象的名称相同, 在对象名称前面加上 CMapX 就构成了类名。例如, 数据集 (Dataset) 对象对应的类名为 CMapXDataset。MapX.cpp 和 MapX.h 文件在 MapX 安装目录的 Sample\C++ 子路径中可找到, 也可以从该书光盘 Sample01-03 路径中找到。

MapX 对象的属性是通过 C++ 类的成员函数来实现的。例如，像数据集对象的可读/写属性 Name 是由 CMapXDataset 类中的两个成员函数实现——一个用于设置属性的值，另一个用于得到属性的值。这些成员函数的名称是在属性名称前加“Get”或“Set”而成。

例如：

```
CString GetName();  
void SetName(LPCTSTR);
```

只读属性只有“Get”成员函数，而没有“Set”成员函数。

MapX 对象的方法通过与该方法同名的成员函数实现。许多方法的参数是“const VARIANT &”，这意味着成员函数要求传递一个类型为 VARIANT 的变量。

同时许多方法的一些参数，对于像 Visual Basic 或其他脚本语言来说是可选择。但是在 Visual C++ 中，当调用这些方法时，必须设置所有的参数。所有的可选参数都是 VARIANT 类型。可以利用如下的方法调用带可选参数的 MapX 方法：

```
VARIANT vtOptional;  
vtOptional.vt = VT_ERROR;  
vtOptional.scode = DISP_E_PARAMNOTFOUND;
```

为了在方法中便于调用标准的 C++ 类型而不是 VARIANT 的变量，在 MapX.h 和 MapX.cpp 中重载了大多数 MapX 方法。但是一些方法没有重载，如果需要使用，用户可以自己重载这些方法。

在 MapX.h 中的 C++ 类都是从 MFC (Microsoft Foundation Class, Microsoft 基础类库) 的 COleDispatchDriver 类派生而来。所有的属性和方法最终都调用 IDispatch->Invoke() 来告诉 MapX 的 OCX 做什么。COleDispatchDriver 通常能正确调用从属性或方法返回的 IDispatch 指针的 Release() 函数，但还是有一些需要特别注意的地方。当一个 MapX 对象通过它的 IDispatch 接口被传递给一事件句柄时，必须确保不要释放该指针，因为传递给事件的对象没有调用 AddRef 函数来增加引用计数。

下面将通过实例 1.3 来演示如何创建一个 MapX 应用程序，介绍在 Visual C++ 中创建并利用 MapX 控件。

【实例 1.3】 首先选择 File 菜单中的 New 命令，弹出 New 对话框，选择 MFC AppWizard (exe)，并在 Project Name 中输入 MapxSample 作为工程名称。然后单击 OK 按钮，进入应用程序向导的第一步。通过应用程序向导创建一多文档界面应用程序。

在创建 MapX 控件之前，必须将 MapX.h 和 MapX.cpp 文件加入到工程中。对于 Visual C++ 5.0 及以上版本，从 Project 菜单中，选择 Add To Project > Files 命令，打开 Insert Files into Project 对话框，选择 MapX.cpp 和 MapX.h 文件加入到工程中。最后将这两个文件拷贝到工程所在的路径下面。



不要选择 Project 菜单中的 Add To Project > Components And Controls 命令。如果选择该命令加入 MapX 控件，将创建一新的.cpp 文件，但是该文本没有包括所有的 MapX 对象。

在工程中加入 MapX.h 和 MapX.cpp 文件后，在包含该控件的 CMapxSampleView 中加

入 MapX 对象的头文件。如下所示：

```
#include "MapX.h"
class CMapxSampleView : public CView
{
protected:
    CMapX m_ctrlMapX;
}
```

然后需要为 MapX 创建一资源 ID 来代表该控件。步骤如下：

- (1) 选择 View 菜单的 Resource Symbols 命令，打开 Resource Symbols 对话框。
- (2) 在 Resource Symbols 对话框中单击 New 按钮，弹出 New Symbol 对话框，如图 1.11 所示。

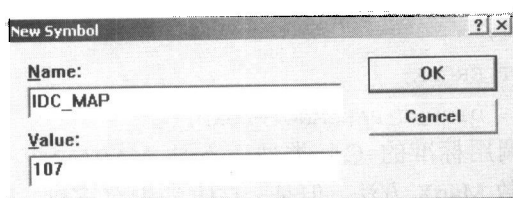


图 1.11 New Symbol 对话框

- (3) 在 New Symbol 对话框中输入 IDC_MAP 作为名称。

通过类向导创建 CMapxSampleView 类的 WM_CREATE 和 WM_SIZE 消息映射函数，然后切换到 MapxSampleView.cpp 中，在 CMapxSampleView::OnCreate 中加入如下所示的代码，用于创建地图：

```
//-----
int CMapxSampleView::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CView::OnCreate(lpCreateStruct) == -1)
        return -1;

    // 使用默认的大小创建地图
    // 在 resize 消息映射函数中调整地图的大小，使它充满整个客户区
    if (!m_ctrlMapX.Create(NULL, WS_VISIBLE,
        CRect(0,0,100,100),this, IDC_MAP))
        return -1;

    return 0;
}
//-----
```

在 CMapxSampleView::OnSize 中加入如下所示的代码，用于调整地图的大小：