

新版软件步步高 提 高 本

中文

Visual FoxPro 6.0

高级教程

捷新工作室 编著



- 新增功能详解
- 客户/服务器应用程序设计
- 如何访问API
- 如何使用外部库
- 群体开发、企业级开发与国际化开发
- OLE、Internet/ Web 应用
- 程序编译、调试、优化与发布

国防工业出版社

新版软件步步高(提高本)

中文 Visual FoxPro 6.0 高级教程

捷新工作室 编著

国防工业出版社

·北京·

图书在版编目(CIP)数据

中文 Visual FoxPro 6.0 高级教程/捷新工作室编著. —
北京:国防工业出版社,1999.4
(新版软件步步高:提高本)
ISBN 7-118-02069-9

I.中… II.捷… III.关系数据库-数据库管理系统,
FoxPro-教材 IV.TP311.13

中国版本图书馆 CIP 数据核字(1999)第 02765 号

国防工业出版社出版发行

(北京市海淀区紫竹院南路 23 号)

(邮政编码 100044)

北京怀柔新华印刷厂印刷

新华书店经售

*

开本 787 × 1092 1/16 印张 25 572 千字

1999 年 4 月第 1 版 1999 年 4 月第 1 次印刷

印数:1-4000 定价:34.00 元

(本书如有印装错误,我社负责调换)

总 序

在信息时代,知识成为推动社会生产力发展的一个最重要的因素,知识经济的轮廓在一些发达国家已经出现。以知识为基础的知识经济,其重要特点是信息产业的迅猛发展和产业的信息化,其内核是创新。我国是发展中国家,应该更加重视运用世界最新技术成果,有重点有选择地引进先进技术,增强自主创新能力,逐步实现技术发展的跨越。

在充分估量未来科学技术,特别是以计算机技术为先导的高技术发展对综合国力、社会经济结构和人民生活的巨大影响的基础上,为实现科教兴国战略多于实事,国防工业出版社组织了由数十位在计算机应用开发第一线工作的年富力强的博士、硕士组成的捷新工作室,编写出版《新版软件步步高(基础本)》和《新版软件步步高(提高本)》丛书。前者重在普及,后者追求提高,两者结合起来,力图满足多层次读者的需要。

《新版软件步步高(基础本)》的出版目的是普及新版软件的基本知识、基本操作技术,主要面向初学者,包括初次涉足该领域的机关、公司、企事业单位技术人员、大专院校师生及各类培训班学员,也可作为广大再就业职工理想的培训和学习教材。另外,对新技术感兴趣的读者也可将本丛书作为了解世界最新技术的窗口。

《新版软件步步高(提高本)》的出版目的是为有一定基础的读者找到提高专业技术水平和最新软件开发、操作技术的最佳途径,主要面向中高级读者,包括对该软件有一定基础知识,又希望提高自己专业技术水平的机关、公司、企事业单位技术人员、大专院校师生及各类高级培训班学员。

丛书的共同特点是突出一个“新”字,强调一个“精”字,力争一个“快”字。“新”是指软件的版本新;“精”是指精选的国内外流行最广、叫得最响的新版软件;“快”是指在保证质量的前提下,实现周期短,面市快。

丛书的内容覆盖最新高级语言开发环境(Visual J++ 6.0 Visual Basic 6.0, Visual Fox-Pro 6.0, Delphi 4.0),微机操作系统(中文 Windows 98, 中文 Windows NT 5.0),图形图像处理软件(Photoshop 5.0, 中文 CorelDRAW 8.0),Internet 浏览器(Internet Explorer 5.0),Web 页面设计环境(FrontPage 98),Internet 开发环境(Visual InterDev 6.0),大型数据库客户端开发工具(PowerBuilder 6.0)等。它们都是1998年以来推出的最新版软件。同时,我们还将把握新技术的脉搏,适时充实新的内容。

我们相信,丛书的出版必将为广大读者开辟跟踪、掌握、运用、进而创造新技术的最佳途径。

由于时间仓促,书中疏漏之处,敬请广大读者指正。

前 言

Visual FoxPro 6.0 是 Microsoft 公司推出的最新一代数据库管理系统,它继承了以往所有旧版本数据库管理系统的功能,并扩展了对应用程序的管理和在 Internet 上发布用户数据的功能,使得用户开发数据库的工具更加完善与快捷,从而成为当今功能最为强大、最为可靠的数据库管理系统之一。

Visual FoxPro 6.0 关系型数据库系统不仅可以简化数据管理,而且使应用程序的开发流程更为合理。Visual FoxPro 6.0 使数据组织、数据库规则定义和应用程序建立等工作变得简单易行。利用可视化的设计工具和向导,可以快速创建表单、查询和报表。Visual FoxPro 6.0 还提供了一个集成化的开发环境,不仅拥有功能强大的面向对象程序设计工具以及客户/服务器能力,而且支持 ActiveX,这些特点为快速开发功能全面的应用程序创造了良好的条件。

本书详细介绍了 Visual FoxPro 6.0 这一可视化数据库管理系统的使用方法和特点,并提供了一些实用技巧方面的知识。通过学习本书,读者将能深入掌握如何利用 Visual FoxPro 6.0 的设计工具和面向对象程序设计技术来创建大型、复杂的应用程序,并具备合理地组织、应用一个数据库系统来完成复杂数据管理任务的能力。

全书主要内容如下:程序设计简介,Visual FoxPro 面向对象程序设计概念,Visual FoxPro 6.0 对编程的改进,创建帮助,升迁 Visual FoxPro 数据库,设计客户/服务器应用程序,群体开发,企业级开发与国际化开发,使用外部库和访问 Visual FoxPro 6.0 API,共享访问,OLE 和 Internet 应用,程序的编译、调试、优化与发布。

本书内容深入浅出,图文并茂,实例丰富,是 Visual FoxPro 6.0 应用和开发人员必备的工具书,也可供数据库和信息管理系统开发人员学习参考。

内 容 简 介

本书详细介绍了 Visual FoxPro 6.0 这一可视化数据库管理系统的使用方法和特点,并提供了一些实用技巧方面的知识。通过学习本书,读者能深入掌握如何利用 Visual FoxPro 6.0 的设计工具和面向对象程序设计技术来创建大型、复杂的应用程序,并具备合理地组织、应用一个数据库系统来完成复杂数据管理任务的能力。

全书主要内容如下:程序设计简介,Visual FoxPro 面向对象程序设计概念,Visual FoxPro 6.0 对编程的改进,创建帮助,升迁 Visual FoxPro 数据库,设计客户/服务器应用程序,群体开发,企业级开发与国际化开发,使用外部库和访问 Visual FoxPro 6.0 API,共享访问,OLE 和 Internet 应用,程序的编译、调试、优化与发布。

本书内容深入浅出,图文并茂,实例丰富,是 Visual FoxPro 6.0 应用和开发人员必备的工具书,也可供数据库和信息管理系统开发人员学习参考。

目 录

第一章 简介 1

- 1.1 设计前的考虑 1
- 1.2 人工操作与编程 2
- 1.3 Visual FoxPro 的编程机制 3
- 1.4 有关语言概念 4
 - 1.4.1 存储数据 5
 - 1.4.2 处理数据 5
 - 1.4.3 使用过程和用户自定义函数 8
 - 1.4.4 流程控制 10
- 1.5 程序设计的基本步骤 13
 - 1.5.1 对问题进行说明 14
 - 1.5.2 分解问题 14
 - 1.5.3 编制模块 14
 - 1.5.4 测试模块 16
 - 1.5.5 组装全部模块 17
 - 1.5.6 整体测试 19
 - 1.5.7 使程序更可靠 19
 - 1.5.8 将程序转换成函数 20
- 1.6 帮助与示例文件 21

第二章 面向对象程序设计 22

- 2.1 Visual FoxPro 中的对象 22
 - 2.1.1 对象的属性 22
 - 2.1.2 对象的事件和方法程序 23
- 2.2 Visual FoxPro 中的类 24
 - 2.2.1 类、子类与继承性 24
 - 2.2.2 Visual FoxPro 的类分层结构 24
 - 2.2.3 什么情况下创建类 26
 - 2.2.4 Visual FoxPro 的基类 26
 - 2.2.5 扩展 Visual FoxPro 基类 27
 - 2.2.6 创建新类 28
 - 2.2.7 修改类定义 29
 - 2.2.8 创建类定义的子类 29
- 2.3 “类设计器”的使用 29

- 2.3.1 在控件类或容器类中添加对象 30
- 2.3.2 将属性和方法程序添加到类 30
- 2.4 保护和隐藏类成员 31
 - 2.4.1 指定属性的默认值 33
 - 2.4.2 为类指定设计时的外观 33
 - 2.4.3 使用类库文件 34
- 2.5 将类添加到表单中 34
- 2.6 调用父类的方法程序 35
- 2.7 在容器分层结构中引用对象 36
 - 2.7.1 相对引用 36
 - 2.7.2 设置属性 37
 - 2.7.3 设置多个属性 37
 - 2.7.4 调用方法程序 38
 - 2.7.5 响应事件 38
- 2.8 以编程方式定义类和对象 38
 - 2.8.1 保护和隐藏类成员 39
 - 2.8.2 由类创建对象 39
 - 2.8.3 在容器类中添加对象 40
 - 2.8.4 使用方法程序代码添加类和创建类 40
 - 2.8.5 指派方法程序代码和事件代码 40
 - 2.8.6 按类层次调用事件代码 41
 - 2.8.7 防止基类代码被执行 41
 - 2.8.8 创建表定位按钮集合 42
 - 2.8.9 定义表格控件 46
- 2.9 创建对象的引用 48
 - 2.9.1 返回对象的引用 49
 - 2.9.2 从内存中释放对象和引用 49
 - 2.9.3 检查对象是否存在 49
- 2.10 创建成员数组 50
- 2.11 创建对象数组 50
- 2.12 使用对象存储数据 51
- 2.13 事件模型 53
 - 2.13.1 Visual FoxPro 中的事件 53
 - 2.13.2 追踪事件序列 55

2.13.3 监视事件发生	56	4.3 参数的传递与接收	88
2.13.4 为事件指定代码	59	4.4 向 Visual FoxPro 返回值	92
第三章 新版本的主要改进	60	4.4.1 从 ActiveX 控件返回值	92
3.1 新的和改进的语言元素	60	4.4.2 从一个 FLL 库中返回值	93
3.2 性能、坚固性和实用性	67	4.5 向 Visual FoxPro API 传 参数	94
3.2.1 Visual FoxPro 性能的提高	67	4.5.1 Visual FoxPro API 数据类型	95
3.2.2 Visual FoxPro 坚固性的提高	68	4.5.2 Visual FoxPro API 数据结构	95
3.2.3 Visual FoxPro 实用性的提高	68	4.6 访问 Visual FoxPro 变量和 字段	96
3.3 新增和改进了的向导及 生成器	68	4.6.1 管理内存	97
3.4 改进的应用程序框架	69	4.6.2 使用句柄	97
3.4.1 启动“应用程序生成器”	70	4.6.3 理解堆栈	99
3.4.2 文件	70	4.6.4 遵守句柄规则	99
3.4.3 创建框架	71	4.7 FLL 库和 ActiveX 控件的 连编与调试	99
3.5 对 2000 年的支持	71	4.7.1 连编项目	99
3.5.1 严格的日期格式	72	4.7.2 调试一个 ActiveX 控件 或 FLL 库	100
3.5.2 SET STRICTDATE	72	第五章 扩充 Visual FoxPro 的 功能	102
3.5.3 严格日期格式错误	73	5.1 关于外部库	102
3.5.4 “选项”对话框	74	5.2 使用 ActiveX 控件和对象	103
3.5.5 DATE() 和 DATETIME() 函数	74	5.3 使用动态链接库	103
3.5.6 FDATE() 函数	74	5.3.1 调用 DLL 函数	104
3.5.7 Century 属性	74	5.3.2 向 DLL 传递参数	104
3.6 Access 和 Assign 方法程序	74	5.4 使用 Visual FoxPro 库	105
3.6.1 Access 和 Assign 方法 程序的优点	75	第六章 升迁 Visual FoxPro 数据 库	107
3.6.2 创建 Access 和 Assign 方法 程序	75	6.1 构造应用程序的本地原型	107
3.6.3 表单和类设计器	76	6.1.1 减少开发时间	107
3.6.4 THIS-ACCESS 方法程序	77	6.1.2 降低开发费用的同时满足 用户要求	107
3.7 对 GIF 和 JPEG 图形的支持	79	6.1.3 有助于成功地完成任务	107
3.7.1 Visual FoxPro 语言	79	6.1.4 构造应用程序的本地原型	108
3.7.2 Visual FoxPro 控件和对象	80	6.1.5 使用升迁向导	108
3.7.3 Visual FoxPro 界面	81	6.2 升迁到 SQL Server 上	108
第四章 访问 Visual FoxPro API	82	6.2.1 准备 SQL Server 端	109
4.1 创建 ActiveX 控件或 FLL 库	82	6.2.2 准备客户端	110
4.1.1 创建一个基本的 ActiveX 对象	82		
4.1.2 创建一个基本的 FLL 库	83		
4.2 调用 Visual FoxPro API	87		

6.2.3 备份数据库.....	110	7.5.3 加速查询和视图的运行.....	154
6.2.4 关闭表.....	111	7.5.4 加速表单的运行.....	155
6.2.5 启动升迁向导.....	111	7.5.5 提高更新和删除的性能.....	156
6.2.6 升迁向导如何工作.....	111		
6.2.7 完成 SQL Server 升迁过程	118	第八章 群体开发和企业级开发	159
6.2.8 对 SQL Server 采取的步骤	118	8.1 什么是源代码管理.....	159
6.2.9 Visual FoxPro 对客户端 采取的步骤	120	8.1.1 源代码管理工具的特性.....	159
6.3 升迁到 Oracle 上	121	8.1.2 源代码管理软件.....	160
第七章 构造客户/服务器应用		8.1.3 集成源代码管理系统和 Visual FoxPro 项目	160
程序	122	8.1.4 启用源代码管理系统	161
7.1 设计客户/服务器应用程序.....	122	8.2 在源代码管理系统下管理 Visual FoxPro 项目	162
7.1.1 性能方面的考虑.....	122	8.2.1 处理项目文件和项目列表 文件.....	162
7.1.2 确保数据完整性和开发 准确性	128	8.2.2 将项目置于源代码管理之下.....	163
7.2 实现方法:远程视图和 SQL pass-through 技术	129	8.2.3 向源代码管理项目添加文件.....	163
7.3 使用 SQL pass-through 技术	130	8.2.4 加入一个现有的源代码管理 项目	164
7.3.1 使用 SQL pass-through 函数.....	130	8.2.5 更新项目列表.....	164
7.3.2 向数据源传递 SQL 语句	132	8.2.6 解除一个项目的源代码管理.....	165
7.3.3 建立参数化查询.....	132	8.2.7 从源代码管理项目中移去 文件.....	165
7.3.4 使用 SQL Server 的输入/输出 参数.....	133	8.2.8 在多个源代码管理项目之间 共享文件	166
7.3.5 建立与远程数据的外部连接.....	134	8.3 在源代码管理项目中 管理文件	166
7.3.6 对 SQL 实施 ODBC 扩展	135	8.3.1 处理多文件组件.....	166
7.3.7 用 SQL pass-through 管理 连接.....	135	8.3.2 签出文件.....	167
7.3.8 用 SQL pass-through 处理远程 数据.....	141	8.3.3 签入文件.....	167
7.3.9 设置远程数据的临时表属性.....	141	8.3.4 比较文件或项目.....	169
7.3.10 选择有效的 SQL pass-through 处理方式	146	8.3.5 检查表单、报表和其他表文件的 区别.....	169
7.3.11 处理多个结果集合	147	8.3.6 显示文件和项目信息	170
7.3.12 控制数据类型的转换	148	8.4 群体开发和修改数据库	171
7.4 处理 SQL pass-through 错误	150	8.5 群体开发类库	171
7.5 客户/服务器性能优化	150	8.6 企业级开发	172
7.5.1 优化连接的使用.....	151	8.6.1 使用 Visual FoxPro 作为应用 程序的前台	173
7.5.2 加速数据检索.....	152	8.6.2 扩展 Visual FoxPro 中可视化设计 工具的功能.....	173
		8.6.3 建立子类.....	173

8.6.4 使用 ActiveX 控件	174	代码页	190
8.6.5 集成其他程序的功能	174	9.6 在国际化应用程序中	
8.6.6 使用向导	175	排序数据	191
8.6.7 使用自动服务	175	9.6.1 深入了解排序次序	191
8.6.8 扩展 Visual FoxPro 的数据		9.6.2 排序次序准则	191
存储能力	176	9.6.3 指定排序次序	192
8.6.9 使用 Visual FoxPro 作为		9.6.4 检查排序次序	192
数据源	177	9.6.5 认识排序次序的影响	193
8.6.10 在其他应用程序中使用 Visual		9.6.6 使用索引	194
FoxPro 的数据	177	9.7 处理双字节字符集	194
8.6.11 在其他应用程序中使用 Visual		9.7.1 使用 DBCS 字符命名对象	194
FoxPro 对象和命令	178	9.7.2 排序 DBCS 数据	194
8.6.12 使用 Visual FoxPro 建立		9.8 创建或修改程序	195
数据库	178	9.8.1 测试国际化版本	195
8.6.13 使用 Visual FoxPro 作为一个		9.8.2 使用字符串	195
WWW 搜索引擎	179	9.8.3 处理日期、时间和货币的格式	196
8.6.14 如何将 Visual FoxPro 作为 Web		9.8.4 使用预处理指令	197
网的搜索引擎	180	9.9 在国际化应用程序中	
第九章 开发国际化应用程序	181	管理文件	197
9.1 准备工作	181	第十章 添加 OLE	199
9.1.1 准备国际化数据	181	10.1 设计 OLE 应用程序	199
9.1.2 如何编写代码	182	10.1.1 链接或嵌入 OLE 对象	199
9.1.3 如何设计用户界面	182	10.1.2 添加绑定型或非绑定型 OLE	
9.1.4 测试应用程序	182	对象	201
9.2 设计界面	182	10.2 在应用程序中添加 OLE	
9.2.1 创建应用程序信息	182	对象	201
9.2.2 设计菜单和表单	183	10.2.1 在表中添加 OLE 对象	201
9.2.3 使用图标和位图	183	10.2.2 在表中追加 OLE 对象	202
9.3 输入国际化数据	184	10.2.3 刷新 Microsoft Graph	203
9.3.1 输入国际化字符	184	10.2.4 在表单中添加 OLE 对象	204
9.3.2 追加和复制国际化数据	185	10.2.5 与 OLE 对象交互作用	205
9.4 处理代码页	185	10.3 使用 ActiveX 控件	206
9.4.1 深入理解 Visual FoxPro 中的		10.3.1 向表单添加 ActiveX 控件	206
代码页	185	10.3.2 管理绑定型 ActiveX 控件	207
9.4.2 指定 dbf 文件的代码页	186	10.4 应用自动服务管理对象	207
9.4.3 指定文本文件的代码页	188	10.4.1 管理外部对象的属性	207
9.4.4 确定项目文件的代码页	189	10.4.2 设置时间期限	208
9.4.5 指定变量的代码页	189	10.4.3 访问对象集合	209
9.4.6 禁止对字符型或备注型字段中的		10.4.4 使用对象数组	209
数据进行转换	189	10.4.5 释放外部对象	210
9.5 Visual FoxPro 所支持的			

10.5 派生对象的子类	210	12.1.1 拖放数据	245
10.6 从其他应用程序中控制 Visual FoxPro	211	12.1.2 拖动源	245
10.6.1 Visual FoxPro 的 Application 对象 模型	212	12.1.3 放落目标	246
10.6.2 通过集合属性访问对象	212	12.1.4 移动数据	246
10.7 创建 Automation 服务程序	213	12.1.5 复制数据	246
10.7.1 创建服务程序	213	12.1.6 放落目标和拖动源不支持 OLE 拖放	247
10.7.2 编译服务程序	214	12.1.7 取消操作	247
10.7.3 注册一个 Automation 服务程序	216	12.2 对设计时刻的 OLE 拖放的 支持	247
10.7.4 使用 Automation 服务程序	217	12.3 对运行时刻的 OLE 拖放的 支持	248
10.7.5 从 Automation 服务程序引起错误 或返回错误	217	12.3.1 早期版本 Visual FoxPro 中的 拖放	249
10.8 使用远程自动服务	217	12.3.2 DataObject 对象	249
10.8.1 远程自动服务	218	12.3.3 固有和人工 OLE 拖放模式	249
10.8.2 系统安全策略选项	219	12.4 Active Document	250
10.8.3 在远程自动服务中使用权限	219	12.4.1 创建 Active Document	251
10.8.4 远程自动服务疑难解答	220	12.4.2 对表单对象的更改	253
第十一章 共享访问	221	12.4.3 Active Document 中的菜单	253
11.1 控制对数据的访问	221	12.4.4 运行 Active Document	255
11.1.1 访问数据	221	12.4.5 Active Document 安全性	256
11.1.2 锁定数据	222	12.4.6 Internet Explorer 注意事项	257
11.1.3 解锁数据	225	12.4.7 Active Document 示例	257
11.1.4 使用数据工作期	225	12.5 Automation 服务程序的 改进	257
11.1.5 缓冲访问数据	227	12.5.1 Visual FoxPro 6.0 Automation 服务程序的改进	258
11.2 更新数据	231	12.5.2 Automation 服务程序编程的注意 事项	261
11.2.1 使用缓冲进行数据更新	231	12.6 在 Web 页面上发布信息	261
11.2.2 使用事务处理管理进行数据 更新	232	12.6.1 Web 发布	261
11.2.3 使用视图管理进行数据更新	238	12.6.2 使用 Active 文档	266
11.3 管理冲突	241	12.7 在 WWW 上搜索信息	269
11.3.1 管理缓冲冲突	241	第十三章 创建帮助文件	272
11.3.2 检测并解决冲突	243	13.1 HTML 帮助	272
11.3.3 使用备注字段检查冲突	244	13.1.1 计划对联机帮助的访问	273
11.3.4 管理冲突的规则	244	13.1.2 添加帮助上下文的相关性	273
第十二章 互操作性和 Internet 应用	245	13.1.3 实现“这是什么?”帮助	274
12.1 OLE 拖放	245	13.1.4 编程实现帮助特性	275
		13.1.5 发布已编译的 HTML 帮助	

系统	275	15.1.4 控制事件循环	315
13.2 用 WinHelp 4.0 创建图形		15.1.5 恢复初始的开发环境	315
方式的帮助	276	15.1.6 将程序组织为一个主文件	315
13.2.1 选择帮助特性	276	15.2 将文件加入到项目中	316
13.2.2 计划对联机帮助的访问	276	15.2.1 人工添加文件到某一项目	317
13.2.3 帮助菜单	277	15.2.2 引用或排除可修改的文件	317
13.2.4 添加帮助的上下文相关性	277	15.3 为一个项目创建应用程序	318
13.2.5 实现“这是什么?”帮助	278	15.3.1 测试项目	318
13.2.6 编程实现帮助特性	279	15.3.2 从项目中绑定应用程序	319
13.2.7 使用 WinHelp 函数	279	15.4 测试程序	320
13.2.8 指定 WinHelp 参数	280	15.4.1 建立测试环境	321
13.3 创建 dbf 样式帮助	282	15.4.2 设置验证信息	321
13.3.1 设计 dbf 样式帮助	283	15.4.3 查看事件发生的序列	321
13.3.2 查看 dbf 样式帮助文件示例	283	15.5 逐步发现错误	322
13.3.3 使用 dbf 样式帮助	285	15.5.1 启动调试工作期	322
13.3.4 定制 dbf 样式帮助	286	15.5.2 跟踪代码	323
第十四章 快速开发工具	288	15.5.3 挂起程序的执行	323
14.1 组件管理库	288	15.5.4 查看存储的值	327
14.1.1 使用“组件管理库”管理项目	288	15.5.5 显示输出结果	328
14.1.2 定制“组件管理库”	291	15.6 记录代码的覆盖范围	329
14.1.3 “组件管理库”的对象成员	292	15.7 处理“运行时刻错误”	330
14.1.4 “组件管理库”的类库	295	15.7.1 预防错误	330
14.1.5 “组件管理库”的表结构	296	15.7.2 处理过程的错误	331
14.2 代码范围分析器	297	15.7.3 处理类和对象中的错误	332
14.2.1 代码范围分析器日志文件	297	15.7.4 从错误处理代码中返回	334
14.2.2 检查应用程序的代码范围并		第十六章 应用程序的优化	335
进行分析	298	16.1 优化表和索引	335
14.2.3 确保“代码范围分析器”的正确		16.1.1 使用索引	335
运行	300	16.1.2 优化连接	335
14.2.4 代码范围分析器加载项	300	16.1.3 使用项目管理器	336
14.3 项目管理器挂接程序	303	16.1.4 优化表和索引的一般提示	336
14.3.1 Project 对象的分层结构	303	16.2 加速数据访问	337
14.3.2 Project 对象的结构	307	16.2.1 掌握 Rushmore 技术	337
14.3.3 语言的改进	307	16.2.2 在表中使用 Rushmore	338
14.3.4 项目事件	307	16.2.3 关闭 Rushmore	339
第十五章 程序编译与调试	312	16.2.4 优化 Rushmore 表达式	340
15.1 构造应用程序框架	312	16.3 优化表单和控件	342
15.1.1 设置起始点	313	16.3.1 使用数据环境	343
15.1.2 初始化环境	313	16.3.2 在表单集里限制表单数目	343
15.1.3 显示初始的用户界面	314	16.3.3 在页框中动态加载页面控件	343
		16.3.4 动态绑定控件与数据	343

16.3.5 延迟屏幕刷新	344	17.2.4 在应用程序中包含资源	356
16.3.6 在经常使用的方法程序中减少 代码	344	17.2.5 删除受限制的 Visual FoxPro 功能和文件	358
16.4 优化应用程序	344	17.3 定制要发布的应用程序	360
16.4.1 提高性能的一般技巧	345	17.3.1 保护源代码并将其存档	360
16.4.2 使用名称表达式代替宏替换	345	17.3.2 调用错误处理例程和终止系统 运行例程	361
16.4.3 有效地引用对象属性	345	17.3.3 给应用程序添加帮助信息	361
16.5 优化 ActiveX 控件	346	17.3.4 修改应用程序的外观	362
16.6 优化自动操作	347	17.3.5 备份源代码	362
16.7 在多用户环境中优化应用 程序	347	17.3.6 生成应用程序	363
16.7.1 调整锁定重试的间隔	348	17.4 准备制作发布磁盘	363
16.7.2 有效地使用事务处理	348	17.4.1 创建发布目录	363
16.8 优化对远程数据的访问	348	17.4.2 创建发布磁盘	364
16.8.1 只检索需要的数据	349	第十八章 创建发布磁盘	365
16.8.2 有效地更新远程表	349	18.1 发布目录树和安装向导	365
16.8.3 批处理方式发送语句	350	18.1.1 发布目录树	365
16.8.4 设置数据包大小	351	18.1.2 安装向导	366
16.8.5 延迟对备注和二进制数据的 检索	351	18.2 如何用安装向导创建发布 磁盘	367
16.8.6 将待查阅数据保存在 本地机上	351	18.2.1 指定发布目录树	367
16.8.7 创建本地规则	352	18.2.2 选择可选组件	367
16.9 优化国际化应用程序	352	18.2.3 指定磁盘映像	367
16.9.1 有效地使用排序序列	352	18.2.4 定制要发布的安装对话框	368
16.9.2 用多个排序序列发送 SQL SELECT 命令	352	18.2.5 指定默认文件的安装目的地	368
第十七章 建立可发布的应用程序	354	18.2.6 查看文件总结报告	368
17.1 发布过程	354	18.2.7 完成安装向导过程	368
17.2 准备工作	354	附录 A 常用 Visual FoxPro 的 属性	369
17.2.1 选择连编类型	354	附录 B 常用系统内存变量	378
17.2.2 考虑硬件、内存和网络问题	355	附录 C 常见错误代码	380
17.2.3 确保正确运行	355	附录 D 常用术语	384

第一章 简介

Visual FoxPro 关系型数据库系统不仅可以简化数据管理,而且使应用程序的开发流程更为合理。Visual FoxPro 使数据组织、数据库规则定义和应用程序建立等工作变得简单易行。利用可视化的设计工具和向导,可以快速创建表单、查询和报表。Visual FoxPro 还提供了—个集成化的开发环境,不仅拥有功能强大的面向对象程序设计工具以及客户/服务器能力,而且支持 ActiveX,这些特点为快速开发功能全面的应用程序创造了良好的条件。

Visual FoxPro 将过程化程序设计与面向对象程序设计结合在一起,帮助用户创建出功能强大、灵活多变的应用程序。从概念上讲,程序设计就是为了完成某一具体任务而编写一系列指令;从深层次来看,Visual FoxPro 程序设计涉及到对存储数据的操作。

1.1 设计前的考虑

Visual FoxPro 应用程序通常由以下几部分组成:一个或多个数据库,设置应用程序系统环境的主程序,用户界面(诸如表单、工具栏和菜单等)。此外,还可以包括查询和报表,允许用户检索或输出自己的数据。

在设计 Visual FoxPro 应用程序前,认真细致的规划可以节省时间、精力和资金。在规划阶段,应该让最终用户更多地参与进来。无论多么仔细的规划,在项目实施过程中也需要不断润色,并接受最终用户的反馈。

在开发之前所做的设计方案往往会对最终结果产生很大的影响,许多问题都应在深入开发之前加以考虑。例如:这个应用程序的用户是谁,用户的主要操作是什么,要处理的数据集合有多大,是否要使用后台数据服务器,以及是单用户还是网络上的多用户等。

1. 一般的用户操作

最终用户可能需要处理顾客、订单或各种产品,他们处理信息的方式将决定应用程序如何进行数据操作。例如,位于目录 Visual Studio ... \ Samples \ Vfp98 \ Tastrade 中的程序所使用的订单输入表单也许对一些应用程序有用,但这个表单可能并不适用于管理货物清单或追踪销售记录。

2. 数据库的大小

当需要处理庞大的数据集合时,考虑最多的恐怕是如何提高性能。有时,可能需要调整用户在数据之间移动的方式。例如,如果一个表中只有二三十个记录,那么记录指针一次只移动一个记录不会有什么问题,但如果表中有二三万个记录,就必须为用户提供能找到所有数据的其他方法(比如增加搜索列表、对话框、过滤器和定制查询等)。

3. 单用户和多用户

创建应用程序时,最好考虑到几个用户同时访问数据库的情况。Visual FoxPro 提供了一些技术,使开发者能够很容易地进行共享访问方面的编程。

4. 国际化考虑

若事先知道应用程序仅在单一语言环境中使用,可以不考虑国际化问题。但是,如果想扩大市场,或用户要处理国际化的数据和环境设置,则应在创建应用程序时考虑这些因素。

5. 本地数据和远程数据

若应用程序需要处理远程数据,则必须使用新的管理方法,这种方法与处理本地 Visual FoxPro 数据有所不同。

1.2 人工操作与编程

通常,在程序中能够完成的工作都可以通过人工操作来完成,但需要足够的时间。例如,在 Customer 表中要查找一个特定顾客的信息(如 Ernst Handel 公司),可以通过人工执行一系列指令来实现。若要以人工方式在表中查找一个订单,可以按照下列步骤进行:

(1) 在“文件”菜单中选择“打开”命令。

(2) 在“文件类型”列表框中选择“表”。

(3) 进入 Visual FoxPro 目录下的 Samples \ Data \ 子目录,在文件列表中双击 Customer.dbf。

(4) 在“显示”菜单中选择“浏览”命令。

(5) 滚动遍历表,查找 Company 字段为“Ernst Handel”的记录。

若要通过编程实现上述目的,可以在“命令”窗口中键入下列 Visual FoxPro 命令:

```
USE Samples \ Data \ Customer
```

```
LOCATE FOR Company = "Ernst Handel"
```

```
BROWSE
```

在表中找到这家公司以后,就可以执行预定的操作(如将订单的最高限额提高 3 个百分点)。

若要通过人工操作提高订单的最高限额,可以按照下列步骤进行:

(1) 选定 Maxordamt 字段。

(2) 将 Maxordamt 字段值乘以 0.03,再加上原来的字段值,然后在该字段处键入得到的结果。

若要通过编程实现以上结果,可以在“命令”窗口中键入以下命令:

```
REPLACE Maxordamt WITH Maxordamt * 1.03
```

如果要修改某个顾客订单的最高限额,用人工方式或在“命令”窗口中键入指令都很容易实现,但如果要把所有订单的最高限额都提高 3 个百分点,恐怕就不是一件轻而易举的事情。若还是用人工方式来做,不但费时费力,而且容易出错。解决此类问题的更好办法是编写一个可执行的程序文件,该文件可以轻松无误地完成这一工作。

增加所有顾客订单最大限额的示例程序如下:

```
USE Samples \ Data \ Customer      && 打开 Customer 表
```

* 浏览表中所有记录, 针对每条记录执行 SCAN 与 ENDSCAN 之间的所有指令
SCAN

* 将订单的最高限额提高 3 个百分点(分号 (;)指令将继续下一行指令)

REPLACE Maxordamt WITH Maxordamt * 1.03

ENDSCAN && 结束为浏览表中每一记录而执行的代码
与在“命令”窗口中单独键入每条命令相比，运行程序有如下优点：

- (1) 程序可被修改并重新运行。
- (2) 可从菜单、表单和工具栏启动程序。
- (3) 一个程序可调用其他程序。

以下各节将详细介绍 Visual FoxPro 程序设计的概念、结构和流程。

1.3 Visual FoxPro 的编程机制

Visual FoxPro 程序由代码组成,代码包括以命令形式出现的指令、函数以及 Visual FoxPro 可以理解的任何操作。这些指令包含在“命令”窗口、程序文件、“表单设计器”或“类设计器”的事件或方法程序代码窗口、“菜单设计器”的过程代码窗口、“报表设计器”的过程代码窗口中。

1. 使用“命令”窗口

可以在“命令”窗口中键入 Visual FoxPro 命令并按 Enter 键执行。若要重新执行该命令,还可以将光标移到此命令所在行并按 Enter 键。

甚至可以在“命令”窗口中像执行独立程序一样执行多行代码。要在“命令”窗口中运行多行代码的步骤为：

- (1) 选择代码行。
- (2) 按 Enter 键或在快捷菜单中选择“运行所选区域”。

因为“命令”窗口是一个编辑窗口,所以在编辑命令时可以使用 Visual FoxPro 提供的编辑工具。在“命令”窗口中可以编辑、插入、删除、剪切、复制和粘贴正文。

在“命令”窗口中执行命令的优点是:能够立即执行被键入的命令,不需要将其保存为文件并用程序方式执行。此外,在菜单和对话框中所做的选择可以马上转换为“命令”窗口中的命令,用户可以将这些命令复制并粘贴到 Visual FoxPro 程序中,然后重复执行这些程序。这样做可以很容易地重复执行成百上千条的命令。

2. 创建程序

Visual FoxPro 程序是包含一系列命令的文本文件。在 Visual FoxPro 中,可以通过以下途径创建程序:

- (1) 在“项目管理器”中选定“代码”选项卡中的“程序”项。
 - (2) 选择“新建”命令。
- 或者通过以下途径:
- (1) 在“文件”菜单中选择“新建”命令。
 - (2) 在“新建”对话框中选择“程序”。
 - (3) 选择“新建文件”按钮。

也在“命令”窗口中键入：

MODIFY COMMAND

这时 Visual FoxPro 打开一个名为“程序 1”的新窗口，这时就可以键入应用程序了。

3. 保存程序

创建程序后，应注意保存。

若保存程序，从“文件”菜单中选择“保存”命令。

若要关闭一个没有保存的程序，则会弹出相应对话框，提示用户是保存还是放弃已做的修改。

若保存了一个由“项目管理器”创建的程序，则该程序被加入项目中。

若保存一个尚未命名的程序，则会打开“另存为...”对话框，提示用户在其中为程序指定程序名。程序保存后，用户可以运行或修改它。

4. 修改程序

程序保存后可以修改。首先，按以下四种方式之一打开想要修改的程序：

(1) 若程序包含在一个项目中，则在“项目管理器”中选定它并选择“修改”命令。

(2) 在“文件”菜单中选择“打开”命令，这时弹出一个包含文件列表的对话框，在“文件类型”列表框中选择“程序”，然后在文件列表中选定要修改的程序，按下“确定”按钮。

(3) 在“命令”窗口中按如下方式键入要修改的程序名：

MODIFY COMMAND

(4) 在“命令”窗口中键入：

MODIFY COMMAND ?

然后，从文件列表中选择要修改的程序，并选择“打开”。

打开文件之后便可进行修改，修改完毕后应注意保存。

5. 运行程序

程序创建之后便可运行。若要运行程序，有三种方法：

(1) 若程序包含在一个项目中，则在“项目管理器”中选定它并选择“运行”命令。

(2) 在“程序”菜单中选择“运行”菜单项。在程序列表中，选择想要运行的程序，单击“运行”按钮。

(3) 在“命令”窗口中，按如下方式键入 DO 以及要运行的程序名：

DO Myprogram

6. 使用 Visual FoxPro 设计工具编写代码

借助“表单设计器”、“类设计器”和“菜单设计器”，程序员可以很简单地把程序代码与用户界面连接起来，这样应用程序便可响应用户的输入并执行相应的代码。同样，“报表设计器”将程序代码与报表文件联系起来，以此定制结构复杂并且符合用户要求的报表。如果想充分发挥 Visual FoxPro 的强大功能，可使用刚才提到的设计工具。

1.4 有关语言概念

设计程序时，就是用一系列指令存储数据并操作这些数据。程序设计的原材料是数据和数据存储容器，而处理这些原材料的工具是命令、函数和操作符。