可视化

FoxPro 3.

编程手册

张海兰 木林森 编著
王小丽 仙珊珊 审校

航空工业出版社

可视化 FoxPro 3.0 编程手册

张海兰　木林森　编著
王小丽　张海芝　审校

航空工业出版社

1996

内 容 提 要

本书重点介绍了数据库管理系统 Visual FoxPro 3.0 的编程方法和编程技巧。全书首先对 Visual FoxPro 3.0 的编程基础知识作了介绍；然后讲述了 Visual FoxPro 3.0 的面向对象编程技术；接着通过具体实例介绍了数据库、表、视图、格式和控件的使用，并对菜单、查询、报表的设计技术作了介绍；最后对 OLE 技术和共享程序的设计技术作了介绍。

结合具体实例介绍新概念、新方法和新技术是本书的一大特色，全书语言浅显易懂、内容新颖，可供管理软件开发人员、大专院校师生、计算机爱好者和各种培训班使用。

图书在版编目(CIP)数据

可视化 FoxPro 3.0 编程手册/木林森等编著－北京：
航空工业出版社，1996.3
ISBN 7-80046-943-3

Ⅰ.可… Ⅱ.木… Ⅲ.数据库管理系统-手册 Ⅳ.TP311.13-62

中国版本图书馆 CIP 数据核字(95)第 11096 号

航空工业出版社出版发行
（北京市安定门外小关东里 14 号 100029）
海洋出版社印刷厂印刷 全国各地新华书店经售
1996 年 4 月第 1 版 1996 年 4 月第 1 次印刷
开本：787×1092 1/16 印张：19.375 字数：481 千字
印数：1－4000 定价：28.00 元

前　言

FoxPro 是美国 Fox 软件公司推出的全新的微机关系型数据库管理系统，它所具有的强大的性能、丰富而完整的工具、无以伦比的速度、极其友好的图形用户界面、简单的数据存取方式、良好的兼容性、独一无二的跨平台特性以及真正的可编译性，使其成为目前最快、最完美的数据库管理系统。

自从 Fox 软件公司并入 Microsoft 公司之后，FoxPro 版本不断更新、功能不断增强，其最新版本 Visual FoxPro 3.0 不仅仅是新增加了 150 个命令与函数，更为重要的是引入了可视化编程技术，提供了众多的工具（如 Wizards、Builders 和 Designers 等）和工具条，使得对一些常用功能的操作更为简单直观。Visual FoxPro 3.0 对数据库概念作了根本上的修正，使得数据库已不再是传统上的单纯用户存储数据的 .DBF 文件，而是表以及表的视图、连接、关联、存储过程、规则、缺省值、触发器等的集合和管理者。Visual FoxPro 3.0 不仅支持面向过程的编程方法，而且支持面向对象的编程方法，通过使用 Visual FoxPro 3.0 的对象模型，用户可使用面向对象编程的所有特征：类、继承、封装、多态性及子类。利用 Visual FoxPro 3.0 的事件模型，用户既可在属性窗口中直观地控制事件，也可用语言以编程的方式来控制，从而生成真正事件驱动的应用程序。Visual FoxPro 3.0 可方便地与其他应用程序共享数据，也可方便地与其他应用程序交换数据。Visual FoxPro 3.0 支持客户/服务器计算，在开发客户/服务器应用程序时将 Visual FoxPro 3.0 作为前台，使用 SQL 语言直接访问服务器。所有这些功能给用户带来了极大的方便。

为使读者尽快掌握 Visual FoxPro 3.0 的使用方法和编程方法，我们应航空工业出版社的约请，特编写了这套可视化 FoxPro 3.0 丛书。全套丛书共分三册，即使用手册、编程手册和语言实用详解。本书为《可视化 FoxPro 3.0 编程手册》，重点介绍了 Visual FoxPro 3.0 的编程方法和编程技巧。

本书由张海兰和木林森主编，参与本书编写工作的还有章东灵、林珊珊、刘志强、韦玉、郑群、李海林、王小华、叶光明、郭福林、张强华、刘东平、李正格、何志强和莫为东。

由于时间仓促，不当之处在所难免，尚希读者批评指正。

编　者

1996 年 3 月

目　录

第1章　Visual FoxPro 3.0 编程基础

FoxPro 是美国 Fox 软件公司推出的全新的微机平台上的关系型数据库管理系统，它具有强大的性能、丰富的工具、无与伦比的速度、良好的图形用户界面、简单的数据存取、完备的兼容性、独一无二的跨平台特性以及真正的可编译性等，使得 FoxPro 系统日益成为目前最快、最完美的数据库管理系统。

自从 Fox 软件公司并入 Microsoft 公司之后，其雄霸微机平台数据库管理系统市场的势头更加强劲。该公司在 1993 年 3 月推出的 FoxPro 2.5 在市场上获得了极大的成功，它不但成为微机上的首选数据库产品，而且可以运行在 MS-DOS，Windows，Macintosh，UNIX 等多种操作系统环境下。更让 Fox 用户欣喜的是 FoxPro 产品间的完全兼容性，以及更强大的工具组。

随着面向对象程序设计技术的成熟与推广，可视化编程技术的引入，Microsoft 于 1995 年 9 月成功地推出了新一代的 FoxPro 系列产品 Visual FoxPro 3.0。

Visual FoxPro 3.0 不但是一个强大的交互式的数据管理工具，而且又是一个可以通过应用程序全面管理数据的语言系统，它不但支持传统的面向过程的编程方法，而且提供了强有力的面向对象编程技术，因此，对于 Visual FoxPro 3.0，充分理解和应用它所提供的面向对象编程技术和事件驱动方式，将最大限度地发挥 Visual FoxPro 3.0 的性能。

1.1　FoxPro 的背景及其发展

FoxPro 是美国 Fox 软件公司自 FoxBase 系列软件之后，在 Fox 数据库应用系列方面又推出的一个 PC 平台上的杰出的关系型数据库管理系统软件。Fox 软件公司自 1985 年开始，陆续推出了 FoxBase 1.0，FoxBase 2.0，FoxBase +2.10，FoxPro 1.0，FoxPro 2.0 等版本的 Fox 系列关系数据库管理软件产品。在并入 Microsoft 公司后，其产品的生产势头更为强劲，于 1993 年 3 月推出了 FoxPro 2.5 for MS-DOS，FoxPro 2.5 for Windows，FoxPro 2.5 for Unix，FoxPro 2.5 for Macintosh 等多平台 FoxPro 版本，之后又推出了支持面向对象程序设计的 FoxPro 2.6，至 1995 年 9 月成功地推出了 Visual FoxPro 3.0。

数据库理论的研究在 70 年代后期已进入了成熟阶段，随着 80 年代初期微型计算机的普及和性能的大幅度提高，Ashton Tate 公司的 dBASE 关系数据库产品很快进入了微机世界，并取得了令人欣喜的成功。由于 dBASE 具有简单、易操作、功能较强、交互性好等特点，迅速成为微机数据库的主导产品，形成了 dBASE II，dBASE III，dBASE III plus，dBASE IV 系列产品，功能逐渐加强。尽管 dBASE 系列产品在实际应用上存在一些问题和缺陷，然而，正是由于 dBASE 产品的广泛使用，带来了 PC 平台关系数据库产品市场的繁荣。

1986 年，Fox 软件公司推出的与 dBASE III plus 全兼容的 FoxBase$^+$，给 PC 平台的关系数据库产品带来了巨大的影响，以及以后推出的 FoxBase 2.0 和 FoxBase$^+$ 2.10 两个版本，不仅在速度上全面超越了前期的各产品，而且扩充了许多有利于开发人员的语言功能，更为

重要的是提供了良好的界面和较丰富的工具。

随着软件技术和数据库技术的飞速发展，PC 平台上的关系数据库管理系统的日益成熟，拥有更多的功能和应用工具，并且有面向对象程序设计的特点，以及图形用户界面的广泛使用，网络技术的更加普及，多媒体技术的应用等，使得 PC 平台上的 DBMS 需要有一个质量的提高，能更加适应软件技术和数据库技术的发展，正是在这一背景下，产生了 FoxPro。

自 1989 年 Fox 软件公司正式推出的 FoxPro 1.0 并以此作为 FoxBase 的替代产品开始，PC 平台上的 DBMS 开始飞速的发展。在 FoxPro 1.0 中初步引入了图形用户界面设计和字符窗口技术，并通过窗口和菜单系统在 FoxPro 集成环境中实现数据库的基本管理和操作。FoxPro 的功能强大，运行速度快，语言能力强，具有更丰富的命令、函数和工具，日益成为 XBase 语言标准。

1991 年 7 月问世的 FoxPro 2.0，除了继续保持原有的兼容性外，由于使用了 Rushmore 查询优化技术、RQBE 举例相关查询技术、SQL-SELECT 查询技术、Distribution Kit 编译技术、C 语言接口技术以及提供的诸如报表、屏幕、菜单、标签、项目管理等工具，使得 FoxPro 的性能有了一次质的飞跃。

1992 年 6 月，Fox 软件公司并入 Microsoft 公司后，于 1993 年 3 月推出了更为成功的 FoxPro 2.5，这是一个跨平台的 Fox 产品，它使得 FoxPro 可以在 MS-DOS，Windwos，UNIX，Macintosh 等平台上运行。同时，FoxPro 的图形界面技术、查询技术、自动生成技术等更有了长足的发展。随后的 FoxPro 2.6，增加了面向对象编程的能力。

而于 1995 年 9 月问世的 Visual FoxPro 3.0，不但比 FoxPro 2.5 有了极大的变化，而且更为主要的是它将 FoxPro 扩充成了一种可视编程的语言，使得 Visual FoxPro 3.0 成为面向对象的程序设计语言。

Visual FoxPro 3.0 不仅仅是新增加了 150 个命令与函数，而且引入了可视技术，提供了众多的工具条，使得对一些常用功能的操作更为简单直观。Visual FoxPro 3.0 中，对数据库概念作了根本上的修改，使得数据库已不再是传统上的单纯用户存储数据的 .DBF 文件，传统上的用于存储数据的 .DBF 文件已用表来代替，而 Visual FoxPro 3.0 中的数据库，就是这样的表，以及表的视图、连接、关联、存储过程、规则、缺省值、触发器等的集合和管理者。

Visual FoxPro 3.0 支持功能更全面的面向对象的程序设计技术，对 FoxPro 中原有的一些工具，如屏幕生成器、项目管理器、报表生成器等作了大幅度的调整，增加了许多的功能。此外，Visual FoxPro 3.0 还提供了众多的 Wizard（智能生成器）工具，使得用户能根据系统的指示一步步地完成各项基本操作。

1.2 Visual FoxPro 3.0 的特点

在 Visual FoxPro 3.0 中，面向过程的程序设计和面向对象的程序设计是共存的，这使得通过 Visual FoxPro 3.0 可以建立功能强大的、灵活的应用程序。由于 Visual FoxPro 3.0 本身就是一个 PC 平台上的 DBMS，它的集成环境就具有分类、追踪、处理、存储、打印、传输数据的能力，通过 Visual FoxPro 3.0 的环境界面，就可以顺利完成许多的数据库操作任务。

而在另一方面，Visual FoxPro 3.0 又像其他的程序设计语言一样，它提供了开发大型应

用程序的能力和手段，支持众多的数据类型，提供了诸如表、数组、变量以及其他的一些数据载体；也像其他的高级程序设计语言一样支持各种各样的运算，使得 Visual FoxPro 3.0 也可以成为一个完全独立的高级程序设计语言。

Visual FoxPro 3.0 作为一个数据库管理系统，它又提供了丰富的命令和函数以支持大量数据的录入、存储、传输、保护、处理、显示、打印、管理等，而 Visual FoxPro 3.0 较其他数据库产品更为出色的则是它的查询、过滤或检索数据库与表的功能，由于它采用的 Rushmore 优化技术，使得 Visual FoxPro 3.0 成了检索速度最快的数据库产品。而由于 Visual FoxPro 3.0 引入面向对象的程序设计技术和可视编程技术，使它又向 Visual Basic，Visual C 等这些可视编程语言发起了强劲的挑战，更由于 Visual FoxPro 3.0 本身的数据库管理功能，使得它成了独一无二的可视编程语言。

(1)Visual FoxPro 3.0 是检索速度最快的 PC 平台上的数据库。造成这一结果的原因首先在于 Visual FoxPro 3.0 中继续采用了 Rushmore 技术对 Visual FoxPro 3.0 的命令进行优化查询处理，使得长达数小时数分钟的查询可以降低到数秒钟内完成；其次在于复合索引技术的广泛采用，改变了传统的单一入口的索引文件结构，使得在这个索引文件中可以包含更多的索引，也利于 Rushmore 技术的使用；再次在于 SQL-Select 命令的引入，使得我们能以最少的途径、代码和最快的速度从一个或多个表中检索数据；最后还在于 Visual FoxPro 3.0 能根据系统运行的环境调整自身的配置，最充分地利用环境资源，以期获得最优的性能。

(2)丰富的开发工具又是 Visual FoxPro 3.0 向广大用户、开发人员作出的一大贡献。Visual FoxPro 3.0 不但有一个功能极其强大的集成环境提供给用户，使得用户可以通过菜单、界面、图形浏览工具、对话窗口、Help 系统及嵌入的各种生成器来轻松操作系统中存储的表和数据库外，开发人员还能充分利用功能强的编辑器、设计器、跟踪与调试工具、项目管理器等工具，开发功能齐全的应用程序。而 Form Designer，Report Designer，Menu Designer 等工具更是为开发人员提供了最好的自动工具。

(3)面向对象的可视化编程技术又是 Visual FoxPro 3.0 的杰出贡献。虽然 Visual FoxPro 3.0 仍然支持传统的面向过程的程序设计，但它现在也引入了真正的面向对象的可视编程技术，支持类、子类、对象、继承、封装、多态性等面向对象程序设计的各种特征，更为重要的是这些特征已经与 XBase 语言相融合，成了功能更加强大、更加丰富的一代 PC 平台数据库管理系统。Visual FoxPro 3.0 中，不但可以利用它的类设计工具交互地生成类和子类，也可以完全采用编程的方式来实现，这使得 Visual FoxPro 3.0 更具有灵活性和适应性。

(4)事件驱动又是 Visual FoxPro 3.0 中的一大特点，用户可以生成真正的事件驱动的应用程序，摆脱传统上的 READ 层次嵌套。用户不但可以利用 Visual FoxPro 3.0 提供的标准事件处理代码，而且可以编写自己的事件处理代码来响应全部的标准的 Windows 事件。

(5)工具条的使用使得 Visual FoxPro 3.0 与其他的 Microsoft 产品一样有了直观方便的操作。在工具条上，可以设置一些图形化的按钮来代表一些经常执行的动作或经常使用的对象。此外，Visual FoxPro 3.0 的工具条还可以进行用户化的处理，即允许用户从 Visual FoxPro 3.0 的工具条中选择一些按钮组成用户自己的工具条，而且在应用程序中还可以通过编写代码来定义工具条和按钮。

(6)简单、灵活、多样的数据交换手段又是 Visual FoxPro 3.0 的一大特色。在 Visual FoxPro 3.0 中，与其他应用程序进行数据交换的文件格式是多种多样的，如文本文件、电子

表格、表等。Visual FoxPro 3.0 外部的数据不但可以轻而易举添加到 Visual FoxPro 3.0 的表中，而且，Visual FoxPro 3.0 可以将它的表转换成其他格式的数据文件交付给其他的应用程序。

(7)OLE 的自动化处理又扩展了 Visual FoxPro 3.0 与其他应用程序之间的交流手段和能力，在 Visual FoxPro 3.0 中，用户可以编写其他应用程序中的命令，然后发送给该应用程序，以控制它的运行。例如，我们可以在 Visual FoxPro 3.0 中编写 Excel 的宏命令，将这些宏命令发送给 Excel 后控制它的运行。这样一来，就增强了 Visual FoxPro 3.0 与其他应用程序之间的交互能力。

(8)数据字典或者是数据库(.DBC)文件，不但组织管理 Visual FoxPro 3.0 中的表(.DBF)，而且它变成了一个数据字典，可以存储和管理有关表、记录和字段的规则、缺省值、触发器、视图、表间关联以及表到数据库的连接等。

(9)事务处理又是 Visual FoxPro 3.0 的一个特色。在 Visual FoxPro 3.0 中引入的事务处理技术、数据缓冲技术——悲观的表缓冲、悲观的记录缓冲、乐观的表缓冲和乐观的记录缓冲等大大地简化了多用户环境下开发者对数据更新问题的处理。

(10)对客户/服务器结构的支持又是 Visual FoxPro 3.0 的一大特色，在开发客户/服务器应用程序时可以把 Visual FoxPro 3.0 作为前端，它使用 SQL 直接访问服务器，并综合了对服务器数据的可更新视图的高级技术，所有这些，都为用户提供了一个坚实的基础以开发功能全面的客户/服务器应用程序。系统中提供的数据库、远程视图、空值、事务处理以及对 ODBC 数据源的访问，都支持了客户/服务器结构应用程序的开发。

(11)用户化的帮助系统使得 Visual FoxPro 3.0 更加容易操作，此外，Visual FoxPro 3.0 专业版中提供的 Help 设计工具，可以帮助用户建立 Windows 风格或 .DBF 风格的 Help 系统。

总之，Visual FoxPro 3.0 在 DBMS 之上引入了面向对象的编程技术，继续支持 XBase 语言的规范，同时引入了 Rushmore、SQL 等查询技术，丰富和完善了 FoxPro 的功能。

1.3 Visual FoxPro 3.0 编程入门

尽管 Visual FoxPro 3.0 是一个功能强大的数据管理工具，然而 Visual FoxPro 3.0 所提供的全面的编程能力，使我们不但可以使用传统的面向过程的方法编写用户自己的应用程序，而且更可以使用 Visual FoxPro 3.0 所提供的面向对象编程技术和事件驱动编程技术以大幅度提高应用程序的生产率。

通常，在 Visual FoxPro 3.0 中，任何可以用程序完成的工作，我们都可以用 Visual FoxPro 3.0 的交互环境来自己控制实现。例如，我们希望查询 Customer 表中的一个顾客信息，可以按下述的步骤利用 Visual FoxPro 3.0 交互环境来实现。

1. 从 File 菜单中选择 Open 选项。

2. 从文件类型列表框(List Files of Type)中选择 Table 类型。

3. 从列出的文件列表中选取 CUSTOMER.DBF，然后用鼠标双击这个文件。

4. 打开 Customer 表后，从 View 菜单中选择 Browse 选项，开始浏览 Customer 中的顾客信息。

5. 滚动整个表，寻找自己希望查阅的顾客信息，然后选择该记录的相应字段，例如，寻找 Company 字段值等于"Evnst Handel"的记录。

完成相同的工作，我们可以用 Visual FoxPro 的命令/程序来实现，如：

```
USE CUSTOMER
BROWSE
LOCATE FOR Company="Evnst Handel"
```

上面的简单命令，我们可以在 Visual FoxPro 3.0 的 Command 窗口中作为三条命令依次键入就可以完成指定的工作。很显然，在 Customer 表中顾客数很多时，使用程序或命令将获得极大的效率，而唯一麻烦的只是需要编写程序。

上面只是一个十分简单的例子，一些用户也许会觉得用手工操作的方式更简单和直观，更能充分体现交互环境的功效，然而这仅仅是对熟练的数据库用户而言是简单的，对于大多数的最终用户，他的工作并不会如此简单，需要计算机管理的数据也更多、更为复杂，而且数据的处理也具有一定的规程，指望他每次自己来依据规程作大量的固定工作是很不合适的。因此需要这些处理编写成程序，使得用户简单启动程序运行之后就能获得预期的结果，而且这也有助于发挥计算机快速、自动的特性。

所谓的程序，是一系列 Visual FoxPro 3.0 可以理解的命令、函数和操作的集合，它们是可以由 Visual FoxPro 3.0 解释执行的代码。在 Visual FoxPro 3.0 中，我们不但可以在 Command 窗口中执行 Visual FoxPro 3.0 的命令和函数，而且可以编写成程序文件的形式来执行。此外，Visual FoxPro 3.0 中的一些设计工具如 Form Designer，Menu Designer，Report Designer 等设计器所产生的事件、方法或者过程的代码，也是可以由 Visual FoxPro 3.0 执行的。

1.3.1 Visual FoxPro 3.0 的程序代码编写工具

1.3.1.1 Command 窗口

在 Visual FoxPro 3.0 中，提供了一个 Command 窗口，在这个窗口中，我们可以键入一系列的 Visual FoxPro 3.0 命令进行执行。当希望重新执行某个命令或者修改以前的某个命令后再执行时，可以将光标移回到包含该命令的那个行上，然后按 ENTER 键，或者修改后再按 ENTER 键。

由于 Command 窗口是一个编辑窗口，因此，可以用 Visual FoxPro 3.0 中的各种编辑工具对命令进行编辑处理，即可以在 Command 窗口中进行修改、插入、删除、剪切、拷贝、粘贴等操作。

此外，Visual FoxPro 3.0 中，当我们通过菜单或者对话框进行选择和操作时，将在 Command 窗口显示一条与我们的选择和操作相对应的命令。这样，在 Command 窗口中就出现了我们进入 Visual FoxPro 3.0 之后交互执行的所有命令。正是由于 Visual FoxPro 3.0 具有一个编辑窗口的特性，可以进行拷贝和粘贴，因此，我们可以把 Command 窗口中执行过的命令任意拷贝或者粘贴成一个 Visual FoxPro 3.0 程序，然后再重复执行这个程序，这样一来，对于那些成百上千条命令所组成的程序，一遍又一遍的反复执行就变得很容易了。

例如，我们前面介绍的简单例子，就可以在 Command 窗口中依次键入三个命令来执行：

```
USE Customer
LOCATE FOR Company="Ernst Handel"
BROWSE
```

1.3.1.2 创建 Visual FoxPro 3.0 程序文件

Visual FoxPro 3.0 程序是一个 ASCII 文本文件，其中包含了一系列的命令。一般，程序文件的文件扩展名是 .PRG。

1. 创建程序文件

由于 Visual FoxPro 3.0 的程序文件是一个文本文件，因此可以使用任何的文本编辑窗口来创建和编辑 Visual FoxPro 3.0 程序文件，这些文本编辑窗口既可以是 Visual FoxPro 3.0 提供的标准编辑器，也可以是其他的一些文本编辑软件。在此，我们主要介绍一下 Visual FoxPro 3.0 中创建一个程序文件的方法：

(1)从 File 菜单中选择 New 选项，建立一个新文件；

(2)在 New 对话框中，选择 Program，表示建立一个程序文件；

(3)选择 New File，开始新程序文件的编写。

上面介绍的是通过菜单选择来建立一个新的程序文件，而在 Command 窗口中，还可以使用命令 MODIFY COMMAND 来建立新的程序文件，这时，Visual FoxPro 3.0 将打开一个名为 Program 1 的窗口，并在此窗口中接收输入的程序。

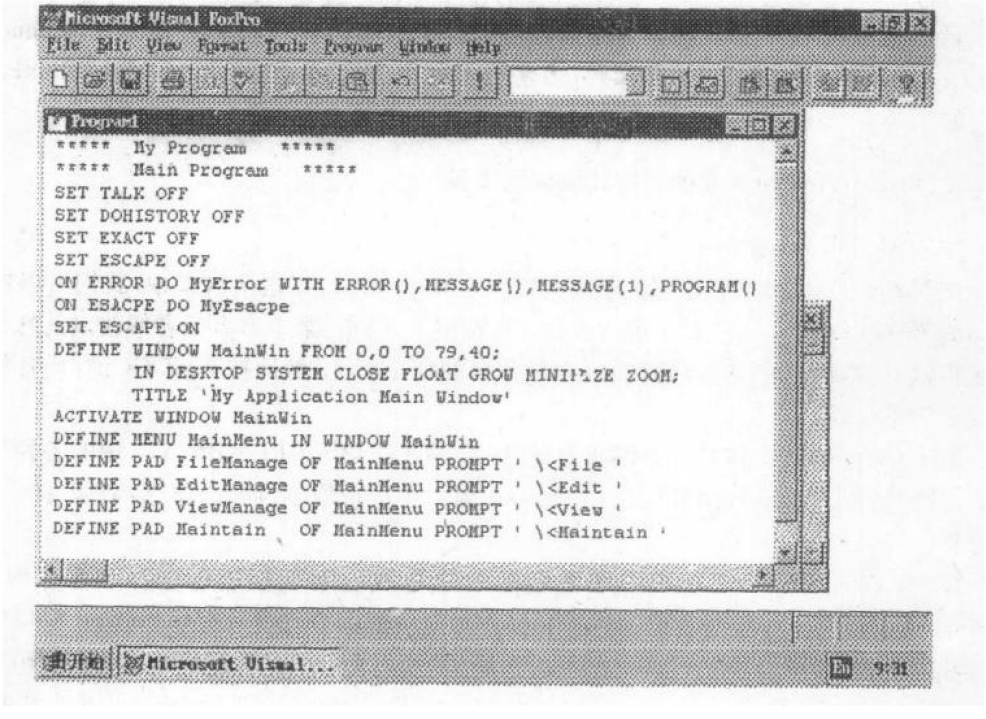

图 1.1 创建 Visual FoxPro 3.0 程序文件

2. 存储程序文件

程序输入完成后必须进行存储，这时最简单的办法是从系统的 File 菜单中选择 Save 选项将新建的程序文件存入磁盘中。程序一旦存储在磁盘上之后，我们就可以很方便地修改和

执行这些程序文件。

3. 修改程序文件

当一个文件已经建立后，如果我们希望改变其中的某些内容时，就可以打开这个文件进行修改。为了打开一个已经存在的程序文件，可以从 File 菜单中选择 Open 选项，然后出现一个文件列表框，并从文件类型列表框(List File of Type)中选择 Program 文件类型，最后再从文件名列表中选取要修改的程序后按 Open 按钮。

上面是菜单系统中文件的打开方法，而在 Command 窗口中，却可以使用如下的命令打开一个指定的程序文件：

```
MODIFY COMMAND myprogram
```

或者使用命令：

```
MODIFY COMMAND ?
```

与程序的输入一样，当完成程序的修改之后也应当保证将文件存入磁盘中。

4. 运行程序文件

一个程序创建之后，它就可以运行了。为了运行一个程序，可以有两种方法，其一是从 Program 菜单中选择 Do 选项，然后从列出的程序文件中选择要运行的文件。其二是在 Command 窗口中键入 Do〈程序名〉命令。

1.3.1.3 利用设计工具创建程序代码

在 Visual FoxPro 3.0 中，Form Designer 和 Menu Designer 可以创建有关用户界面部分的完整程序代码，能够适应用户界面部分的各种要求。而 Report Designer 工具则用于创建复杂的报表，并将完整的程序代码存入报告文件中。

为了充分发挥 Visual FoxPro 3.0 的强大功能，就需要这些强有力的工具。有关 Form Designer，Menu Designer 和 Report Designer 设置工具的详细信息，我们将在本书的后面加以介绍。

1.3.2 数据类型

Visual FoxPro 3.0 是开发应用程序的强有力工具，能很好地实现信息的分类、追踪和处理，尽管通过 Visual FoxPro 3.0 的界面，我们已经能够操作许多的数据库，完成许多的数据处理任务，然而，还应当了解 Visual FoxPro 3.0 提供的语言能力，这样，就能编制出威力更大的 Visual FoxPro 3.0 应用程序。

与其他的程序设计语言一样，Visual FoxPro 3.0 提供了多种数据类型，并可以将数据存入各种类型的表、数组、变量和其他存储容器中。

Visual FoxPro 3.0 的数据类型分为两大类，一类是适应用变量和数组的，另一类则适应用表中字段的。

1.3.2.1 字符类型

字符(Character)数据类型是由字母、数字、空格、符号和标点等组成的，字符型的字段、内存变量、数组元素等存储的是诸如名、地址、提示信息以及不用于算术运算的数字等形式的文本信息。

Chanacter 类型字段或者内存变量的长度介于 1 到 254 字节之间，并且每个字符占用一个字节。

1.3.2.2 货币类型

货币(Currency)数据类型用于代替数据类型(Numeric)的货币值,如果在Currency类型字段或者变量中指定的值小数位数超过四位,那么Visual FoxPro 3.0在计算之前将对这个货币值进行舍入处理,因为Visual FoxPro 3.0中的Currency类型只允许最多有四位小数。

Currency类型字段和变量的取值范围介于-922337203685477.5808到922337203685477.5807之间,并且它占用8个字节的存储空间。

例如,下面的两个语句就定义了两个货币类型的变量,并且其中的一个还进行了舍入处理:

```
Money=$100.35
Moremoney=$645.72356
```

1.3.2.3 日期类型

日期(Date)数据类型是用于存储有关年、月和日的时间序列的一种数据类型,一个日期类型的字符存储格式为"yyyymmdd",其yyyy表示年号,占用4个字节,mm表示月份,占用2个字节,dd表示日子,占用2个字节。

Date类型字段和变量中表达日期的日期格式有许多种,最常用的格式为mm/dd/yyyy,如:

```
mdate={10/14/94}
```

对于日期类型常量,必须用花括号({和})将一个日期串封闭起来,而对于空值的日期常量,则可以用{}或者{ }或者{/}表示。

注意,SET DATE、SET MARK、SET CENTURY命令的设置值将影响日期的格式。

Date类型占用8个字节,取值介于{1/1/100}到{12/31/9999}之间。

1.3.2.4 日期时间类型

日期时间(DateTime)数据类型用于存储日期和时间值,DateTime类型字段或者变量的字符存储格式为"yyyymmddhhmmss",其中yyyy表示日期中的年号,mm(第一个)表示月份,dd表示日子,hh表示时间中的小时,占用2个字节,mm(第二个)表示时间中的分钟,占用2个字节,ss表示时间中的秒,占用2个字节。DateTime类型的值中既可以只包含一个日期或者只包含一个时间值,但也可以同时包含一个日期和一个时间值,因为存储缺省日期时,Visual FoxPro 3.0给它自动加上1989年12月30日这个日期,如果省略时间,则Visual FoxPro 3.0则自动加上午夜零点这个时间。如:

```
mdatetime={4/1/95 10:00 am}
mtimeonly={10:00 am}
```

注意,对于DateTime类型的常量,与Date类型的常量一样,需要用一对花括号({})来加以说明。而描述一个空的DateTime类型值,则需要用{:}来表示。同时,DateTime类型中的日期部分,与Date类型完全一样,具有多种的显示格式,并受SET DATE、SET MARK、SET CENTURY命令设置值的影响,而时间部分的显示格式则受SET HOURS和SET SECONDS命令设置的控制。

DateTime类型中的日期部分的取值介于{1/1/100}到{12/31/9999}之间,时间部分的取值介于00:00:00 a.m.到11:59:59 p.m.之间。

1.3.2.5 双精度类型

双精度(Double)数据类型用于取代 Numeric 类型,以便能提供更高的数值精度。Double 数值类型只用于表中字段的定义,它采用固定存储长度的浮点数形式。

Double 类型占用 8 个字节,取值范围介于+/-4.94065645841247E-324 到+/-1.79769313486232E308 之间。

Double 类型不同于 Numeric 类型,它的小数点位置是由输入的数据值来决定的。

1.3.2.6 浮点类型

浮点(Float)数据类型与 Numeric 类型是完全等价的,Float 类型主要用于 Visual FoxPro 3.0 的兼容处理。

1.3.2.7 通用类型

通用(General)数据类型用于存储 OLE 对象,在这个 General 字段中,它包含了对 OLE 对象的引用,而一个 OLE 对象的具体内容则可以是一个电子表格、一个字处理器的文档、图片等,这些 OLE 对象是其他的应用软件建立的。

General 字段在表中的长度为 10 字节,而 OLE 对象的实际内容、类型和数据量则取决于建立该 OLE 对象的服务器以及是连接或者嵌入 OLE 对象。如果采用连接 OLE 对象方式,那么表中只包括对 OLE 对象中数据的一个引用说明,以及到创建该 OLE 对象的应用软件之间的引用说明。如果是采用嵌入 OLE 对象方式,那么,表中在包含到创建该 OLE 对象的应用软件之间的引用之外,还包括了 OLE 对象中的实际数据。这时,General 字段的长度仅受限于内存的可用空间。

1.3.2.8 整数类型

整数(Integer)数据类型用于无小数部分数值的存取,在表中,Integer 类型的字段占用 4 个字节,而且是用二进制形式表示的,因此,它比 Numeric 类型的字段占用的空间要少得多,而且 Integer 类型的值不需要转换成 ASCII 字符来存储。

注意,Numeric 类型是需要进行转换的。

Integer 类型占用 4 个字节,取值介于-2147483647 到 2147483646 之间,而且 Integer 类型仅用于表中字段类型说明。

1.3.2.9 逻辑类型

逻辑(Logical)数据类型用于存储只有两个值的数据,它是一种高效的存储方法,存入的值为真(.T.)和假(.F.)两种状态,Logical 类型可以用于表中字段和内存变量及数组元素的数据类型说明中。

1.3.2.10 备注类型

Memo(备注)字段类型只用于表中,它用于数据块的存储。在表中,Memo 字段只包含 10 个字节,并用这 10 个字节来引用备注的实际内容。而实际的备注内容的多少只受内存可用空间的限制,但它们是以块的方式存放的。

由于备注字段的实际内容变化很大,不能直接将备注内容存储在表.DBF 文件中,因此需要一个相对独立的文件来存储这些备注内容,这个文件的扩展名是.DBT。

Visual FoxPro 3.0 的备注可以包含任意的数据,只要适用于字符串中的所有内容,都可以写入备注中。由于 Memo 类型只用于表中,因此内存中没有 Memo 类型的变量和数组元素。为了能够实现对备注内容的处理,需要将备注字段的内容转换成驻留内存的字符串,然

后使用所有的字符函数进行处理。

1.3.2.11 数值类型

数值(Numeric)数据类型用于表示数量的一种数据类型，它是由数字0到9以及一个符号(+和-)和一个小数点(.)组成的。

Numeric类型既可以用于表中对字段进行定义，也可以用于内存变量和数组元素。在表中，Numeric类型的长度介于1到20字节之间，在内存中，Numeric类型则占用8字节。这种数据类型的值介于-.9999999999E-19到+.9999999999E+20之间。

在Numeric类型字段中，小数点的位置和字段的长度由用户创建该字段时指定。注意，字段的宽度中将包含小数点和小数位数在内。例如，定义一个6字节长的数值字段，其中小数部分占4位，那么，该字段可存储的最大值为9.9999。

在FoxPro中，Numeric类型数据是转换成ASCII字符来存储的。

1.3.3 数据存储

数据在系统中进行加工处理时，需要一个数据载体，用于暂时存储内存中的数据，我们将用于存储数据的变量、数组、字段、对象特性、常量等都称为数据存储容器，它们决定了数据的类型及使用方法。而对数据存储容器的声明方法和位置又决定了该数据存储容器的可用范围。

正如我们了解到的那样，多数的程序设计语言都将数据存入常量、变量和数组中，而在Visual FoxPro 3.0中，数据还可以存入记录和对象之中。

1.3.3.1 常量

一个常量是一个命名的数据项，在所有的操作期间，它的值都将保持不变，例如，π或者3.1415926535就是一个数值常量，而字母A就是一个字符常量。

对于数值数据类型，任何的数字串，如2.159、3.1415926535等都是数值常量；对于字符型，任何字符串，如"Visual FoxPro"等都是一个字符常量；对于逻辑类型，其常量只有两个，即真(.T.)和假(.F.)。而日期类型，其常量是用花括号对({和})来定义的，如{4/4/94}，而日期时间类型，其常量也是用花括号({和})定义，如{4/1/95 10:00:00 a.m.}。

此外，在Visual FoxPro 3.0中还可以创建一种特殊的常量——编译时常量，这种常量只存在于Visual FoxPro 3.0应用程序的编译之时，编译之后，将用常量的具体内容置换该常量在源代码中的位置，而且常量中可以包含任意类型的数据。

Visual FoxPro 3.0中的编译时常量是需要定义的，定义常量的内容、数据类型和作用范围。在Visual FoxPro 3.0中，编译时常量用伪编译指令(指示符)#DEFINE进行定义，如：

```
#DEFINE TABLERR1 "This table is not available"
```

定义了TABLERR1这个常量后，在源代码中就可以使用TABLERR1，而在编译之后，凡是出现TABLERR1的地方都用字符串"This table is not availble"取代。

注意，#DEFINE定义的编译时常量，必须用另一个伪编译指令#UNDEF取消它的定义，如：

```
#UNDEF TABLERR1
```

否则，编译时常量将一直有效。

1.3.3.2 变量

一个变量是内存中存储一个数据的位置名称，在这个存储位置中存放的数据在操作期间是可以通过这个名称来读和写的。

Visual FoxPro 3.0 中的变量称为内存变量，它有两种定义方式，一种是使用 STORE 赋值命令，如 STORE 7 TO nVar，另一种是使用等号(=)赋值操作，如 nVar=7。当然，在 Visual FoxPro 3.0 中，有一些命令和函数也可以在其执行期间创建一些工作用内存变量或数组。

Visual FoxPro 3.0 中的内存变量是不需要特别声明的，当使用 STORE 命令和=操作进行赋值的同时，完成变量的定义，并且确定该变量的内容和数据类型。

每个内存变量都有一定的作用域，在一般情况下，一个变量的作用域包括定义它的程序以及该程序所调用的子程序范围。然而，Visual FoxPro 3.0 中，还可以使用 LOCAL，PRIVATE 和 PUBLIC 命令强制规定变量的作用范围。

对于 LOCAL 创建的内存变量和数组，只能用于创建它们的那个程序，并由该程序进行存取，而较高或者较低层上的程序都不能存取 LOCAL 定义的本地变量和数组。一旦定义 LOCAL 变量和数组的程序执行完毕，这些变量和数组将自动释放。

PRIVATE 用于定义私有变量和数组，它用于定义当前程序的内存变量和数组，并将以前程序定义的同名变量和数组保存起来，并且在当前程序及其以下调用的子程序中都使用 PRIVATE 定义的变量和数组，一旦定义 PRIVATE 变量和数组的程序执行完毕，返回到上层程序中时，将恢复那些被保存的同名变量和数组，恢复它们原来的值和类型。实际上，PRIVATE 命令起到隐藏和屏蔽上层程序的变量和数组的作用。

PUBLIC 则用于定义全局变量和数组，用 PUBLIC 定义的全局变量，则本次 Visual FoxPro 3.0 运行期间，所有的程序都是可以存取这些全局变量和数组的。此外，在 Command 窗口中建立的变量和数组，将自动默认为全局变量。

对于存储在变量或数组中的数据，可以通过变量名和数组元素名来进行访问，例如，下面的三个命令，都将显示同样的内容：

```
? m.cFname
? m->cFname
? cFname
```

其中 m.cFname 和 m->cFname 用来与字段名相区别，当字段名与内存变量名相同时，同样的名将优先用来存取字段，因此将屏蔽同名的内存变量，为了明确指定访问的内存变量，需要使用 m. 或者 m->来强制说明。

1.3.3.3 数组

数组是一组有序的数据值的集合，其中的每个数据值称作数组元素，每个数组元素在数组中的位置是固定的，可以通过一个称作下标的编号来进行访问。数组提供了内存中快速和简便的数据存取方式，而且可以很容易地指定、定位和操作数组的每个元素。尤其在 Visual FoxPro 3.0 中，数组的应用大大提高了数据库和表中记录操作的灵活性。

数组是一种特殊的内存变量，它与内存变量一样存在一个作用域的问题，与变量不同的是，Visual FoxPro 3.0 的数组中可以存放任何类型的数据，而且同一个数组的元素之间存放的数据类型可以是不同的，这一点与许多的高级程序设计语言中的数组是不同的。