

第一部分



VFP 项目开发基础

- 第 1 章 Visual FoxPro 应用程序开发概述
- 第 2 章 Visual FoxPro 高级应用
- 第 3 章 Visual FoxPro 界面设计

Visual FoxPro 应用程序开发概述

本章导读

Visual FoxPro(简称 VFP)是一个功能强大的交互式数据管理工具,而且可以创建应用程序来充分发挥其完善的功能。本章将介绍开发应用程序项目的大致过程以及需要注意的事项。在开发应用程序时,可以利用“项目管理器”将应用程序的各个部分组织起来,用集成化的方法建立应用程序并进行测试。

本章将学习以下内容:

- 项目设计基础知识;
- 使用 VFP 进行项目设计。

1.1 应用程序设计概述

VFP 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,VFP 程序设计涉及到对存储数据的操作。

VFP 是一个功能强大的交互式数据管理工具,而且可以创建应用程序来充分发挥其完善的功能。掌握 VFP 的面向对象程序设计技术以及事件驱动模型,可以最大限度地提高程序设计的效率。

1.1.1 应用程序概述

应用程序通常被看成是一个项目,它是文件、数据、文档以及 VFP 对象的集合,也就是我们常说的“软件”。它主要用来完成对一定事件任务的自动化、规程化操作,将数据流和操作流统一起来。使用 VFP 将能够很好地进行程序设计,VFP 提供了出色的程序设计语言和可视化程序设计环境,我们可以通过使用如“项目管理器”等来提高程序设计的速度和质量。

1.1.2 应用程序开发过程

应用程序开发的基本过程如下:

- ① 对问题进行说明。
- ② 分解问题。
- ③ 编制各模块。
- ④ 测试并完善各模块。
- ⑤ 组装全部模块。
- ⑥ 整体测试。

开始程序设计之前,请注意以下几个问题:

(1) 在解决问题之前,必须把问题说明清楚,否则会不断进行修改,丢弃已编好的代码并从头再来,而且最终也不可能得到满意的结果。

(2) 将问题分解成可单独处理的几个步骤,而不是一下子解决全部问题。

(3) 在开发过程中不断测试和调试已编好的代码。通过测试,检查代码是否能实现所需的功能;调试并找出代码在哪里出错并纠正这些错误。

(4) 精炼数据和数据存储方式,便于程序对其进行处理。这需要正确构造表格的结构。

1.2 VFP 应用程序开发过程

使用 VFP 开发应用程序,首先应该做好对项目的需求分析,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。当然,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,是否要使用特殊的处理方法和使用时应注意的事项,以及是单用户还是网络上的多用户等。

1.2.1 VFP 的应用程序框架

一个典型的数据库应用程序由数据结构、用户界面、查询选项和报表等组成。在设计应用程序时,应仔细考虑每个组件将提供的功能以及与其他组件之间的关系。

一个经过良好组织的 VFP 应用程序一般需要为用户提供菜单;提供一个或多个表单供数据输入并显示。同时还需要添加某些事件响应代码,提供特定的功能,保证数据的完整性和安全性。此外,还需要提供查询和报表,允许用户从数据库中选取信息。

在建立应用程序时,需要考虑如下的任务:

1. 设置起始点

将各个组件链接在一起,然后使用主程序(又称主文件)为应用程序设置一个起始点。主程序作为应用程序执行的起始点,可以包含一个程序或者表单。当用户运行应用程序时,VFP 将为应用程序启动主程序,然后主程序再依次调用所需要的应用程序其他组件。所有应用程序必须包含一个主程序。一般来讲,最好的方法是为应用程序建立一个主程

序。使用一个表单作为主程序,可以将主程序的功能和初始的用户界面集成在一起。

可以从“项目”菜单中选择“设置主文件”命令或者用鼠标右键单击选择“设置主文件”命令。注意应用程序的主文件自动设置为“包含”。这样,在编译完应用程序之后,该文件作为只读文件处理。项目中仅有一个文件可以设置为主文件。设置完毕后文件名称以粗体形式显示。

如果在应用程序中使用一个程序文件(*.prg)作为主程序,必须保证该程序中包含一些必要的命令,这些命令可控制与应用程序的主要任务相关的任务。在主程序中,没有必要直接包含执行所有任务的命令。常用的一些方法是调用过程或者函数来控制某些任务,例如环境初始化和清除等。

主程序所包含的代码如表 1-1 所示。

表 1-1 主程序中的代码功能

代 码	功 能
DO SETUP. PRG	调用程序建立环境设置(在公有变量中保存值)
DO MAINMENU. MPR	将菜单作为初始的用户界面显示
READ EVENTS	建立事件循环。另外一个程序(如 Mainmenu. mpr)必须可执行一个 CLEAR EVENTS 命令,否则会出现无法退出 VFP 错误
DO CLEANUP. PRG	在退出之前,恢复环境设置

当然也可以把 4 个程序功能的代码放入到一个文件中,以一个独立文件形式出现。

2. 初始化环境

主文件或者主应用程序对象必须做的第一件事情就是对应用程序的环境进行初始化。在打开 VFP 时,默认的 VFP 开发环境将建立 SET 命令和系统变量的值。但是,对应用程序来说,这些值并非最合适。

对于应用程序来说,初始化环境的理想方法是将初始的环境设置保存起来,在启动代码中为程序建立特定的环境设置。

在一个应用程序特定的环境下,可能需要使用代码执行以下操作:

- 初始化变量。
- 建立一个默认的路径。
- 打开任一需要的数据库、自由表及索引。如果应用程序需要访问远程数据,则初始的例程序也可以提示用户,由用户提供所需的注册信息。
- 添加外部库和过程文件。

3. 显示初始的用户界面

初始的用户界面可以是菜单,也可以是一个表单或其他的用户组件。通常,在显示已打开的菜单或表单之前,应用程序会出现一个启动屏幕(闪屏)或注册对话框。在主程序中,可以使用 DO 命令运行一个菜单,或者使用 DO FORM 命令运行一个表单以初始化用户界面。

4. 控制事件循环

应用程序的环境建立之后,将显示出初始的用户界面,这时,需要建立一个事件循环来等待用户的交互动作。我们可以通过执行 READ EVENTS 命令,该命令使 VFP 开始处理例如鼠标单击、按键等用户事件。

从执行 READ EVENTS 命令开始,到相应的 CLEAR EVENTS 命令执行期间,由于主文件中所有的处理过程全部挂起,因此将 READ EVENTS 命令正确地放在主文件中十分重要。例如,在一个初始过程中,可以将 READ EVENTS 作为最后一个命令,在初始化环境并显示了用户界面后执行。如果在初始过程中没有 READ EVENTS 命令,应用程序运行后将返回到操作系统中。

在启动了事件循环之后,应用程序将处在所有最后显示的用户界面元素控制之下。例如,如果在主文件中执行下面的两个命令,应用程序将显示表单 Startup.scx。

例如,如果要测试 SET TALK 命令的默认值,同时保存该值,并将应用程序的 TALK 设为 OFF,可以在启动过程中包含如下的代码:

```
IF SET('TALK')="ON"
    SET TALK OFF
    cTalkVal="ON"
ELSE
    cTalkVal="OFF"
ENDIF
```

如果要在应用程序退出时恢复默认的设置值,一个好的方法是把这些值保存在公有变量、用户自定义类或者应用程序对象的属性中。

```
SET TALK & cTalkVal
DO FORM STARTUP.SCX
READ EVENTS
```

如果在主程序中没有包含 READ EVENTS 或等价的命令,在开发环境的“命令”窗口中,可以正确地运行应用程序。但是,如果要在菜单或者主屏幕中运行应用程序,程序将一闪即逝。

应用程序必须使用 CLEAR EVENTS 命令结束事件循环。

5. 退出应用程序

退出应用程序时,恢复原始的开发环境。

1.2.2 VFP 开发过程

在计划好应用程序中所需组件后,可能会希望建立一个目录框架和项目,以组织那些为应用程序而建立的组件文件。可在“Windows 资源管理器”中建立框架,在“项目管理器”中创建项目,或使用“应用程序向导”同时建立。这种新增的“应用程序向导”将打开

“应用程序生成器”，可以进一步组织那些最初用向导建立的项目和组件。在对项目进行分析后，就可以根据需要进行项目开发准备和实际项目开发。由于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，在具体设计过程中，可以按照以下步骤进行(可以根据实际情况删减或更改)。

- ① 规划应用程序。
- ② 创建数据库。
- ③ 创建类。
- ④ 设计用户操作界面。
- ⑤ 设计访问信息的方法。
- ⑥ 测试和调试。

下面，介绍具体的 VFP 开发步骤，用户可以根据实际情况自己安排。

1. 建立应用程序的目录结构

VFP 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以及用户界面(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。即使一个很小的应用程序，也会涉及到多种类型的文件，如 DBC 数据库、DBF 表以及菜单、表单、报表、位图等。如果把这些文件都放在一个目录下，将会给以后的修改、维护工作带来很大的不便。因此，我们需要建立一个层次清晰的目录结构，让不同类型的文件存放在不同的文件夹中，这样使用方便、清晰，易于管理，有时候会带来意想不到的效果。具体的目录结构可以参考图 1-1 所示的人事管理系统目录结构。也可以根据自己的需要和习惯删减或更改，例如建立音频文件夹等。按照项目安排，通常各个子文件夹用来存放如表 1-2 所示的内容。

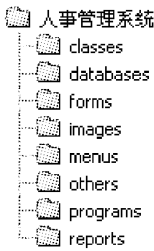


图 1-1 应用程序文件夹示例

表 1-2 应用程序文件夹存放内容示例

文 件 夹	存 储 内 容
classes	自己设计的类或者引用的第三方的类
databases	存放数据库、数据表、视图等
forms	存放表单
images	存放背景图片、图标等
menus	存放菜单文件
others	存放其他需要的文件，如配置文件等
programs	存放程序文件
reports	存放报表、标签等文件

有了表 1-2 那样的存储结构，就可以设计具体的应用程序。

2. 使用项目管理器

VFP 为开发人员提供了一个优秀的工作平台“项目管理器”。我们可使用“项目管理器”组织和管理项目中的文件。项目是文件、数据、文档以及 VFP 对象的集合,项目文件以 .pjx 扩展名保存。当激活“项目管理器”窗口时,VFP 在菜单栏中显示“项目”菜单,在这里能看到组成应用程序的元素,如图 1-2 所示。



图 1-2 “项目管理器”窗口

在“项目管理器”中,以类似于大纲的形式组织各项,可以展开或折叠它们。在项目一中,如果某类型数据项有一个或多个数据项,则在其标志前有一个加号。单击标志前的加号可查看此项的列表,单击减号可折叠展开的列表,符号⊖表示此项已从项目中排除。如果希望能够把程序做成独立性非常强的软件,应该尽可能把所需要的东西包括进来。但是如果数据库等包括进来后,就不可以进行更改了;否则,更改时就会出现错误,设计的时候应该特别关注这个问题。

与工具栏类似,可以将“项目管理器”拖动到屏幕顶部或双击标题栏,从而停放“项目管理器”。“项目管理器”停放后,被自动折叠,只显示选项卡。

位于项目管理屏幕右侧的控制是用于管理项目的组成部分。

- 新建: 创建一个新部件。
- 添加: 选择一个已经存在的部件以包含到项目中。
- 修改: 编辑高亮显示的部件,这与在一个项目元素上双击效果相同。
- 运行: 运行高亮显示的部件。
- 移去: 从项目中移去高亮显示的部件。
- 连编: 建立一个应用程序。

项目管理器建立项目后,在 VFP 的系统菜单中多了一个新选项“项目”。选择菜单“项目/项目信息”,打开“项目信息”对话框。在此可以输入以下类型的信息。

- 开发者的信息(如姓名、地址等)。
- 项目的主目录。这对于项目的连编和生成等有莫大的好处。
- 在应用程序文件中是否包含调试信息。这会略微增加程序的大小,但是对程序的

调试有很大的帮助。

- 是否对应用程序进行加密。如果应用程序加密了,要想对这样的应用程序进行逆向工程以重组源代码对任何人都是非常困难的,当然如果希望对数据库和数据表进行响应的操作,恐怕需要做的工作更多一些,这也是 VFP 一个弱点所在。
- 当应用程序处于最小化的时候使用什么图标(可以更改系统提供的默认图标,设计个性化程序)。

具体内容如图 1-3 所示。以上的信息填写对项目的版权,项目的个性化等有很多好处。

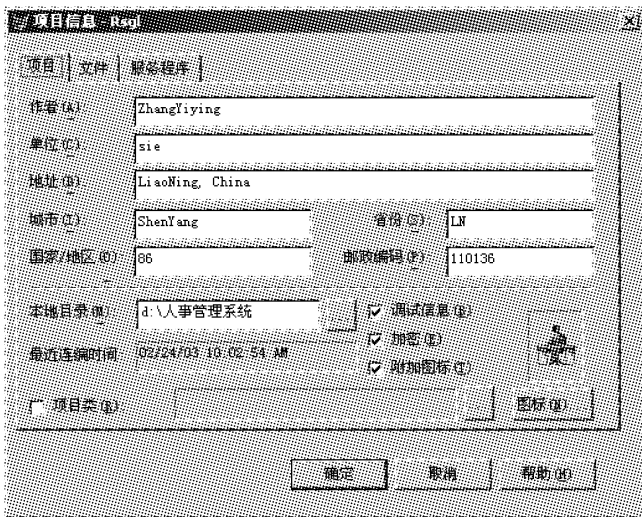


图 1-3 “项目信息”窗口

3. 增加项目元素

在 VFP 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以及用户界面(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。可以通过项目管理器把这些元素依次创建或添加到项目中去。

- ① 创建或添加数据库(包括数据表)到项目中。
- ② 建立或添加表单(根据应用程序要求,添加相应的功能窗口)。
- ③ 建立或添加菜单。
- ④ 加入主程序(也可以依照实际需要制作一些功能程序,如函数,API 引入等)。
- ⑤ 测试与连编。

以上各个元素可根据实际需要随时在项目管理器中创建或添加,没有固定的先后顺序。在这个过程中,还要进行如界面设计、环境设计(如配置网络)等,后面的章节将进行专题介绍或举例。

4. 设置配置文件 Config.fpw

程序打包后,安装运行时无法脱离 VFP 主窗口环境。这时可以编辑一个 Config.fpw 系统配置文件,对生成的程序进行运行环境等各方面的设定配置文件。

下面介绍添加配置文件 Config.fpw 的方法。选择项目管理器的“其他”页面内的“文本文件”,单击“新建”,创建文本文件。编写如下代码:

```
SYSMENU=OFF      && 关闭系统菜单  
SCREEN=OFF       && 关闭 VFP 主窗口
```

然后单击菜单“文件”,选择“另存”,在弹出的保存窗口中“保存类型”选择 All File (*. *),在文件名中输入 Config.fpw 即可。

1.3 设计数据库

数据是应用程序的操作对象,程序说到底可以看成对数据流的控制。因此若要创建高效的应用程序,必须首先分析数据需求,确定建立数据库的目的,然后设计相应的数据库、表和索引,以满足这些需求。

在 VFP 中,可以使用数据库组织和建立表和视图间的关系。数据库不但提供了存储数据的结构,而且还有很多其他的好处。在使用数据库时,可以在表一级进行功能扩展,例如创建字段级规则和记录级规则、设置默认字段值和触发器等,还可以创建存储过程和表之间的永久关系。此外,使用数据库还能访问远程数据源,并可创建本地表和远程表的视图。

注意:“数据库”和“表”不是同义词。“数据库”(。dbc 文件)指的是关联的数据库,它是一个或多个表(。dbf 文件)或视图信息的容器。通过将信息拆分入表,来增加数据库的灵活性。

1.3.1 设计数据库的步骤

1. 确定建立数据库的目的

用于确定应用程序需要保存的信息,一般情况下,不同类别的数据集合需要存放在不同等的数据库中。

2. 确定需要的表

在明确了建立数据库的目的之后,依据数据的类别不同,如人事方面的、工资方面的等,就可以着手把信息分成各个独立的主题,例如 Employee(职员表)或 Pay(工资表)等。每个主题都可以是数据库中的一个表。

3. 确定所需字段

确定在每个表中要保存哪些信息。在表中,每类信息称作一个字段(属性),浏览表时

在表中显示为一列。例如,在 Employee 表中,可以有这样的字段: Name(姓名)和 Workday(工作时间)等。

4. 确定关系

分析每个表,确定一个表中的数据和其他表中的数据有何关系。由于 VFP 是关系型数据库,因此在具体使用中可以设计良好的关系来进行数据之间的连动。必要时,可在表中加入字段或创建一个新表来明确关系。关系是程序中实现某些功能(如查询等)的好帮手。

5. 设计求精

对设计进一步分析,查找其中的错误。创建表,在表中加入几个示例数据记录,看能否从表中得到想要的结果。需要时可调整设计。

当然,在最初的设计中,不要担心发生错误或遗漏东西。这只是一个初步方案,可在以后对设计方案进一步完善。在完成初步设计后,如果需要,很容易在创建数据库时对原设计方案进行修改。但是需要注意的是,在数据库中输入了数据或连编表单和报表之后,再要修改这些表就困难得多。因此,在连编应用程序之前,应确保设计方案已经考虑得比较全面。

1.3.2 分析数据需求

这是进行数据库设计的第一步,就是要明确数据库的目的、数据格式特征和数据流程等。也就是说需要从数据库中得到哪些信息。明确目的之后,就可以确定需要保存哪些主题的信息(表),以及每个主题需要保存哪些信息(表中的字段)。

请务必和数据库的使用人员(很多是主管人员和实际操作人员)多交换意见,推敲那些需要数据库回答的问题,勾划出要生成的报表,收集当前用来记录数据的表单。所有这些信息在后面的设计步骤中都要用到,当然整个过程还要来回反复几次才能成功。

具体办法可以按照如下方法或原则去做。

(1) 将需求分类放入表:确定数据库中的表是数据库设计过程中技巧性最强的一步。可以根据想从数据库中得到结果(包括要打印的报表、要使用的表单等)设计表结构的线索,也可以根据数据完整性的要求去分类存放数据到表。

(2) 同一信息只保存一次将减少出错的可能性。

(3) 防止删除有用信息。

(4) 确定所需字段:为了确定表的字段,首先决定需要在表中了解有关人、物或事件的哪些信息。可以把字段看作是表的属性。表中每个记录(或每行)包含了同样的字段或属性集合。

(5) 使用主关键字段:为使 VFP 更有效地工作,数据库的每个表都必须有一个或一组字段可用以唯一确定存储在表中的每个记录,通常使用唯一的标识号作为这样的字段(如雇员 ID 号或系列号)。利用主关键字可以迅速关联多个表中的数据,并把数据组合在一起。