

Visual C++ 程序设计

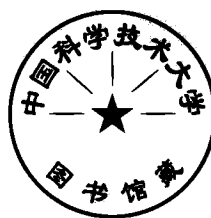
沈一梁 胡明成 等 编



科学出版社

Visual C++ 程序设计

沈一梁 胡明成 等 编



科学出版社

1996

(京)新登字 092 号

内 容 简 介

Visual C++ 是用户所喜爱的面向对象的 Windows 程序设计环境,它不仅具有方便的用户界面,而且还提供了丰富的类库。借助 Visual C++ MFC 类库中的类和 Visual C++ 的交互式工具,就能很容易生成一个可以实际运行的 Windows 应用程序。

本书不仅讨论了如何利用 Visual C++ 类库来开发 Windows 应用程序,还从一般的数据结构着手,讨论如何设计和实现具有特定功能的类。

本书可作为大专院校计算机专业学生的教材或参考书,也可供普通 C++, Visual C++ 和 Windows 程序员参考。

Visual C++ 程序设计

科学出版社出版

北京东黄城根北街 16 号
邮政编码:100717

中国科学院印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1996 年 1 月第 一 版	开本:787×1092 1/16
1996 年 1 月第一次印刷	印张:22
印数:1-3 000	字数:510 000

ISBN 7 03 004690 0/TP·433

定 价: 32.00 元

前 言

Visual C++是一个面向对象的 Windows 应用程序开发环境。在这个开发环境中,程序员不必面对每建立一个应用程序都要亲手设计用户界面的枯燥工作,而只需选取菜单命令,Visual C++系统就会生成一个可实际运行的 Windows 应用程序框架,然后利用基于 Windows 的 C++源程序编辑器(它也是第一个可由 C/C++编译器使用的基于 Windows 的编辑器),就可以在 AppWizard 的基础上建立自己的面向对象的应用程序。

除了具有 C++应用程序生成器和基于 Windows 的编辑器之外,Visual C++还包含下列组件:

- (1) MFC 类库:它是多个常用类的集合。
- (2) 代码实例和联机帮助:它包括多个完整的应用程序实例(它们演示了 MFC 类库的主要功能)和若干个技术要点(它们描述了 Visual C++的实现细节)。
- (3) 基于 Windows 的、面向图形的编程工具,包括源程序浏览器、源程序编辑器、使 C++代码和 Windows 消息及类成员函数相联系的交互式工具、可用来编写 Visual C++文本程序的 QuickWin 库、预编译的头文件和源文件。

尽管 Visual C++能够帮助程序员构筑应用程序框架,也能生成高效率的代码,但它无法自动写出实现应用程序功能的代码。所以,熟练掌握一些程序设计方法,尤其是会利用 MFC 类库编程是非常必要的。本书的目的即在于此。本书将满足那些对 C 语言和 Windows 环境比较熟悉而又想学习使用 Visual C++编程的人的需要。

本书不只讨论 Visual C++本身(软件包中的联机文档和书面手册对此有相当详细的说明),而是更侧重于面向对象的 Windows 编程。本书不是为初学者编写的,读者应至少有一些 C 语言编程经验,并了解 Windows 编程环境。

本书前三章由沈一梁和胡明成执笔,第四、五两章由何萧云执笔,第六、七两章由陈欣元执笔,第八至十章由童长明和蔡宪钟执笔,最后一章由胡俊执笔。何晓莉和魏明明为本书的编排付出了辛勤的劳动。

由于 Visual C++中的许多概念还没有形成一个公认的标准,加上时间有限,因此书中难免有不足甚至错误之处,望读者指正。

目 录

第一章 Visual C++ 环境简介	1
1.1 Visual C++ 的不同版本	1
1.2 Visual C++ 组件	1
1.2.1 Visual Workbench 集成环境	3
1.2.2 Visual Workbench 编辑器	3
1.2.3 Visual C++ 中的 VBX 控件	5
1.3 Visual C++ 的安装	6
1.3.1 安装 Visual C++ 的硬件和软件需求	6
1.3.2 单步与多步安装	6
1.3.3 关于安装 Visual C++ 的简要说明	6
1.4 用 Visual C++ 设计程序	7
1.4.1 QuickWin 应用程序	7
1.4.2 Visual C++ 工程	8
1.4.3 建立一个简单的 QuickWin 程序	8
1.5 总结	11
第二章 类与对象	12
2.1 抽象类的设计	12
2.1.1 抽象类作为基类	12
2.1.2 抽象类的对象	19
2.1.3 几种预定义的抽象类	27
2.2 多态行为	28
2.2.1 举例	29
2.2.2 CStrArray 类	34
2.2.3 CStrFixedQue 类	34
2.2.4 CStrFixedStack 类	35
2.2.5 测试程序	35
2.3 结构	39
2.3.1 模拟可扩展结构	39
2.3.2 类体系中的可扩展结构	44
2.4 总结	50
第三章 类体系	52
3.1 类的特殊实例	52
3.2 测试	63
3.3 使用特殊实例	69
3.4 总结	80
第四章 一些常用类的用法	81
4.1 字符串类	81
4.1.1 构造函数	84
4.1.2 属性函数	84
4.1.3 访问函数	84

4.1.4	赋值操作符	85
4.1.5	连接操作符	85
4.1.6	比较函数	86
4.1.7	提取函数	87
4.1.8	转换函数	87
4.1.9	查找函数	88
4.1.10	测试程序	89
4.2	字符串类的扩充	96
4.3	数组类	97
4.3.1	字符串数组类	98
4.3.2	字符串数组类的扩充	99
4.3.3	其他数组类	100
4.3.4	测试程序	100
4.4	表类	107
4.4.1	串表类	107
4.4.2	测试程序	110
4.5	映射类	118
4.5.1	串映射类	119
4.5.2	测试程序	121
4.6	总结	130
第五章	动态内存管理	131
5.1	内存模式	131
5.2	各种类型的指针	131
5.3	内存管理函数	132
5.3.1	内存分配函数	132
5.3.2	内存释放函数	144
5.3.3	内存扩展和重分配函数	149
5.3.4	内存信息查询函数	164
5.3.5	内存校验函数	169
5.4	操作符 new, delete 和不同的内存模式	170
5.4.1	set_new_handler 函数	171
5.4.2	重载操作符 new	171
5.4.3	重载操作符 delete	173
5.4.4	重载 new 和 delete 的使用	173
5.5	实例程序	173
5.6	重载操作符 ->	176
5.7	总结	177
第六章	虚拟内存管理	179
6.1	_VHEAPINIT 函数	179
6.2	_VMALLOC 函数	180
6.3	_VREALLOC 函数	180
6.4	_VMSIZE 函数	181
6.5	_VLOAD 函数	182
6.6	_VLOCK 函数	183
6.7	_VLOCKCNT 函数	183
6.8	_VUNLOCK 函数	183

6.9	_VHEAPTERM 函数	184
6.10	_VFREE 函数	184
第七章	访问对话框数据	185
7.1	自定义数据传输类	185
7.2	模态对话框的数据传输(一)	187
7.3	非模态对话框的数据传输	193
7.4	模态对话框的数据传输(二)	202
7.5	利用数据交换类传输数据	209
7.5.1	简介	209
7.5.2	数据传输机制	213
7.5.3	数据交换类	213
7.5.4	传输模态对话框中的数据(一)	214
7.5.5	传输模态对话框中的数据(二)	220
7.5.6	传输列表框中的数据	227
7.6	总结	233
第八章	对话框类的使用	234
8.1	软件需求	234
8.2	文件对话框类	234
8.2.1	支持类和结构	235
8.2.2	激活文件对话框	238
8.2.3	帮助函数	238
8.2.4	一个改进的文件统计程序	238
8.3	字体对话框类	241
8.3.1	支持类和结构	242
8.3.2	帮助函数	244
8.3.3	一个程序实例	244
8.4	颜色对话框类	247
8.4.1	支持类和结构	248
8.4.2	帮助函数	249
8.4.3	一个程序实例	249
8.5	打印对话框类	251
8.5.1	支持类和结构	253
8.5.2	帮助函数	255
8.5.3	一个程序实例	256
8.6	查找与替换对话框类	259
8.6.1	支持类和结构	261
8.6.2	通知父窗口	263
8.6.3	帮助函数	263
8.6.4	一个程序实例	264
8.7	总结	268
第九章	C++与 Visual C++异常	269
9.1	C++异常概述	269
9.1.1	异常的鉴别	270
9.1.2	异常的命名	271
9.1.3	异常和无错误代码转移	272
9.1.4	未处理的异常	272

9.1.5 处理异常的方法	272
9.2 Visual C++ 异常	272
9.2.1 Visual C++ 异常语法	273
9.2.2 MFC 异常类	274
9.2.3 异常的产生	274
9.3 CException 类	274
9.4 内存异常类	274
9.5 文件异常类	278
9.6 档案异常类	287
9.7 资源异常类	296
9.8 用户异常类	303
9.9 非支持异常类	310
9.10 OLE 异常类	310
9.11 总结	313
第十章 与绘图有关的类	315
10.1 绘图对象类	315
10.1.1 GDI 对象类	315
10.1.2 画笔类	316
10.1.3 画刷类	317
10.1.4 字体类	317
10.1.5 位图类	320
10.1.6 调色板类	320
10.1.7 区域类	321
10.2 绘图类	321
10.2.1 CDC 类	322
10.2.2 CPaintDC 类	326
10.2.3 CClientDC 类	327
10.2.4 CWindowDC 类	327
10.3 CDC 类的图形属性	328
10.3.1 选择对象	328
10.3.2 画线	329
10.3.3 绘制外形	332
10.3.4 绘图属性	337
10.3.5 坐标变换	341
10.3.6 区域操作	342
10.4 总结	343
参考文献	344

第一章 Visual C++ 环境简介

用 Visual C++ 为 IBM 个人计算机及 PC 兼容机开发基于 Windows 的应用程序与用其他 C++ 语言开发基于文本的程序一样容易,甚至更容易一些,因为 Visual C++ 中有一套功能强大的、预先写好的类库 MFC,它可以极大地简化 Visual C++ 的编程工作。

MFC 2.0 不仅是一个 C++ 类库,还是一个为开发 Visual C++ 程序而特殊设计的“可视”系统。借助 Visual C++ MFC 类库提供的类和 Visual C++ 的交互工具,很容易就能生成一个可以实际工作的 C++ 应用程序。所有要做的就是选择菜单和对话框,告诉工作台(Workbench)希望生成哪一类应用程序,然后用 VWB 编辑器或其他 Visual C++ 工具编写应用程序代码。

1.1 Visual C++ 的不同版本

Visual C++ 软件有两种版本:Standard 版和 Professional 版。Professional 版出自 Microsoft C++ 7.0,而 Standard 版出自 Microsoft Quick C。这两种版本都可开发 Windows 应用程序,其主要区别在于用 Professional 版开发的程序既可在 Windows 下也可在 MS-DOS 下运行,而 Standard 版则只能开发 Windows 程序。本书重点讨论基于 Windows 的编程,因此两种版本的 Visual C++ 都可使用。

除了具有两种版本所共有的功能之外,Professional 版还有下列特点:

(1) 可以用来开发 MS-DOS 应用程序、MS-DOS 覆盖应用程序、MS-DOS.COM 应用程序以及 P 代码应用程序(P 代码是一种特殊代码,它所占的空间比机器代码还少,但运行起来却很慢)。比较而言,Standard 版只能开发基于 Windows 的应用程序(包括 QuickWin 程序)。

(2) Microsoft CodeView 调试器(尽管 Standard 版不带 CodeView,但它带有 Visual Workbench 源代码调试器);Visual C++ Professional 版的 CodeView 调试器可安装在 Visual Workbench 中。

(3) 包括命令行工具的硬拷贝文档。

(4) 包括软件开发工具包(SDK)的程序实例及 SDK 工具的硬拷贝文档。

1.2 Visual C++ 组件

Visual C++ 不仅是一个开发 C++ 应用程序的新工具,其中还有许多值得注意的新特点:

(1) Visual Workbench (VWB):它是一个集成开发环境,包括 Visual C++ 编辑器、一个功能强大的源程序调试器、一个浏览器和一个交互式 make 实用程序。VWB 编辑器取代了 Microsoft C++ 7.0 中基于文本的程序员开发平台编辑器。

(2) AppWizard:它是一个应用程序生成器,其中可以通过按鼠标按钮来生成完整的 Visual C++ 程序。从 Visual Workshop 的 Project 菜单中选择 AppWizard 命令就可以生成能够实际运行的 C++ 应用程序,并带有一个窗口和一个菜单条,然后用其他 Visual C++ 工具把自己的代码和资源加到该程序中,使 AppWizard 应用程序扩展成实际需要的程序。有了 AppWizard,再也不用在开发软件之前建立和运行 Windows 应用程序。借助 AppWizard,可以交互地使用 MFC 类库提供的类和 Visual Workbench 中基于图形的内置式交互工具来生成 Visual C++ 应用程序。

(3) AppStudio:它是一个建立和编辑 Windows 资源的交互式图形工具。有了 AppStudio,就可以通过用鼠标在屏幕上编辑诸如对话框、菜单、图标、位图和光标等来设计资源。AppStudio 自动为所建立的资源生成资源文件,还可以用 Visual Workbench 编辑器编辑 ASCII 资源文件,以便对资源作更细致的调整。

(4) ClassWizard:它是一个交互式工具,可用来建立新的类,把消息映射成类成员函数,或者把控件映射成类变量成员。利用 ClassWizard 建立类时,只需从可使用的类清单中选择一个基类,然后键入派生类的名字,ClassWizard 就会自动建立用以定义和实现新类的源文件,并把这个新类加到正在开发的程序中。在开发程序时,可用 ClassWizard 建立程序所需要的类,包括消息处理和消息映射例程(用来定位处理消息的代码)。

(5) Visual Workbench Browser:可用来查找关于模块、常量、宏、变量、符号和类之间关系的信息。借助 Browser,我们可以观察类体系、函数调用树以及符号定义和引用等。

除了前面提到的开发工具之外,Visual C++ 中还有下列组件:

(1) 一套运行库,主要指类库和函数库。

(2) 一套 Windows 库,主要指动态链接库(DLL)。

(3) Windows 软件开发工具包(SDK)的联机文档。

(4) 一个基于 Windows 的源程序调试器,可在开发程序的任何时候激活该调试器。

图 1.1 展示了 VWB 中的 Debug 菜单。Visual C++ 随带的文档对 Visual C++ 调试器作了详细的说明。

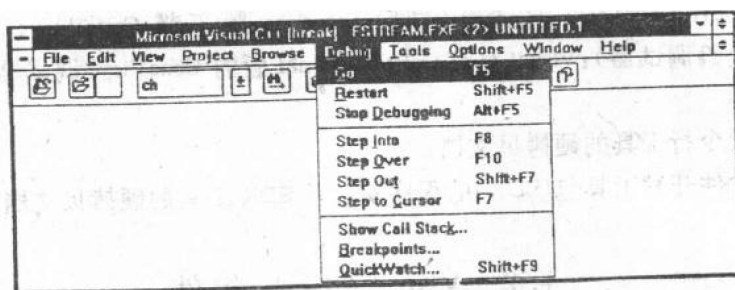


图 1.1 Visual Workbench Debug 菜单

从 Debug 菜单中选择 Go 命令就可从 Visual Workbench 进入调试器,而且不必中断程序开发就可完成下列工作:

(1) 设置、使用和清除断点。

(2) 单步调试、多步调试或退出函数。

(3) 测试和改变变量的值。

- (4) 利用观察窗口监视变量和表达式的值。
- (5) 检查 CPU 寄存器和标志的值。
- (6) 跟踪程序的运行或对所有已调用但尚未返回的函数进行跟踪。

1. 2. 1 Visual Workbench 集成环境

Microsoft Visual Workbench 是一个在 Microsoft Windows 操作系统下运行的集成化的程序开发环境,其中包含七个组件,每一个都在 Windows 下运行。Visual Workbench 的七个组件是:

(1) 一个基于 Windows 的功能强大的文本编辑器:启动了 Microsoft Visual C++ 之后, Visual C++ 编辑器就出现在屏幕上,准备好接受用户的输入或者菜单及工具条命令选择。

(2) Visual Workbench 图形浏览器:它是 Microsoft C/C++ 所带的基于文本的 Microsoft Source Browser 的 Windows 版本。有了这个浏览器,就可以查找相关的符号并检查它们之间的关系。浏览器使用了一个数据库,其中含有每个符号在何处被定义和使用的信息,还描述了程序中所用的模块、常量、宏、变量、函数及类之间的关系。

(3) Visual C++ 编译器:Visual C++ Professional 版带有一个优化编译器,Standard 版带有一个标准编译器。前者可以编译 Windows 下的 P 代码应用程序和 MS-DOS 应用程序,而后者却不能。稍裁在 /C++ Compiler Options 对话框中设置 Visual C++ 编译选项。

(4) 一个链接器:Visual C++ 链接器可从 Linker Options 对话框中进行控制。

(5) 一个新的源程序调试器:它在 Windows 下运行,而且不必退出编辑器就可使用。

(6) 一个制作实用程序:在建立一个 Visual C++ 应用程序之前,必须先生成一个应用程序工程,其中包含源文件、资源和库,还包含用来建立程序的编译和链接命令。在 Visual Workbench 中,只需在 Project 菜单中选择 New 命令就可生成一个工程。只要选择相应的菜单命令,Visual C++ 的制作程序就能建立应用程序。

(7) 一个帮助数据库:任何时候都可从 Visual Workbench 中获得联机帮助,只需从 Help 菜单中选择需要帮助的主题即可。图 1.2 列出了 Help 菜单。

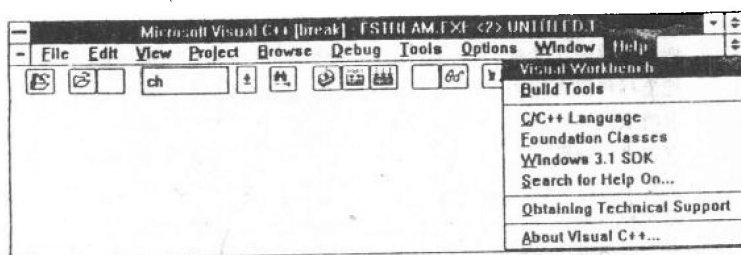


图 1.2 Visual Workbench Help 菜单

1. 2. 2 Visual Workbench 编辑器

读者如果使用过基于 Windows 的文本编辑器,则在使用 Visual Workbench 时将不会有什么困难。它是一个标准的基于 Windows 的编辑器,其中带有一个可用鼠标选择菜单命令的下拉式菜单条。此编辑器还有一个工具条,可在其中用鼠标单击图标以执行命令。图 1.3 展示了 VWB 编辑器的主屏幕。

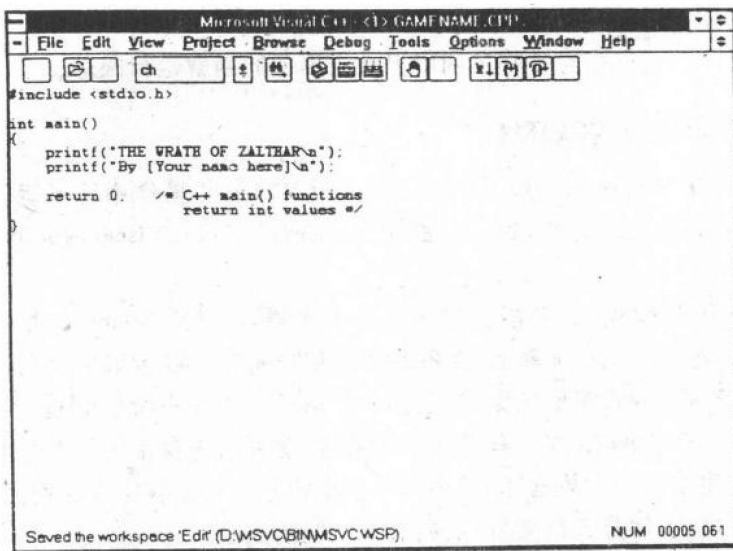


图 1.3 Visual Workbench 编辑器画面

VWB 编辑器提供了 File 和 Edit 菜单, 可以从中选择命令来执行标准文件和编辑操作, 也可使用与菜单命令等价的标准键盘命令。例如, 可以用 Ctrl - C 把文本中所选择的块拷贝到剪贴板上, 用 Ctrl - X 把所选择的块移到剪贴板上, 用 Ctrl - V 把剪贴板上的块粘贴到文档中。表 1.1 列出了其他键盘命令。

表 1.1 Visual Workbench 编辑器的键盘命令

功 能	键
把一个字移到左边	Ctrl + ←
把一个字移到右边	Ctrl + →
把光标移到当前行的行首(缩进格式)	Home
把光标移到当前行的行首	Home, 再按 Home
把光标移到下一行的行首(缩进格式)	Ctrl + Enter
把光标移到当前行的行尾	End
把光标移到文件头	Ctrl + Home
把光标移到文件尾	Ctrl + End
取消上次编辑	Ctrl + Z
取消上次编辑	Ctrl + Backspace
恢复上次编辑	Ctrl + A
删除到字尾	Ctrl + T
把文本拷贝到剪贴板上	Ctrl + C
把文本拷贝到剪贴板上	Ctrl + Ins
把文本剪贴到剪贴板上	Ctrl + X
把文本剪贴到剪贴板上	Shift + Del
从剪贴板粘贴文本	Ctrl + V
从剪贴板粘贴文本	Shift + Ins
把光标移到匹配的括号	Ctrl +]
插入一个制表符	Tab
切换 tab 符号的显示	Ctrl + Alt + T

除了标准功能之外, Visual Workbench 编辑器还有许多特殊功能, 例如:

(1) 彩色文本加亮显示: VWB 编辑器以不同的颜色加亮显示语言关键字、标识符、注释和字符串, 使源代码中的循环和注释等文本块易于识别。

(2) 一个工具条: 包括对应于 14 种常用操作的图标, 如查找-替换工具和几个有用的调试工具。借助工具条可以设置和清除断点, 打开一个窗口来监视变量, 跳过、进入或退出函数。

(3) 灵活的对话框: File 菜单中的 Open File 和 Save As 对话框可以“记住”以前输入的文件名, 在编辑期间要多次查看某一特定类型的文件时, 它可以免去多次输入文件名的麻烦。

(4) 文件查找: VWB 中有一个功能强大的查找-替换实用程序, 可以查找整个单词, 进行区分大小写的查找, 或者查找正则表达式; 可以在整个文件中进行前向和后向查找; 可以从 View 菜单中选择 Line 命令而使光标移到文件中的某一行。

(5) 书签: 可以从 View 菜单中选择 Bookmark 命令来设置、清除、切换和移动书签。

(6) 调试: Visual Workbench 中有一个可通过在 Debug 菜单中选择命令来调用的源代码调试器(不必退出 VWB 编辑器)。如果有 Professional 版本的 Visual C++, 则可从 Tools 菜单中运行 CodeView 调试器。通过从 Options 菜单中选择 Debug 命令, 可以指定 VWB 调试器在调试程序时所用的选项。

(7) 从编辑器中启动的 VWB 工具: VWB 调试器并非唯一能从 VWB 编辑器中启动的工具, 通过从 VWB 菜单上选择命令可启动前面讨论的三种工具以及一些新工具:

1) AppStudio: 一个可以生成菜单、对话框、位图和图标资源的、基于图形的交互式工具。

2) AppWizard: 可方便地生成能实际运行的 Visual C++ 应用程序, 程序员可再用 AppWizard 把前面生成的应用程序扩展成一个完整的 Windows 应用程序。

3) ClassWizard: 一个可生成和管理 C++ 类的工具, 它还能自动生成并编辑用来处理对话框消息及数据的代码。

4) VWB 浏览器: 可通过观察类体系、函数调用树、符号定义和符号引用来查找信息。

5) VWB 帮助工具: 对与 Visual C++ 及 VWB 环境有关的各种主题提供了联机帮助。

(8) 定制编辑器: 可以调整 Visual Workbench 中的许多部件, 使之适应于程序员的个人喜好和编程的需要。例如, 可在 Tools 菜单中增加命令, 设置 Visual Toolbox 所用的目录路径, 设置 VWB 文本窗口所用的字体和字型, 改变用来加亮显示源程序的颜色。

1.2.3 Visual C++ 中的 VBX 控件

有时被称作 VBX (Visual Basic) 的许多用户控件是为 Microsoft Visual Basic 环境开发的。作为一个独立的产品, 有了被称作 Microsoft Control Park 的软件包之后, 就可在 Visual C++ 中使用用户控件了。

用户控件是 Visual C++ AppStudio 或 Microsoft Visual Basic Toolbox 中的 Control Palette 的一个扩展。在 AppStudio 中安装了用户控件之后, 就可与使用任何标准内置控件一样来使用这些用户控件。在把用户控件加到用 Visual C++ 编写的应用程序中之后, 这些控件就成了应用程序的一部分, 可像使用其他任何控件一样来使用它们。

Microsoft Visual Control Park 中包含一套具有三维造形的控件: 一个动画按钮(用鼠标“点”一下就会移动)、一套用于多种目的的对话框(如装载和保存文件、进行通信管理等)、一个

用来设计多种图形的控件和许多其他控件。

要在 Visual C++ 中使用 VBX 控件,必须先 在 AppStudio 资源编辑器中安装它,使之成为 AppStudio 环境的一部分,然后才能用 ClassWizard 定义所安装控件的消息映射;也可用 ClassWizard 对这些控件的属性进行初始化。

1.3 Visual C++ 的安装

Visual C++ 中包含一个可以自动安装它自身的安装程序,用户所要做的只是把 Visual C++ 安装盘插入到驱动器中并运行安装程序。安装程序显示出一个 Installation Options 对话框,利用这个对话框可在各种配置下安装 Visual C++。

1.3.1 安装 Visual C++ 的硬件和软件需求

安装 Visual C++ 的最低硬件和软件的需求如下:

- (1) 一台 IBM 个人计算机或与 IBM 100% 兼容的 PC 机,带 80386 或更高档的处理器。
 - (2) 操作系统是 MS-DOS 3.3 或更高版本。
 - (3) 配置有 VGA 显示器。
 - (4) 主机上有 4MB 可用内存(推荐采用 6MB 内存)。
 - (5) 硬盘空间足以安装各种选项。Visual C++ 安装程序根据安装时的不同选项给出所需硬盘空间的信息,并在用户确认有足够的空间后才开始拷贝文件。
 - (6) 一个 1.2MB 的 5.25 英寸驱动器或一个 1.44MB 的 3.5 英寸驱动器。
 - (7) Microsoft Windows 3.1 或 Microsoft Windows for Workgroups,在增强型模式下运行。
- 如果采用 Visual C++ Professional 版,则可配置计算机系统,使之在 MS-DOS 命令行中编译和链接 Visual C++ 程序,但必须在 Windows 下安装 Visual C++。

1.3.2 单步与多步安装

可以一次把 Visual C++ 软件包全部安装好,也可以先安装一个较小的配置,以后再安装更多的库、程序实例、帮助文件和其他组件。

在安装过程中,可以按 F1 键或单击对话框中的 Help 按钮来得到联机帮助。

注意,由于 Visual C++ 主要是与 Visual Toolbox 一起使用的,所以安装程序一般不把系统配置成利用 MS-DOS 命令行中的命令来编译和链接 Visual C++ 程序,但是对于 Visual C++ Professional 版,对 AUTOEXEC.BAT 作一些改动就可以用 MS-DOS 命令行来建立 Visual C++ 程序;Standard 版则不然,因为它与 MS-DOS 不兼容。

有关安装 Visual C++ 的完整说明,可参阅其随带的文档。

1.3.3 关于安装 Visual C++ 的简要说明

Visual C++ 软件包中包含一个安装程序,它通过让用户回答问题来提供所用计算机系统的信息及个人偏爱,并安装 Visual C++ 及其支持工具、联机帮助和有关文档。

运行安装程序时可作如下选择:

- (1) 应考虑编写 C/C++ 程序时所使用的内存模式。选择内存模式时要考虑可用内存

的大小以及要编写的是小型程序、“紧凑”型程序、中型程序还是大型程序。如果对这些问题没有把握,请先参考本书中的有关内容。

(2) 程序是否支持浮点数学运算。如果支持,应考虑怎样实现它。

(3) 所用的 C++ Windows 库。如果有疑问,可选择缺省选项。

如果在选择安装选项时出错或者对计算机系统作了扩展,可重新运行安装程序以选取合适的选项。

对 Visual C++ 安装程序的详细描述超出了本书的范围,Visual C++ 软件包所带的文档提供了安装程序的详细信息及运行方法。

1.4 用 Visual C++ 设计程序

可能用户所编写的应用程序的大多数都是基于 Windows 的。实际上,利用 Visual C++ Standard 版也只能编写基于 Windows 的程序。

利用 Visual C++ 可以编写两种基于 Windows 的程序:

(1) Windows 用户所习惯的、带有全部“修饰组件”的“纯”Windows 应用程序:多窗口支持、对话框、按动与拖动图标等。

(2) QuickWin 程序:就是把 Windows 应用程序和 MS-DOS 应用程序相混合后得到的程序,它使用了一个称作 QuickWin 库的特殊库。QuickWin 应用程序在屏幕上显示一个窗口和一个菜单条,除此之外几乎没有其他东西。QuickWin 程序不支持对话框,不使用图标,也没有 Windows 应用程序所用的其他许多交互工具,相反,QuickWin 程序所显示的窗口像一个视频监视器,用户可在窗口中输入文本,可用标准 MS-DOS 输入/输出例程实现输入/输出操作。由于 QuickWin 应用程序易于生成,因此它在程序开发中还是有用的,它在生成原型代码和完成不需要完整 Windows 界面而更注重速度的工作中是比较方便的。

图 1.4 所示为一个 QuickWin 程序的屏幕显示。

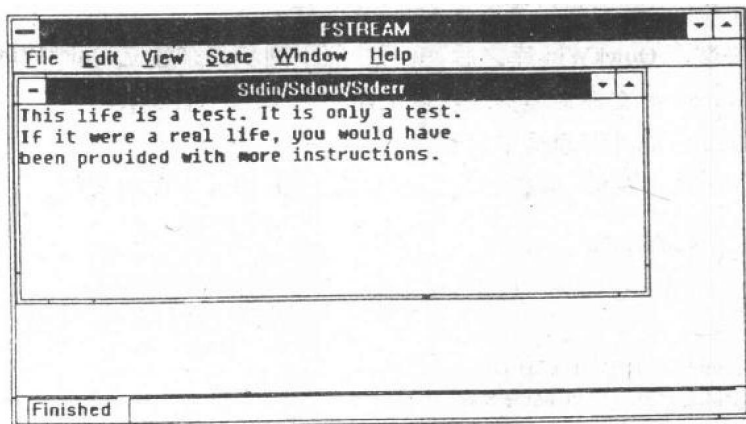


图 1.4 一个 QuickWin 程序的屏幕显示

1.4.1 QuickWin 应用程序

QuickWin 应用程序在 Windows 操作系统下运行,但其用户接口采用的是文本 I/O,所以

更像在 MS-DOS 下运行。QuickWin 程序在屏幕上显示一个多文档界面(MDI)窗口,用户通过在窗口中输入文本与程序进行交互,就像与 MS-DOS 下运行的程序进行交互一样。

由于 QuickWin 程序运行起来像标准的文本程序,所以可以使用 STDIO. H 头文件中定义的 printf 和 scanf 函数,也可以使用 IOSTREAM. H 头文件中定义的 C++ 输入/输出流。

由于 QuickWin 程序和 MS-DOS 程序使用同一种文本 I/O,所以用 QuickWin 库重新编译 MS-DOS 程序就可方便而快速地把这些现有的 MS-DOS 程序移植到 Windows 中。

QuickWin 库可处理大多数 MS-DOS 程序(包括使用了 MS-DOS GRAPHICS. LIB 库的程序)中用到的 I/O 操作,但它不能处理直接对硬件进行操作的 DOS 过程,如 BIOS 中断和对视频内存的直接操纵。

注意,尽管不能用 Visual C++ Standard 版编写 MS-DOS 程序,但它的确支持 QuickWin。因此,如果拥有 Visual C++ Standard 版而又想编写 MS-DOS 风格的程序,就可以用 QuickWin 库编译这些程序,然后在 Windows 下运行。

1.4.2 Visual C++ 工程

在编写 QuickWin 程序或任何 Visual C++ 程序之前,必须先建立一个工程。每一个 Visual C++ 程序都是围绕着一个工程的。工程中包含组成程序的所有源文件、资源和库,还有建立程序所需要的编译和链接命令。

工程用简单的方法对建立程序或类库所需要的文件及库作了记录,这就简化了程序开发的难度,只需对上一次编译或建立程序之后修改过的文件重新进行编译,因而加快了软件开发的速度。例如,假定工程中有五个文件,自上次编译之后只编辑了其中一个文件,Visual C++ 编译器在链接前就只对修改过的那个文件进行编译。

1.4.3 建立一个简单的 QuickWin 程序

要建立 QuickWin 程序,必须在 Visual C++ 中安装 QuickWin 库。如果还没有安装,可以再次运行 Visual C++ 安装程序来安装 QuickWin 库。

在确信已安装了 QuickWin 库之后,就可按照下列步骤来建立 QuickWin 程序:

- (1) 利用 Windows 文件管理员在硬盘上建立一个工程子目录。
- (2) 从 Windows 文件管理员中启动 Visual C++。
- (3) 当 Visual Workbench 编辑器出现在屏幕上时,输入下面的程序:

```
#include <stdio.h>

int main ( )
{
    printf ("THE WRATH OF ZALTHAR\n");
    printf ("By [Your name here]\n");
    return 0;    /* C++ main(), returns int value */
}
```

参考图 1.3,看看输入上面的程序之后屏幕是什么样子。

读者可能意识到这是用旧式的标准 C 编写的“Hello, world”程序的一个变体,它在 C 和 C++ 中都可运行。

(4) 选择 File 菜单中的 Save As, 将这个程序保存到 GAMENAME.CPP 中。如果以前保存过该程序, 则应选择 Save 菜单而不是 Save As 来保存文件, 如图 1.5 所示。

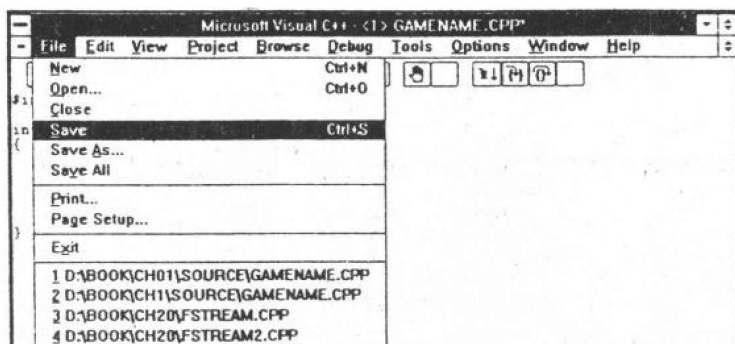


图 1.5 保存一个源代码文件

(5) 从 Project 菜单中选择 New, 如图 1.6 所示。

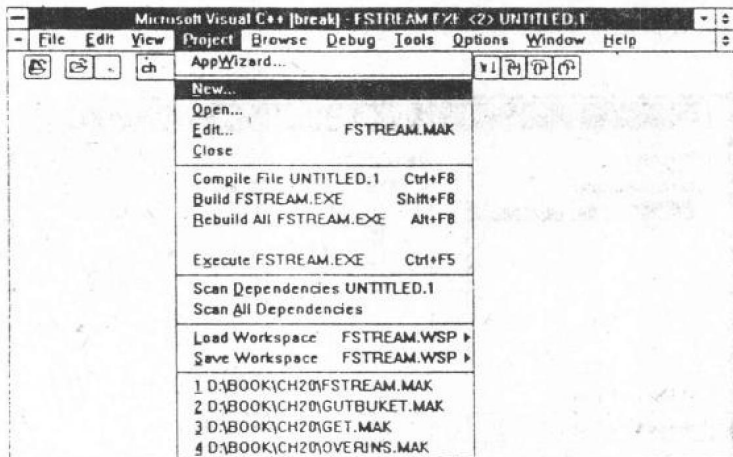


图 1.6 从 Project 菜单中选择 New

(6) 在出现 New Project 对话框之后, 在标题为 Project Name 的文本框中键入 GAMENAME, 如图 1.7 所示。

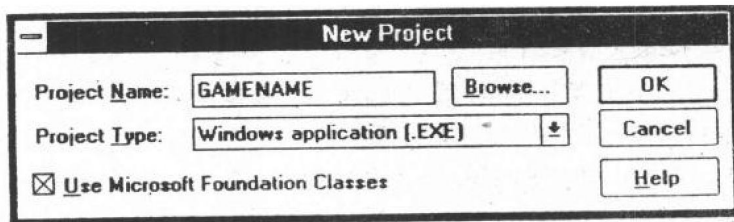


图 1.7 命名一个工程

(7) 打开标题为 Project Type 的列表框, 选择 QuickWin application[.EXE]作为项目类型, 如图 1.8 所示。图示的列表框是 C++ Professional 版显示的; 如果采用 Standard 版, 则显