

巧学巧用

Visual C++ 2.X

for Windows 95

钱子明 应 晓 主编

最新出版



電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

巧 学 巧 用

Visual C++ 2.X for Windows 95

钱子明 应 晓 主编

电子工业出版社.

内 容 简 介

本书是为利用 Visual C++ 2.X 进行 C 和 C++ Windows 95 程序设计的人员编写的。首先介绍 Windows 95 的新特点,介绍 Visual C++ 2.X 集成环境(IDE)、工程管理器 and 调试器的用法;然后介绍用 C 编写 16 位的 Windows 95 应用程序,用 C 编写 32 位的 Windows 95 应用程序的技术,介绍用 C++ 的 Microsoft Foundation Class Library (MFC)编写 Window 应用程序;最后介绍用 MFC 设计 Windows 95 OLE 2.X 应用程序的技术。本书内容详实,图文并茂,示例说明性好,讲述由浅入深,适合于软件开发人员、大专院校师生和广大计算机爱好者参考和阅读。

35/14/1/7

巧学巧用

Visual C++ 2.X for Windows 95

钱子明 应 晓 主编

责任编辑:冯文全(特约) 文宏武

电子工业出版社出版

北京海淀区万寿路 173 信箱(100036)

电子工业出版社发行 各地新华书店经销

北京大中印刷厂印刷

*

开本: 787×1092毫米 1/16 印张: 25.5 字数: 653千字

1995年12月第一版 1995年12月北京第一次印刷

印数: 5000册 定价: 40.00元

ISBN 7-5053-3338-0/TP·1273

1 B

出版说明

近几年来,随着计算机的发展及其广泛普及,它在人们的工作和生活中发挥着越来越大的作用。作为计算机“灵魂”的软件的发展也是日新月异,一些功能强大、界面友好、易学易用的优秀软件层出不穷,同时一些深受人们欢迎的流行软件的版本也不断更新、提高。

随着这些优秀软件的推出和更新,一些软件开发人员、软件用户及广大的计算机爱好者,都需要了解和学习这其中的新概念、新技术,需要尽快地熟悉并精通这些软件的操作使用 and 开发的方法与经验。为满足广大读者的迫切需求,我社特邀请了具有丰富教学经验的计算机专家,精心策划、编写了这套最新流行实用软件“巧学巧用”丛书。

这套丛书的主要特点是从最基本的概念入手,以简单易懂的解说,辅以大量的典型应用程序实例及微机运行屏幕显示图例,由浅入深地详细而系统地介绍各种最新流行实用软件的功能、特点、使用和编程方法与技巧。这套书内容翔实,循序渐进,图文并茂,给出了大量的学习经验和使用、编程技巧。因此,读者可以花最少的学习时间,以最佳的捷径,掌握并精通这些流行应用软件;这套书既可作为从零开始的读者的入门向导,又包含了技术内容的深入剖析,把程序设计的专家经验与软件复杂的技术细节紧密地融合在一起,因此,对有经验的程序员来说,也很有帮助。书中的大量例程都是软件专业人员精心设计开发的,并经过上机调试、验证。这套丛书中的大部分都有配套练习、演示磁盘,收录了书中大量的例程和相关的开发工具软件,读者可到新华书店购买或与电子工业出版社发行部软件科联系邮购。

首批计划推出的有以下九种:

巧学巧用 Visual C++ 2.X for Windows 95

巧学巧用 Visual C++ MFC for Windows 95

巧学巧用 Borland C++ 5.0 for Windows 95

巧学巧用 Borland C++ OWL for Windows 95

巧学巧用 Visual FoxPro 3.0 for Windows 95

巧学巧用 Visual Basic 4.0 for Windows 95

巧学巧用中文 Windows 95

巧学巧用 Delphi for Windows

巧学巧用中文 dBASE 5.0 for Windows

今后我们还将不断推出其他的新的流行实用软件选题方面的图书,欢迎广大读者和我们联系,提出宝贵意见和建议,联系地址:北京万寿路 173 信箱(100036),联系人:文宏武。

电子工业出版社

1995 年 9 月

前 言

Microsoft 在 1993 年推出 Visual C++ 1.0 时,就首先在 Windows 环境的 C/C++ 编译器方面占据了领先地位。现在,随着 Visual C++ 2.0(目前最新的是 2.1)的推出,Microsoft 继续占领着基于 Windows 的编译器领域的领先地位。Visual C++ 2.0 是功能强大的 32 位开发环境,能使开发人员生成 32 位的 Windows(包括 Windows 95 和 Windows NT)应用程序。Visual C++ 2.0 也允许 Windows 程序员把 Windows 应用程序移植到 Apple Machintosh 68XXX 平台上。

Visual C++ 2.0 向程序员提供了许多 Visual C++ 1.X 的 16 位和 32 位版本所不具有的新特性,如 Microsoft Foundation Classes (MFC) 3.0, Object Linking and Embedding (OLE) 2.0, OLE Custom Controls 和 Open Database Connectivity (ODBC) 的 32 位支持。Visual C++ 2.0 是 Visual C++ 产品系列的第二次重大改进;另一次重要的改进是从 Visual C++ 1.0 升级到 Visual C++ 1.5(现在是 1.5X)。

Visual C++ 2.0 只通过 CD-ROM 发行,若把它放在磁盘上,则至少需要大约 100 张 3.5 寸的磁盘。在硬盘上完整安装 Visual C++ 2.0,一般需要占据 75 至 100MB 的磁盘空间。本产品的许多文档和许多示例一样,在完整安装时都被安装了。然而,空间需求还不包括 Books Online 或附加的 MFC 或 OLE 例子,这些共有 30MB 的可执行文件;这些都保留在 CD 上。如果不打算安装 Visual C++ 2.0 的示例应用程序,就可以把它们留在 CD 上,这样就能减少对硬盘空间的需求。所有文档都可以通过 Books OnLine 帮助系统来使用。

使用 Visual C++ 2.0 的主平台是由有 8MB RAM、200MB 硬盘的 386/33 组成。386/33 的性能水平是最低的。推荐使用有 16MB 的 RAM 和 500MB 硬盘空间的 486。运行 Visual C++ 2.0 的最实用的系统是有 32 MB 的 RAM 和 2G 硬盘的 100MHz 的 Pentium 系统。

Object Linking and Embedding(对象链接与嵌入),一般称作 OLE,是从 Dynamic DataExchange(动态数据交换)(DDE)发展而来的。DDE 被引入 Windows 的早期版本来允许程序直接交换数据。这种数据交换很复杂,而且很少有程序支持 DDE。Microsoft 用 OLE 实现对象转换的下一代来改进了 DDE 的缺点。OLE 的第一版是对 DDE 的主要改进,但仍然还有许多地方值得改进。使用 OLE1 还需要较多的编程上的努力。OLE 在达到当前的 OLE2 标准的过程中经历了几个发现阶段。有了 OLE2,应用程序可以是客户、服务器或包容器。完整的 OLE 功能包括如拖放、可视编辑和 OLE Automation(自动化)等支持。

Visual C++ 2.0 中包含 ODBC SDK、OLE2 SDK 和 OLE Custom Control 开发工具,这些开发工具是 Visual C++ 2.0 的主要特色之一。

当然,MFC 3.0 的增加也很重要,因为它添加了一些 Visual C++ 1.X 中没有的功能。

本书是为利用 Visual C++ 2. X 进行 C 和 C++ Windows 95 程序设计的人员编写的。首先介绍 Windows 95 的新特点,介绍 Visual C++ 2. X 集成环境(IDE)、工程管理器 and 调试器的用法;然后介绍用 C 编写 16 位的 Windows 95 应用程序,用 C 编写 32 位的 Windows 95 应用程序的技术,介绍用 C++ 的 Microsoft Foundation Class Library(MFC)编写 Window 应用程序;最后介绍用 MFC 设计 Windows 95 OLE 2. X 应用程序的技术。本书内容详实,图文并茂,示例说明性好,讲述由浅入深,适合于软件开发人员、大专院校师生和广大计算机爱好者参考和阅读。

目 录

出版说明	I
前 言	III
第一章 Visual C++ 环境	1
1.1 Visual C++ 各种版本之间的区别	1
1.2 小 结	3
第二章 新的编译器特点	4
2.1 模 板	4
2.2 异常处理	13
2.3 结构化异常处理	15
2.3.1 __try	15
2.3.2 __finally	15
2.3.3 __leave	15
2.3.4 __except (filter)	16
2.3.5 C++ 异常处理	17
2.4 小 结	18
第三章 MFC 中的新特征	19
3.1 对 32 位程序的支持	19
3.2 新的 GDI 函数	19
3.3 新的 USER 函数	19
3.4 对异常处理的支持	19
3.5 关于集合的 C++ 模板类	19
3.6 制表对话框(特征页)	20
3.7 更先进的图示行支持	20
3.8 Unicode 和双字节字符集(DBCS)支持	20
3.9 新的 3D 控制	21
3.10 细标题条窗口	21
3.11 为 MFC 中的命令范围或控件 IDS 赋单个处理程序	21
3.12 MFC 3.0 对 CString 添加新的函数性	21
3.13 从 MFC 2.0 中移植时的 OLE 类支持	21
3.14 MFC 2.0 中移植时对数据库交互的支持	21
3.15 小 结	22
第四章 Visual C++ 2.0 的新特征	23
4.1 Visual Workbench	23
4.2 Visual Workbench 用户界面	23

4.2.1	File 菜单	23
4.2.2	Edit 菜单	27
4.2.3	Search 菜单	28
4.2.4	Project 菜单	31
4.2.5	Resource 菜单	35
4.2.6	Debug 菜单	36
4.2.7	Tools 菜单	41
4.2.8	Window 菜单	44
4.2.9	Help 菜单	45
4.2.10	创建新的工程文件	46
4.2.11	AppWizard 用户界面	46
4.2.12	Calc 计算器	50
4.2.13	为什么要有基于对话框的程序	55
4.3	Visual Workbench 编辑器	55
4.3.1	编辑 C/C++ 源文件	55
4.3.2	编辑资源对象	57
4.4	对 CALC 的总结	86
4.5	小 结	86
第五章	用 Visual C++ 调试	87
5.1	使用 Visual C++ 2.0 的标准调试	87
5.1.1	Disassembly 窗口	88
5.1.2	Memory 窗口	93
5.2	Just-IN-Time 调试	94
5.3	剖 析	95
5.3.1	哪些函数使用最频繁	96
5.3.2	哪些函数使用了大部分的时间	96
5.3.3	哪些函数从未被调用	96
5.4	为剖析准备程序	96
5.5	小 结	98
第六章	外部 Visual C++ 实用程序	99
6.1	PortTool	99
6.2	Spy++	99
6.2.1	Spy 菜单	100
6.2.2	Tree 菜单	101
6.2.3	View 菜单	101
6.2.4	Messages 菜单	101
6.2.5	Window 菜单	102
6.2.6	Help 菜单	102
6.3	DDE Spy	102

6.4	命令行编译器	102
6.5	链 接	105
6.6	Lib	108
6.7	Profiler	109
6.8	BSCMAKE	110
6.9	DumpBin	110
6.10	EditBin	111
6.11	NMake	112
6.12	Resource Compiler	113
6.13	PView	114
6.14	帮助编译器	114
6.15	WinDiff	114
6.16	ZoomIn	115
6.17	Windows 的调试版本	115
6.18	MFC TRACE OPTIONS	116
6.19	Shed:分段的超图 Hotspot 编辑器	118
6.20	帮助文件	118
6.21	小 结	121
第七章	16 位和 32 位 Windows 程序设计	122
7.1	Windows 程序设计语言	122
7.1.1	Windows 环境快速浏览	122
7.1.2	使用 Windows 的优越性	123
7.1.3	新的 Windows 可执行文件的格式	128
7.2	Windows 编程:概念和词汇	129
7.2.1	Windows 的窗口	129
7.2.2	Windows 布局	129
7.2.3	C/C++ 中的 Windows 类	132
7.2.4	OOP 及 Windows	132
7.2.5	发送和接收 Windows 消息	135
7.2.6	访问 Windows 函数	138
7.2.7	Windows 头文件:Windows.h	138
7.2.8	Windows 应用程序的组成成分	139
7.3	Microsoft Visual C++ Windows 工具	139
7.3.1	使用 Make 或 Project 文件	140
7.3.2	创建资源	140
7.3.3	使用集成的 Image 和 Dialog 编辑器	141
7.3.4	在命令行使用资源编译器(RC)	152
第八章	用 C 编写 16 位 Windows 95 应用程序	155
8.1	应用程序框架	155

8.1.1 有效地使用句柄	156
8.1.2 考察 Windows 应用程序中的成份	156
8.2 Make 或 Project 实用工具	165
8.2.1 NMAKE 工具	165
8.2.2 工程工具	166
8.3 一个简单的应用程序和例子	167
8.3.1 画椭圆	169
8.3.2 画弦	171
8.3.3 画饼图瓣	172
8.3.4 画矩形	174
8.4 使用 SWA 开发一个正弦波应用程序	174
8.5 创建 Windows 饼图应用程序	178
8.5.1 工程文件	186
8.5.2 PIE. DEF 文件	186
8.5.3 PIE. H 头文件	186
8.5.4 PIE. RC 资源文件	186
8.5.5 PIE. C 程序	188
8.6 传统 C Windows 编程的更多信息	190
第九章 编写 32 位 Windows 95 应用程序	192
9.1 向 Windows 95 移植的重要改变	192
9.1.1 Win32 的数据类型与结构	193
9.1.2 Win32 的句柄说明	193
9.2 95SWP 应用程序模板	194
9.3 添加资源	204
第十章 设计消息框和菜单	215
10.1 消息框(Message box)	215
10.2 菜单简介	221
10.2.1 使用资源	222
10.2.2 编译 RC 文件	222
10.2.3 创建简单的菜单	222
10.3 在用户程序中引入菜单	224
10.4 响应菜单选择及菜单程序实例	225
10.5 增加菜单加速键	229
10.6 加载加速键表	231
第十一章 对话框简介	235
11.1 对话框同用户的通讯方式	235
11.2 模态(Modal)和非模态(Modeless)对话框	236
11.3 接收对话框消息	236
11.4 激活对话框	236

11.5	创建简单对话框	237
11.5.1	对话框资源文件	237
11.5.2	对话框窗口函数	239
11.6	一个对话框例程	240
11.7	增加列表框	244
11.7.1	列表框基础	245
11.7.2	初始化列表框	245
11.7.3	处理选择	246
11.7.4	完整的列表框例程	247
11.8	增加编辑框	251
11.9	使用非模态对话框	252
第十二章	进程和线程多任务	260
12.1	创建特殊任务	260
12.2	创建多线程程序	269
12.2.1	创建线程	269
12.2.2	终止线程	270
12.2.3	简短的多线程例程	270
12.2.4	使用多线程	277
12.3	同 步	283
12.3.1	理解串行(Serialization)问题	283
12.3.2	Windows 95 同步对象	284
12.3.3	使用信号灯来同步线程	285
12.4	使用事件对象	292
12.5	需进一步研究的内容	294
第十三章	Windows 95 程序设计:Microsoft Foundation Class Library	295
13.1	对基础类库的需求	295
13.2	基础类库设计的原则	295
13.3	基础类库的关键特性	296
13.4	一切从 CObject 开始	297
13.5	重要的基库类	299
13.6	一个简单的应用程序	299
13.6.1	用 SIMPLE.CPP 建立一个窗口	300
13.6.2	运行 SIMPLE.CPP	304
13.7	简化的设计保证容易维护	304
第十四章	开发 Microsoft Foundation Class Library 应用程序	306
14.1	一个简单的应用程序和示例	306
14.1.1	理解 SFCWA.DEF 模块定义文件	308
14.1.2	理解 SFCWA.h 头文件	308
14.1.3	SFCWA.CPP 应用程序文件	309

14.1.4	运行 SFCWA	310
14.2	在窗口中画基本图形	311
14.2.1	GDI.DEF 模式定义和 gdi.h 头文件	315
14.2.2	GDI.CPP 应用程序文件	315
14.2.3	运行 GDI 应用程序	316
14.3	使用菜单和对话框生成科学波形	317
14.3.1	FOUR.h 头文件	323
14.3.2	FOURR.H 资源头文件、FOUR.RC 资源描述文件 和 FOUR.DLG 对话框描述文件	325
14.3.3	FOUR.CPP 应用程序文件	327
14.3.4	运行 FOUR	331
14.4	带有菜单和对话框的直方图	333
14.4.1	BAR.h 头文件	341
14.4.2	BAR.h 资源头文件、BAR.RC 资源描述文件 及 BAR.DLG 对话框描述文件	341
14.4.3	BAR.CPP 应用程序文件	343
14.4.4	运行 BAR	347
第十五章	多线程的程序设计	349
15.1	MFC 3.0 与多线程执行的程序	349
15.1.1	CWinThread();	349
15.1.2	CWinThread(AFX _ THREADPROC pfnThreadProc, LPVOID pParam);	350
15.1.3	void * m _ pThreadParams;	350
15.1.4	AFX _ THREADPROC m _ pfnThreadProc();	350
15.1.5	void CommonConstruct();	350
15.1.6	virtual ~CWinThread();	351
15.1.7	BOOL CreateThread(DWORD dwCreateFlags=0, UINT nStackSize=0, LPSECURITY _ ATTRIBUTES lpSecurityAttrs=NULL);	351
15.1.8	CWnd * m _ pMainWnd;	352
15.1.9	BOOL m _ bAutoDelete;	352
15.1.10	HANDLE m _ hThread;	352
15.1.11	DWORD m _ nThreadId;	352
15.1.12	int GetThreadPriority();	352
15.1.13	BOOL SetThreadPriority(hThread, nPriority);	352
15.1.14	DWORD SuspendThread(hThread);	353
15.1.15	DWORD ResumeThread(hThread);	353
15.1.16	virtual BOOL InitInstance();	353
15.1.17	virtual int Run();	353

15.1.18	virtual BOOL PreTranslateMessage(MSG * pMsg);	354
15.1.19	virtual BOOL OnIdle(LONG lCount);	354
15.1.20	virtual int ExitInstance(); (CException * e,const MSG * pMsg);	354
15.1.21	virtual LRESULT ProcessWndProcException (CException * e,const MSG * pMsg);	354
15.1.22	virtual CWnd * GetMainWnd();	354
15.1.23	void Delete();	354
15.1.24	MSG m _msgCur;	354
15.1.25	virtual BOOL PumpMessage();	354
15.1.26	virtual BOOL IsIdleMessage (MSG * pMsg);	355
15.1.27	AFXWIN.H 中的 CWinThread	355
15.1.28	线程的终止	357
15.2	一个多线程程序	357
15.3	小 结	370
第十六章	标签对话框	371
16.1	CPropertyPage	371
16.1.1	CPropertyPage(UINT nIDTemplate, UNIT nIDCaption=0);	371
16.1.2	LPCSTR lpszTemplateName,UINT nIDCaption=0);	371
16.1.3	void Cancel(ToClose());	372
16.1.4	void SetModified(BOOL bChanged = TRUE);	372
16.1.5	virtual BOOL OnSetActive();	372
16.1.6	virtual void OnOK();	372
16.1.7	virtual void OnCancel();	372
16.1.8	virtual BOOL PreTranslateMessage(MSG * pMsg);	372
16.1.9	AFXDLGS.H 中的 CPropertyPage	373
16.2	CPropertySheet	375
16.2.1	CPropertySheet(UINT nIDCaption,CWnd * pParentWnd= NULL, UINT iSelectPage=0);	375
16.2.2	BOOL Create(CWnd * pParentWnd=NULL, DWORD dwStyle=WS_ SYSMENU WS_ POPUP WS_ CAPTION WS_ MODALFRAME WS_ VISIBLE, DWORD dwExStyle = WS_ EX_ DLGMODALFRAME);	375
16.2.3	int GetPageCount() const	375
16.2.4	CPropertyPage * GetPage(int nPage)const	375
16.2.5	int DoModal();	376
16.2.6	void AddPage(CPropertyPage * pPage);	376
16.2.7	void RemovePage(PropertyPage * pPage) 和 void RemovePage(int nPage);	376

16.2.8 void EndDialog (int nEndID);	376
16.2.9 AFXDLGS.H 中的 CPropertySheet	376
16.3 PropShet 程序	379
16.4 小 结	388
附录 A 编译器选项	389
附录 B 链接器选项	392
附录 C 资源编译器选项	394

第一章 Visual C++环境

使用 Visual C++ 环境来开发 Windows 应用程序与使用 Microsoft C 编译器早期版本相比可以极大地缩短 Windows 应用程序开发周期。用 Visual C++ 2.0, 程序员可以通过使用 Visual Workbench 来加快并简化开发支持 32 位 Windows、MFC3.0、OLE2 及 ODBC 的应用程序的过程。

Visual C++ 2.0 并不意味着完全代替 Visual C++1.5。相反, 在 Windows 3.X 的使用已经到了开发或维护 16 位 Windows 应用程序不再可行之前, Visual C++ 1.5 将仍继续作为大部分 Windows 程序员 16 位 Windows 开发所选择的平台。图 1.1 显示了 Visual C++ 多种版本之间的关系以及它们的关键组成部分。

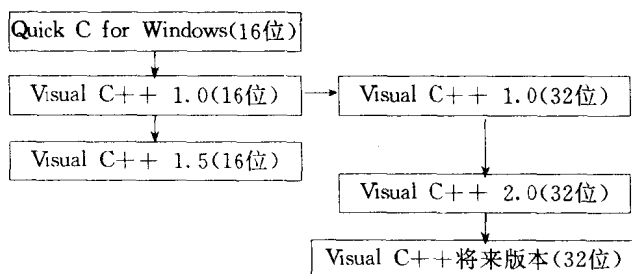


图 1.1 Visual C++ 的版本演化

图 1.1 显示了 Visual C++ 系列中的第一个成员是叫作 Quick C for Windows 的产品。把 Quick C for Windows 包括在此图中, 是因为一些 Visual C++ 的特征(包括 Windows 开发环境和创建 Shell Windows 应用程序的能力)首先在 Quick C for Windows 中引进。Quick C 不提供对 C++ 的支持。

作为一种产品, Quick C for Windows 并没有太长的生命周期, 很快就被 Visual C++ 1.0 的个人版本所代替。当下一个版本被发行时 Visual C++ 产品的两个版本(个人版本和专业版本)的创建取消。那时, Microsoft 对于 Visual C++ 产品的定价很高以至于几乎对这种低成本, 系列化的 Visual C++ 版本没有什么需要和要求。

1.1 Visual C++ 各种版本之间的区别

Visual C++ 始终保持两个版本: 16 位版本和 32 位版本。这两种产品的开发界面不一样。特别是 Visual C++ 32 位版本始终比 16 位版本提供了更好的程序员支持。

每个 Visual C++ 版本为 MFC 库提供支持。MFC 已经发展经历了三个主要版本(1.0, 2.0 和 2.5)。对于 OLE 的支持也走过了几个版本(首先是 OLE1, 然后是 OLE2, 下面将是 OLE3.0 版本)。Microsoft 已经决定去掉 OLE 名称的数字部分只称作 OLE 产品, 这意味着将不会出现 OLE3。

使用 MFC 的程序员将会喜欢一种没有痛苦的升级方法。Microsoft 非常小心不要通过改变或者从 MFC 库中去掉一些特征而明显地影响 MFC 用户。仅有的一些改变是那些改正早期 MFC 版本中发现的错误或缺陷所必需要做的东西。

Visual C++ 2.0 使资源编辑和源代码编辑器联系更加紧密。早期的 Visual C++ 版本,使用 AppStudio 来编辑资源。AppStudio 的名称是来源于 Visual C++ 的一个独立工具。在 Visual C++ 2.0 中,资源在 Visual C++ 的开发环境中直接编辑。这意味着在 Visual C++ 和 AppStudio 之间没有切换(即不得不记住的保存文件)。图 1.2 显示了已装入了 Gddemo 工程文件且对话框已打开的 Visual C++。

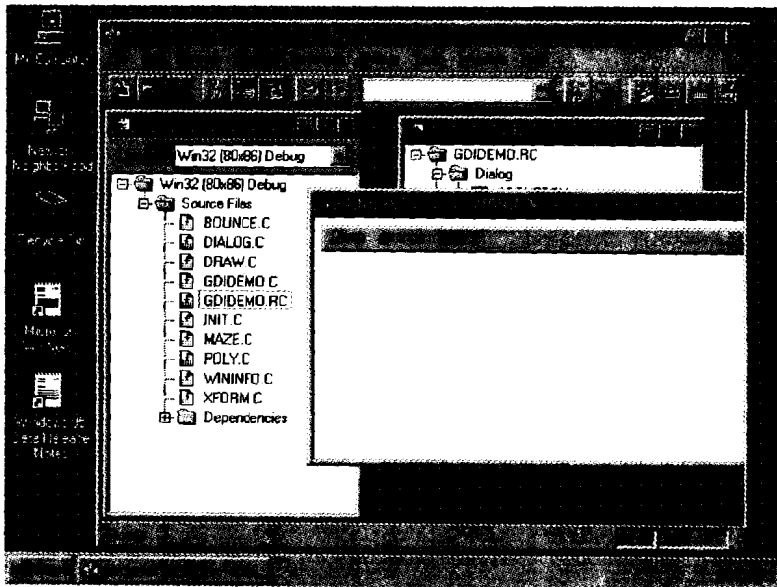


图 1.2 有对话框资源打开 Visual C++ 2.0

用户可以使用从 Microsoft 中获得的附加 RISC 编译器和为 Windows NT 所支持的平台开发应用程序,包括 RISC(精减指令集计算机)平台。

对于创建不得不移植到不同的非 Windows 平台的应用程序开发人员,Visual C++ 2.0 支持对于来自于 Apple 计算机公司的 Macintosh 计算机的交叉平台开发。使用 Visual C++ 2.0,Windows 应用程序可以很容易地移植到 68XXX 平台。

在过去,当程序员开发一个新的 Windows 应用程序时,要花几个月的时间来完成外壳。过去要花几个月的时间和精力所作的事情现在只需在几分钟内用一些键击来完成。虽然没有确定使用 AppWizard 开发 Windows 应用程序要花多长时间,但是时间不会长。对创建第一个 Windows 程序的基本外壳需整四个月时间。以前开发应用程序,使用了一些相当简陋的工具:一个简单的对话框编辑器(可用来给控件定位和确定大小,除此外无其它功能),一个简单的图标编辑器,和一个基于 DOS 编辑器命令行。直到 Windows 3. X 的发行,不可能真正地在运行 Windows 的同时来开发 Windows 程序——没有足够的内存运行编译器和链接器。最好的开发系统是 Windows 和 OS/2 机器的联网,其中编辑和程序的建立在 OS/2 机器

上进行而实际测试在运行 Windows 的其它系统上进行。

现在,使用 Windows NT 和 Windows 95 的保护模式,同 Windows 3.0 之前的版本相比,开发工具可以访问更多的内存。另外,Windows 应用程序(包括 Visual C++ 的工具)在保护模式中运行,它们同早期的实模式开发工具相比已经访问了更多的内存。

得益于使用更多内存这一能力的开发工具的典型例子是 Visual Workbench 程序。最初创建对话框的程序 DLGEDIT 仅有基本功能:只能创建对话框。要创建其它的资源,比如光标和图标,就不得不使用其它的应用程序。有了 Visual C++ 的 Visual Workbench 可以允许用户创建在 Windows 应用程序中需要的所有资源,编辑源文件,管理工程文件,通过调用编译器和链接器创建最后的程序。

今天,由于有了 Visual C++,用户可以在几分钟内开发出 Windows 程序的外壳。可以创建对话框,加上必要的 C++ 类,绘制图标,定义菜单,建立及测试应用程序,这些只需花去过去所需要的几年时间中的一小部分就可完成。

1.2 小 结

本章介绍了 Visual C++ 2.0,它为 Windows 程序员提供了一些很强的功能。本章讲述了 Visual C++ 2.0 和早期 Visual C++ 版本之间的差别。包括了下面的主题:

- ① Visual C++ 产品的起源;
- ② Microsoft Foundation Class 库的升级;
- ③ OLE 的升级。