

学习和使用

Visual C++(上册)

何 亮 朱志强 等编著

潘金贵 顾铁成 审 校



同济大学出版社

内 容 提 要

本书是《学习和使用 Visual C++》丛书的上册,主要介绍了 Visual C++集成环境和 OOP 概念。内容包括 Visual C++集成环境、程序开发工具、面向对象程序设计基础、C++与 ANSI 标准 C 的区别、标准 C++的 I/O类、类对象的创建、定义对象的操作、继承、虚拟函数和多态性、类模板、函数模板、异常处理、如何在 C++中使用 C 库、如何在 C++中建立类库、如何使用 Microsoft 基础类库、如何用 C++建立 MS-DOS 应用程序等。对 C++(尤其是 Visual C++)程序员而言,本书是一本很实用的技术指导书籍。本书可作为程序设计方法、通用程序设计技术、C++、Visual C++等方面的培训教材。

责任编辑 王建中

封面设计 李志云

学习和使用 Visual C++

(上 册)

何 亮 朱志强 等编

潘金贵 顾铁成 审校

同济大学出版社出版

(上海四平路 1239 号)

新华书店上海发行所发行

同济大学印刷厂印刷

开本: 787×1092 1/16 印张: 23 字数: 580 千字

1997 年 1 月第 1 版 1997 年 1 月第 1 次印刷

印数: 1—4000 定价: 37.00 元

ISBN7-5608-1692-4/TP·177

前 言

随着微软公司 Windows 95 的问世,Windows 在微机软件业的主流地位进一步巩固,作为 Windwos 环境下的优秀、正宗 C++ 编译器的 Visual C++ 进一步成为软件开发人员强有力的工具。Visual C++ 是汇集微软公司技术精华的主流产品,它不仅全面地贯彻了面向对象技术,而且在编译优化技术方面较其他同类产品具有明显的优势。《学习和使用 Visual C++》一书共分上、中、下三册。这套书从各个方面说明了应如何在 Visual C++ 环境下以 OOP 的概念设计 DOS 和 Windows 应用程序。

在 Windows 如此火热的时候,任何一个希望在微机平台上开发自己的软件产品的单位和个人都无法回避 Windows 环境。作为一个成熟的软件工程师,面对铺天盖地的 Word,Excel,PowerPoint,FoxPro,Visual Basic 等应用软件,有很多理由需要熟悉 Visual C++。笔者曾与美国微软公司一位资深业务主管在 Windows 95 研讨会上曾讨论过这个问题,他从美国软件产业的统计分析得出的结论是,附加在 Windwos 环境上的第三方应用软件的增值产品产值在过去五年中年平均递增约 70%。由此可见,借助 Visual C++ 开发工具,结合自己的专业技术,开发出各种专用应用软件的动态链接库,就能为广大用户提供各个方面的支持,其商业前景更是无可估量的。

Visual C++ 是一个面向对象、界面友好的 DOS 和 Windows 程序开发环境。在这个集成环境中开发应用程序时,不一定要手工设计用户界面,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。此后,程序员就可利用基于 Windows 的 C++ 编辑器,借助 AppWizard 建立面向对象的应用程序。

除了 AppWizard 外,Visual C++ 还包含 C++ 应用程序生成器、基于 Windows 的编辑器、基于 Windows 的面向图形的编程工具、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 库以及预编译的头文件和源文件。当然,类库丰富更是 Visual C++ 的最大特色。

目前市面上虽然有不少 Visual C++ 方面的书籍,但学习和使用 Visual C++ 的人常常会觉得力不从心,主要问题是:

(1) 对 Visual C++ 缺乏一致的认识,遇到问题总是缺这少那,缺乏一个值得信赖的良师益友。

(2) 只了解 OOP 的直观语法,而对于 OOP 有何好处、为什么 OOP 可取代结构化语言并成为未来程序设计风格的主流,都不太明白。

(3) 无法理解 OOP 对未来程序的管理与维护究竟有何重要性。

(4) 在 Visual C++ 环境下设计 OOP 程序时,对 Visual C++ 提供的资源不太清楚。

(5) 不知道如何用 Visual C++ 和 Windows 解决实际问题。

针对这些问题,这套书从各个方面介绍了 Visual C++ 集成环境、OOP 概念、类库以及基于 DOS 和 Windows 的 C++ 应用程序开发方法,并提供了丰富的类模板和大量的应用程序实例。

本套书的内容基本属于中等难度,适用于初、中级读者。不熟悉 OOP 技术和 C++ 语言的读者可参阅上册的内容,上册主要介绍 Visual C++ 集成环境和 OOP 概念;中册介绍经常从事应用程序开发的读者比较关心的 Visual C++ 类库和通用类的构造方法,并从我们所熟悉的数据结构着手,讨论如何设计和实现自己的通用类,这为进行大型程序开发的用户提供了设计开发环境的手段;下册侧重函数的介绍,包括文件输入输出、数据处理、进程控制及内存管理等内容。

参与上册编写的人员有:何亮、朱志强、覃牧、何国辉、李华清、李杨、魏志国、李蕾、李建、刘心海、刘崇福、李纪鸿、刘石华、赵越、陆静宜、李伯雄、陈楚南、梁友国、陈佳海。全书先由吴佳教授和陈忠敏研究员作了初审,最后由潘金贵和顾铁成审校定稿。此外,朱闽虹为本书的编排付出了辛勤的劳动。在此对以上同志深表感谢。

限于水平和时间,书中的错误和不妥之处,敬请读者批评指正。

编 者

目 录

第一章 Microsoft Visual C++ 集成环境	1
1.1 Microsoft Visual C++ 简介	1
1.2 Microsoft Visual C++ 安装指南	3
1.3 如何构造 DOS MFC 库文件	7
1.4 如何使用 Visual Workbench	7
1.4.1 Visual Workbench 简介	8
1.4.2 Visual Workbench 命令行参数简介	9
1.4.3 Visual Workbench 的帮助系统	9
1.4.4 利用键盘和鼠标操纵 Visual Workbench 的菜单	11
1.5 如何编辑一个程序	16
1.6 如何编译、链接和运行一个程序	18
1.6.1 如何设置工程文件选项	18
1.6.2 Visual C++ 的调试功能	21
1.6.3 Visual Workbench 的浏览功能	22
1.7 如何使用 CL	22
1.7.1 如何用 CL 进行编译和链接	23
1.7.2 如何使用 CL 环境变量	24
1.7.3 CL 选项简介	24
1.8 一个 Visual Workbench 实例	30
1.9 小结	30
第二章 Visual C++ 程序开发工具概述	32
2.1 LINK 链接器	32
2.1.1 如何使用 LINK	33
2.1.2 LINK 的文件名选项	33
2.1.3 覆盖技术	34
2.2 LINK 选项	35
2.3 库文件管理程序 LIB	38
2.3.1 如何使用 LIB	38
2.3.2 LIB 选项及命令的用法	39
2.4 程序维护实用程序 NMAKE	39
2.4.1 建立 MAKE 文件	40
2.4.2 运行 NMAKE	43
2.5 CodeView 调试程序	43
2.5.1 如何使用 CodeView	44
2.5.2 如何启动 CodeView	45
2.5.3 程序调试的方法	45
第三章 ANSI 标准 C 简介	52
3.1 C 程序的结构	52
3.2 ANSI C 转义序列和三字符序列	55

3.3	ANSI C 预处理器伪指令	56
3.3.1	包含文件	56
3.3.2	定义宏	57
3.3.3	条件伪指令	58
3.3.4	其他伪指令	59
3.4	变量的声明和定义	59
3.4.1	基本变量类型	60
3.4.2	枚举类型	60
3.5	结构、联合及位字段	61
3.5.1	数组	62
3.5.2	指针	62
3.5.3	类型的定义	63
3.5.4	类型修饰符 const 和 volatile	63
3.6	表达式	64
3.7	语句	67
3.7.1	break 语句	67
3.7.2	case 语句	67
3.7.3	复合语句或块	67
3.7.4	continue 语句	67
3.7.5	default 标记	68
3.7.6	do 语句	68
3.7.7	表达式语句	68
3.7.8	for 语句	68
3.7.9	goto 语句	69
3.7.10	if 语句	69
3.7.11	if - else 语句	69
3.7.12	null 语句	69
3.7.13	return 语句	70
3.7.14	switch 语句	70
3.7.15	while 语句	70
3.8	函数	71
3.8.1	函数原型	71
3.8.2	void 类型	71
3.8.3	带有可变个数参数的函数	72
3.9	ANSI C 库	72
3.10	小结	73
第四章	Visual C++ 对标准 C 的扩展	74
4.1	Visual C++ 特有的关键字	74
4.2	内存模式的定制	76
4.3	全局变量及预处理器宏	79
4.4	预编译指令	82
4.5	基本数据类型的大小和容量	83
4.6	小结	84
第五章	面向对象程序设计基础	85
5.1	什么是面向对象的程序设计	85

5.2	面向对象程序设计基本概念	92
5.2.1	数据抽象	92
5.2.2	继承	93
5.2.3	多态性	94
5.3	C 语言中的面向对象程序设计	94
5.3.1	在 C 语言中定义对象	94
5.3.2	实现几何形状	97
5.3.3	使用图形	103
5.3.4	增加一个新图形对象	104
5.3.5	用 C 语言实现 OOP 的问题	107
5.4	小结	107
第六章	C++ 和面向对象的程序设计	108
6.1	C++ 与面向对象的程序设计	108
6.1.1	C++ 中的数据抽象	108
6.1.2	C++ 类的继承	112
6.1.3	多态和动态链接	112
6.2	用 C++ 设计几何图形	114
6.2.1	图形类	114
6.2.2	添加新的图形类	117
6.2.3	运行时创建对象	119
6.3	小结	119
第七章	C++ 与 ANSI 标准 C 的区别	120
7.1	C++ 的特性	120
7.1.1	C++ 函数的新特性	120
7.1.2	C++ 和 C 的区别	126
7.2	小结	129
第八章	用于标准 I/O 的 C++ 类	130
8.1	C++ 中的 I/O 库	130
8.2	C++ 中的流 I/O	130
8.2.1	使用 iostream	131
8.2.2	使用操作符	132
8.2.3	使用操作符完成格式化 I/O	133
8.2.4	控制浮点格式	135
8.2.5	重载 <<	135
8.2.6	iostream 类的层次	136
8.3	文件 I/O	138
8.3.1	简单文件 I/O	138
8.3.2	文件定位	141
8.4	字符串 I/O	142
8.4.1	向一个串写入	143
8.4.2	从一个串读取	143
8.5	小结	144
第九章	创建类的对象	145
9.1	对象与类	145
9.1.1	用户定义的数据结构	145

9.1.2	对类成员的访问控制	146
9.1.3	public 函数如何返回 private 值	147
9.1.4	成员函数	148
9.2	类的实现	149
9.2.1	头文件描述界面	149
9.2.2	从界面分离操作	151
9.3	类的使用	152
9.3.1	动态创建对象	153
9.3.2	在自由存储区中分配对象数组	154
9.3.3	调用成员函数	155
9.3.4	使用 static 成员变量	155
9.3.5	初始化 static 成员变量	158
9.3.6	使用静态成员函数	159
9.3.7	使用指向类成员的指针	160
9.4	小结	162
第十章	定义对象的操作	163
10.1	参数和返回值	163
10.1.1	理解指针和引用	163
10.1.2	值传递与引用传递的比较	164
10.1.3	返回一个引用	165
10.1.4	使用引用的准则	166
10.2	对象的创建和释放	166
10.2.1	类 String 的构造函数和析构函数	167
10.2.2	缺省构造函数	168
10.2.3	拷贝构造函数	169
10.2.4	提供一个拷贝构造函数的时机	171
10.2.5	成员初始化表	172
10.2.6	利用构造函数和析构函数的副作用	174
10.3	定义函数和操作符	176
10.3.1	指针 this	176
10.3.2	操作符作为函数	177
10.3.3	为类 String 定义 operator+	179
10.3.4	验证串是否相等	180
10.3.5	访问并改变串中的某个字符	181
10.3.6	定义类型转换操作符	181
10.3.7	为类 String 定义赋值操作符	181
10.3.8	为什么 operator= 返回一个引用	182
10.3.9	装载输入和输出操作符	182
10.3.10	装载操作符 new 和 delete	184
10.3.11	使用友元类	185
10.3.12	把文件当作数组来使用	185
10.4	小结	188
第十一章	在 C++ 中使用继承	190
11.1	派生类	190
11.1.1	继承和“is a”关系	190

11.1.2	继承和类扩展	190
11.1.3	派生类的语法	191
11.1.4	访问基类	192
11.1.5	使用继承来建造子串类	193
11.1.6	派生类的其他情况	200
11.1.7	多重继承	201
11.1.8	iostream 与多重继承	202
11.1.9	虚拟基类	202
11.2	使用继承	204
11.2.1	链表	205
11.2.2	single-link 类	206
11.2.3	双向链表	213
11.2.4	String 对象的队列	218
11.3	小结	220
第十二章	虚拟函数和多态性	221
12.1	动态链接	221
12.1.1	静态链接	221
12.1.2	通过指针调用函数	222
12.2	虚拟函数	223
12.2.1	纯虚拟函数	223
12.2.2	虚拟函数的具体实现	224
12.2.3	通过虚拟函数的动态链接	224
12.2.4	使用多态	226
12.2.5	隐含的类的多态用法	227
12.3	小结	229
第十三章	C++ 的高级技术	231
13.1	函数和类模板	231
13.1.1	整型和浮点型的栈	231
13.1.2	类模板	232
13.1.3	函数模板	233
13.1.4	成员函数模板	233
13.1.5	模板的优越性	233
13.2	异常处理	234
13.2.1	异常处理的优点	234
13.2.2	setjmp 和 longjmp 的问题	235
13.2.3	C++ 建议的异常处理原理	236
13.3	小结	238
第十四章	如何在 C++ 中使用 C 库	239
14.1	C 与 C++ 的链接	239
14.1.1	类型安全链接	239
14.1.2	函数名编码的影响	239
14.1.3	C 的链接命令	240
14.1.4	与其他语言的链接	242
14.2	如何使用 ANSI 标准 C 库	243
14.2.1	ANSI C 库功能概述	243

14.2.2	标准 I/O 函数	243
14.2.3	进程控制函数	244
14.2.4	内存分配技术	246
14.2.5	可变长度参数表	247
14.2.6	数据转换函数	248
14.2.7	数学函数	249
14.2.8	字符分类	250
14.2.9	字符串和缓冲区操作	250
14.2.10	C 和 C++ 中的字符串	250
14.2.11	查找和分类	252
14.2.12	日期和时间	255
14.2.13	DateTime 类	256
14.3	编译器指定的库	258
14.4	小结	258
第十五章	在 C++ 中建立类库	259
15.1	在 C++ 中建立类库	259
15.1.1	如何组织 C++ 类	259
15.1.2	单继承下的继承层次	259
15.1.3	类之间的客户-服务器关系	262
15.2	C++ 类的公共接口	266
15.2.1	缺省构造函数和拷贝构造函数	266
15.2.2	拷贝对象	267
15.2.3	析构函数	267
15.2.4	赋值操作符	267
15.2.5	输入输出函数	268
15.3	小结	268
第十六章	使用 Microsoft Foundation Class 库	269
16.1	用 Microsoft Foundation Class 设计 Microsoft Windows 程序	269
16.1.1	模块显示控制(MVC)文档	269
16.1.2	使用 MFC 的一个 Windows 应用程序	271
16.2	Microsoft Foundation Class 的层次	280
16.2.1	类层次的分解	280
16.2.2	Microsoft Foundation Class 库中的归档和异常处理	282
16.3	小结	284
第十七章	用 C++ 建立 MS-DOS 应用程序	285
17.1	Forms 软件包简介	285
17.2	表格的存储和检索技术	285
17.3	表格的组成	289
17.3.1	FormBackground 类	289
17.3.2	FieldList 类	292
17.3.3	Field 类	296
17.4	显示表格	298
17.4.1	TextGraphics 类的层次	299
17.4.2	基于字符的图形	303
17.4.3	OutputDevice 类	307

17.5	Form 类	309
17.6	创建表格	314
17.6.1	定义表格	314
17.6.2	建立 FORMDEF	318
17.6.3	运行 FORMDEF	318
17.6.4	创建表格数据	319
17.6.5	建立 FORMDAT	320
17.7	填写表格	320
17.7.1	FormView 类	320
17.7.2	EventHandler 类	325
17.7.3	FORMFILL 程序实例	327
17.7.4	建立 FORMFILL	328
17.8	小结	328
第十八章	用 AppWizard 创建 C++ Windows 应用程序	329
18.1	如何启动 AppWizard	330
18.1.1	为应用程序定制选项	331
18.1.2	AppWizard 创建的类	334
18.1.3	AppWizard 创建了什么	335
18.2	如何使用表格文档模板	337
18.3	如何创建 FormFill 的模板	338
18.3.1	如何创建表格的布局	339
18.3.2	如何创建视图类	341
18.3.3	表格变量的使用	342
18.4	如何设置文档模板	346
18.5	如何覆盖成员函数	348
18.5.1	如何覆盖 OnUpdate 成员	349
18.5.2	如何修改其他成员	349
18.6	如何提供多文档模板	351
18.7	完成 FormFill 的操作	352
18.7.1	增加文件输入/输出功能	352
18.7.2	选取一个给定的记录	352
18.7.3	其他功能说明	352
18.8	小结	353

第一章 Microsoft Visual C++ 集成环境

本书的主要目的是帮助 C 程序员学习用 Microsoft Visual C++ 进行 Windows 程序设计、面向对象的程序设计(OOP)和 C++ 程序设计。同时,本书还可以作为 C++ 类(Microsoft Foundation Classes V2.0)和 Microsoft Visual C++ 的 C 函数库的参考指南。当然,在学习 OOP 和 Microsoft Visual C++ 库之前,用户应了解 Microsoft Visual C++ 提供了哪些工具来帮助用户执行必需的编辑、编译、链接和调试等程序设计任务。为了帮助用户熟悉 Microsoft Visual C++ 环境,本章对 Microsoft Visual C++ 产品进行概述,包括其组成部分、如何安装以及如何在运行 Windows 的 PC 机上使用。Visual C++ 从本质上讲是一个 Windows 应用程序。尽管可以在一个 DOS 窗口中进行编译,但 Visual C++ 最好的程序设计界面是 Visual Workbench。

Visual C++ 有两个版本:

(1) 专业版本,它包括一个更高级的优化编译器、更广泛的文档,并能设计基于 DOS 的应用程序。

(2) 标准版本,其费用较低,没有那么高级的编辑器,文档较少,且不能设计基于 DOS 的应用程序。

本章还将介绍称为 Visual Workbench 的交互式开发环境,以及编译和链接的 DOS 命令行接口 CL。

1.1 Microsoft Visual C++ 简介

Microsoft Visual C++ 是多个产品的集成。因此,随软件带了许多手册。Visual C++ 有三种版本:标准版本、专业版本和 C7 升级版。对于这三种版本,都有相应的文档。

作为主产品的专业版 Visual C++ 具有比标准版更多、功能更强的特性。而标准版 Visual C++ 的功能要弱一些(针对非专业软件设计人员)。

Visual C++ 提供完全的面向 C 和 C++ 的 Windows 程序设计环境,并且包含所有的支持工具及编写 Microsoft Windows 和 DOS 应用程序所需的库。作为一名程序设计师,用户一般先用如下工具:

(1) Visual Workbench:用于编辑源文件,并与包括 C 和 C++ 编译器在内的许多实用支持工具接口,Visual Workbench 允许用户建立应用程序,并对其进行调试,而不必退出 Visual Workbench。通常,Visual Workbench 将帮助用户使用其他开发工具。例如,运行编译器、链接器和资源编译器以建立可执行文件。

(2) Application Wizard:通常称为 AppWizard,它是用于建立初始 Windows 应用程序的程序开发工具。

(3) Class Wizard:用于管理 C++ 类的实用工具(用 App Wizard 进行程序开发时使用)。

(4) Application Studio:通常称为 App Studio,用于编辑如菜单、对话框、位图、图标和光标等的 Windows 应用程序资源。

(5) LINK:用于将多个目标模块结合成一个可执行文件的基于 DOS 的链接器。

(6) LIB:用户在一个文件中管理目标模块集合的基于 DOS 的库管理程序。

(7) NMAKE:用于自动程序开发的基于 DOS 的制作实用程序。

(8) CV 和 CVW:分别用于 DOS 和 Windows 的 CodeView 调试器。

也有一些实用工具(都是基于 DOS 的)用户通常不使用,但是一旦用户想使用,都能很容易地得到,这些实用工具包括:

(1) EXEHDR:用来显示和修改包含在 EXE 文件头部中的信息的实用工具。

(2) CVPACK:用来压缩包含 CodeView 调试器所需信息的可执行文件。

(3) HELPMAKE:用于将文本文件(具有特殊格式)转换为数据库的实用工具,这些数据库可被许多像 Visual Workbench 的 Microsoft 应用程序使用,用来显示联机帮助信息。

(4) IMPLIB:用来根据一个或多个动态链接库创建一个特殊类型库的移入库。移入库用于 Microsoft Windows 程序。

(5) MSD:让用户查看有关自己 PC 机硬件和软件配置等重要信息的 Microsoft 诊断工具。

另外,Microsoft Visual C++ 包含一系列基于 Windows 的实用工具。这些 Windows 实用工具是:

(1) SPY:允许用户检查将发往一个窗口的消息和消息参数的程序。

(2) DDESPY:允许用户查看一个 DDE 服务程序和一个客户应用程序之间 DDE 交换的程序。

(3) HEAPWALKER:允许用户检查 Windows 内存,以查看存在哪些内存对象、每个运行应用程序的内存对象的地址,及 Windows 内存是怎么被利用的实用工具。

(4) STRESSAPP:允许用户强制 Windows 分配资源,直到没有资源可分配给应用程序的程序,这就允许用户在某些资源不再可以利用时,查看应用程序出了什么问题。

(5) ZOOMIN:小应用程序,它允许用户放大屏幕的一小部分。

(6) DEBUGWIN:实用工具,它允许用户查看 Windows 产生的调式版本信息。

(7) MFC TRACE OPTIONS:允许用户查看 MFC 使用的调试选项的实用工具。

(8) HOTSPOT EDITOR:用户可用来建立或编辑超图形的程序,超图形是一幅有一个或多个用户可在其上引起动作的热点的位图。

至于如何使用 Microsoft Visual C++,则要根据用户的需要决定。可以采用以下方式之一编译和链接自己的应用程序:

(1) 从 Visual Workbench 中调用编译器和链接器,可使用 Project, Build 和 Project, Rebuild All 菜单选项。

(2) 从 DOS 命令行上,运行编译器和链接器。

(3) 使用 NMAKE 使编译和链接过程自动化。

使用 Visual Workbench 是 Visual C++ 开发应用程序的最效的方法。也可以在 DOS 下运行编译器。但是,这样效率不高,因为用户必须先运行一个编辑器编辑源代码,然后再编译(和链接)源代码。

总之,用户会发现 Visual Workbench 的编辑器是一种高效的程序设计工具,其编辑器在有限范围内可以配置用户的链接操作。使用 Visual Workbench 的工具条,用户可以直接访问编译器和链接器。Visual Workbench 是一个具有相当标准的 Windows MDI 用户界面的 Windows 应用程序。当 Windows 以增强模式(80386 保护模式)运行时,用户可以用调试器运行和调试应用程序,该调试器是 Visual Workbench 集成环境的一部分。在 Visual Workbench 中,可以完成整个程序开发过程,即开发和编辑源文件,建立 Windows 资源,编译、链接及运行程序,调试程序。所有这些操作都不必退出 Visual Workbench。

由于大量软件产品都包含许多源程序模块,所以使用 Visual Workbench 非常必要,Visual Workbench 将建立一个工程文件(与 NMAKE 的 MAK 文件类似),它包含 Visual Workbench 编译和链接必要模块所需要的所有伪指令。有了工程文件,当用户编辑了一个或多个源文件后,选择 Visual Workbench 的 Project Build 图标将编译所有受影响的文件。例如,修改一个头文件就必须编译所有包含该头文件的源文件。

正如用户将从本章有关 Visual Workbench 的描述了解到的,工程文件描述了源文件、目标文件和可执行文件之间的相互依赖关系。Visual Workbench 使用依赖关系知识来编译适当文件,并建立新的可执行文件。

用 Options 菜单可为编译和链接声明各种选项,不论某个工程是第一次被建立还是已经建立。在本章后一部分中,我们将要说明如何从 Visual Workbench 设置编译和链接选项。在本章末尾,要介绍如何建立 Visual Workbench 工程。1.2 节首先描述如何安装 Microsoft Visual C++。1.3 节概述 Visual Workbench 和 CL,CL 是供编译和链接用户的命令接口。Microsoft Visual C++ 所包括的许多其他工具将在第二章中介绍。

注意,Microsoft Visual C++ 需要以增强模式运行的 Windows,80386 或 80486 系统,至少 4MB 内存才能运行。其编译器是一个 32 位 Windows/DOS 应用程序,必须运行在保护模式的 Intel 80386 或 80486 微处理器上。

尽管 Microsoft Visual C++ 编译器、链接器及其他程序通用的实用程序可运行在 DOS 窗口中,但一旦用户开始使用 Visual Workbench 后,一般不会经常把编译器作为 DOS 应用程序来使用。

1.2 Microsoft Visual C++ 安装指南

要安装 Microsoft Visual C++,只须直接运行发行盘中的 SETUP 程序。SETUP 将发行盘中的文件恢复并拷贝到用户所指定的硬盘上。用户有权选择希望 Microsoft Visual C++ 安装时使用的驱动器和目录。在启动 Windows 后,安装 Visual C++ 基本步骤如下:

(1) 将第一张盘(#1)插入到适当的驱动器中(A:或 B:),根据软件大小及系统配置而定。

(2) 从 Program Manager(程序管理器)中选择 File 菜单的 Run 选项,在命令行方框中输入如下程序名(使用正确的驱动器字母):

A:SETUP

(3) 一旦 SETUP 开始运行(见图 1.1),阅读并响应屏幕显示。任何时刻按下屏幕上的 HELP 按钮(见图 1.2)都可调出联机帮助信息。

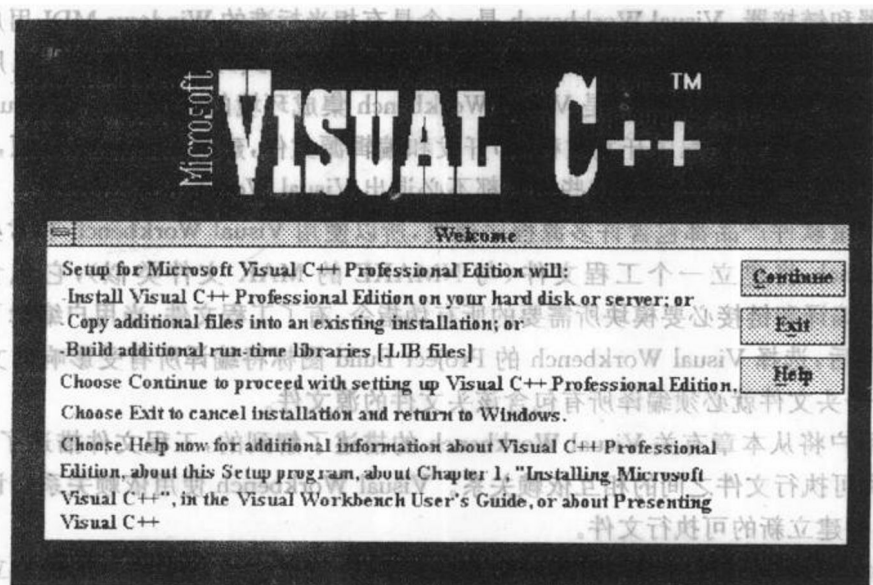


图 1.1 Visual C++ SETUP 程序的初始屏幕

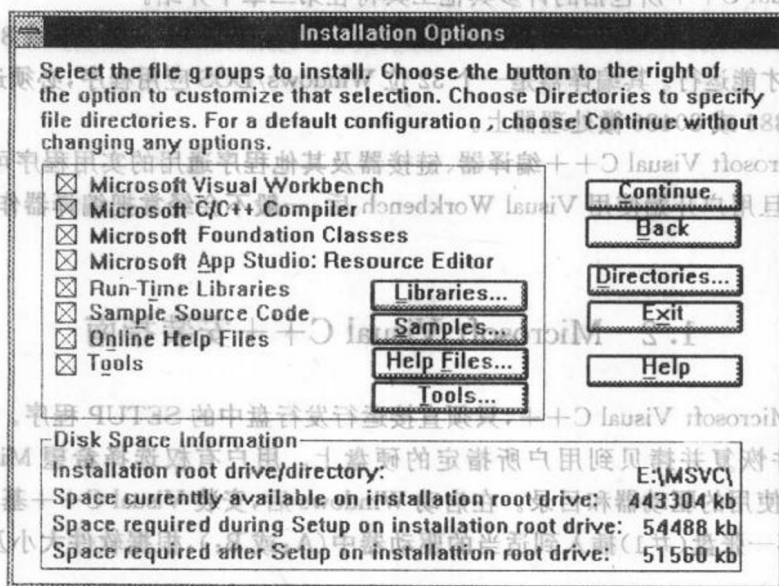


图 1.2 Visual C++ SETUP 程序的初始化选项

通过响应 SETUP 的屏幕所提的问题,用户能完成如下操作:

1) 选择希望将 Visual C++ 安装在其中的驱动器号和目录。注意 Visual C++ 需要多少硬盘空间。在缺省安装情况下,需要 52MB 硬盘空间。如果用户安装了任何 Visual C++

的可选部分,实际可能需要更多的硬盘空间。

2) 为用户需要安装的库选择 memory models 复选框,如图 1.3 所示。如果用户不熟悉内存模式,可以选择一种安全的选择,即安装除 Compact 模式外所有的内存模式,专业程序员一般不使用 Compact 模式。

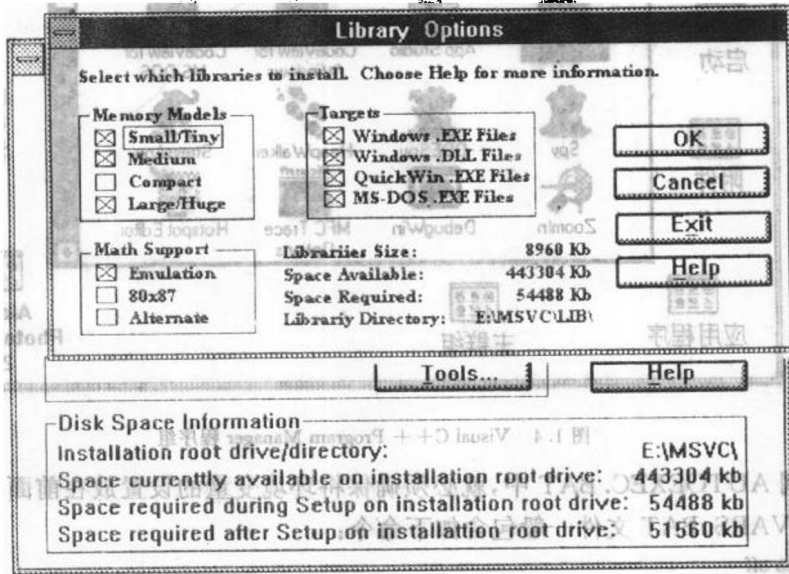


图 1.3 Visual C++ SETUP 程序的库选项

3) 决定是否希望将例程拷到硬盘上。一般来说,阅读这些样本程序能学到颇有价值的各种程序设计技术。

Microsoft Visual C++ 也建议用户修改 CONFIG. SYS 文件,在开始安装时,它建议用户将 BUFFERS 参数设为 30。另外,安装程序建议用户运行磁盘高速缓存软件(如 smartDrv),使高速缓存(只读的)对用户安装软件所使用的软盘为 ON 状态。如果某个产品有几十张软盘,那么任何加速安装过程的方法都是很重要的。将高速缓存设为 ON 和 OFF 状态的区别可能是安装时间相差数倍。

当用户设置了自己需要的配置后(而且用户同意 SETUP 提供的缺省配置选项),就应选择 Continue 按钮。这样就通知了 SETUP 为用户所选的内存模式安装编译器和建立库文件。SETUP 也创建一个程序组(见图 1.4),从而使用户能由此选择 Visual C++ 实用程序。还有许多图标(在 Visual C++ 的 Program Manager 的程序组中)可供用户使用 Visual C++ 所带的帮助文件。

为了能从 DOS 命令行运行编译器和其他实用程序,用户必须使用批处理文件 MSVC-VARS. BAT。该文件由 SETUP 创建,SETUP 向 MSVCVARS. BAT 中加入针对所需环境变量的定义。用户可从一个 DOS 窗口中运行该批处理文件。MSVCVARS. BAT 将为用户设置必需的环境变量。如果用户愿意,可以通过直接包含 MSVCVARS. BAT 中的各行或调用 MSVCVARS. BAT 来将该文件包含到 AUTOEXEC. BAT 中。如果用户将 MSVCVARS.



图 1.4 Visual C++ Program Manager 程序组

BAT 包含到 AUTOEXEC. BAT 中,就必须确保将环境变量的设置放在前面。

MSVCVARS. BAT 文件一般包含如下命令:

```
@echo off
set TOOLROOTDIR=D:\MSVC
set PATH=D:\MSVC\BIN;%PATH%
set INCLUDE=D:\MSVC\INCLUDE;D:\MSVC\MFC\INCLUDE;%INCLUDE%
set LIB=D:\MSVC\LIB;D:\MSVC\LIB;%LIB%
```

用户应该知道,像 PATH 这样的环境变量一般是一任意字符串的名字,而且是用 SET 命令定义的。

在 DOS 提示符下输入 SET 将显示当前环境变量的设置,如

```
SET PATH=C:\;C:\DOS6
```

定义 PATH 为字符串 C:\;C:\DOS6,从而代替了它的先前定义。下面是 SETUP 定义的环境变量:

TOOLROOTDIR	存放 Visual C++ 工具的目录,例如: SET TOOLROOTDIR=D:\MSVC
PATH	当输入一个不是 DOS 内部命令的命令后, DOS 搜索的目录 路径。如: SET PATH=D:\MSVC\BIN;%PATH% 请注意: 原路径是如何将 %PATH% 环境变量加到新路径变 量后的
INCLUDE	由分号分隔的目录名,C 和 C++ 包含文件(一般带.H 扩 展名)存于其中。例如: SET INCLUDE=D:\MSVC\INCLUDE;D:\MSVC\MFC\ INCLUDE;%INCLUDE%