

GOTOP

北京科海培训中心

Visual C++ 6

程序设计经典

林俊杰 编著
谷保山 审校

科学出版社



GOTOP

北京科海培训中心

Visual C++ 6 程序设计经典

林俊杰 编著

谷保山 审校

科学出版社

1999

内 容 简 介

本书以循序渐进的方式,深入浅出地讲述了使用 Visual C++ 6.0、MFC 开发 Windows95/98/NT 应用程序的方法。

全书共分 19 章,内容包括:MFC 程序设计概念,Document/View 程序设计,如何使用 ActiveX 控件,DC 与 Windows 绘图程序设计,MFC 的消息处理,对话框与多页对话框设计,多种字型输出,AppWizard 与 ClassWizard 的操作方法,窗口类型,多线程程序设计与 IPC 应用,自行建立 SDI 与 MDI 应用程序,使用浮动式工具条、状态条及分割式窗口,各种控件与通用控件,位图与 CDib 实例,文件打印及各类成员函数的详细说明与示例。

本书内容全面、讲述清晰明了、实例丰富,针对 Visual C++ 初学者是一本很好的入门教材,对应用开发人员来说不仅具有参考价值,还可作为随机手册。

版 权 声 明

本书为台湾碁峯资讯股份有限公司独家授权的中文简体字版本。本书专有出版权属北京科海培训中心与科学出版社所有。在没有得到本书原版出版者和本书出版者的书面许可时,任何单位和个人不得擅自摘抄、复制本书的一部分或全部内容,不得以任何形式(包括资料和出版物)进行传播。

本书原版权属于碁峯资讯股份有限公司。

版权所有,侵权必究

图书在版编目(CIP)数据

Visual C++ 6 程序设计经典/林俊杰编著. —北京:

科学出版社,1999.11

ISBN 7-03-008002-5

I. V… II. 林… III. C 语言-程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(1999)第 65142 号

图字:01-1999-2093

JS116/03

科 学 出 版 社 出 版

北京东黄城根北街 16 号

邮政编码:100717

北京门头沟胶印厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1999 年 11 月第 一 版

开本:787×1092 1/16

1999 年 11 月第一次印刷

印张:33 3/4

印数:1—5 000

字数:820 800

定价:52.00 元

目 录

第 1 章 综述	1
1.1 本书结构	1
1.2 本书约定	2
1.3 特殊说明	2
第 2 章 C++重点回顾	3
2.1 类、对象、实例	3
2.2 构造函数 Constructor	3
2.3 匿名实例 Nameless Instance	4
2.4 虚函数	5
2.5 异常处理	7
第 3 章 基本概念与简单的MFC类	10
3.1 什么是 Win32 API	10
3.2 主控台模式应用程序 (Win32 Console Mode)	11
3.3 基本数据类型	12
3.3.1 基本的数据结构	13
3.3.2 使用TCHAR设计支持Unicode和SBCS的通用源程序	14
3.4 使用MFC类库	16
3.4.1 MFC的根CObject	16
3.4.2 功能强大的字符串类CString	16
3.4.3 CFile家族	18
3.4.4 CException 异常类家族	21
3.4.5 通用的列表类 CList	24
第 4 章 窗口程序的基本概念	26
4.1 终端接口与GUI接口	26
4.2 组成要素	27
4.2.1 应用程序	27
4.2.2 窗口	28
4.2.3 消息传递机制	32
第 5 章 建立第一个窗口程序	34
5.1 使用Visual C++	34

5.1.1	创建新的工程文件	34
5.1.2	设定工程文件选项	35
5.1.3	建立新 CPP 文件	36
5.1.4	编译与执行示例程序	37
5.1.5	改错	38
5.2	示例程序	39
5.2.1	HelloApp	39
5.2.2	MyFrameWindow	40
5.3	增加资源文件	43
5.3.1	建立新的资源文件	43
5.3.2	符号与资源的包含文件 (Include Files)	44
5.3.3	插入资源	44
5.3.4	编辑位图资源	46
5.3.5	编辑图标资源	47
5.3.6	编辑菜单资源	48
5.3.7	引用资源	49
5.4	用AppWizard产生类似的程序	54
5.4.1	用AppWizard产生工程文件	54
5.4.2	产生了什么代码	60
5.4.3	在CChildView内显示文字	62
第 6 章	窗口的产生与处理	64
6.1	窗口类	64
6.2	产生CWnd对象	65
6.2.1	CWnd::Create()	66
6.2.2	CWnd::CreateEx()	68
6.2.3	WM_CREATE 消息	69
6.2.4	CREATESTRUCT	69
6.3	消息响应	70
6.3.1	映射表	70
6.3.2	给窗口发送消息	71
6.4	关闭窗口	73
6.5	窗口的位置及尺寸	75
6.5.1	取得窗口的位置及大小	75
6.5.2	如何移动窗口并改变大小	75
6.5.3	移动窗口、改变其大小及垂直相对位置(z-order)	75
6.5.4	移动与变更尺寸的消息	77
6.5.5	改变窗口的显示状态	77
6.5.6	工作区的相关信息	78

6.5.7 坐标系统的转换	78
6.6 窗口的状态与样式	79
6.6.1 图标化 (Iconic)	79
6.6.2 可见性 (Visible)	80
6.6.3 放大 (Zoomed)	80
6.6.4 使能 (Enabled)	80
6.6.5 活动 (Active) 与非活动 (Inactive)	80
6.6.6 子窗口	81
6.6.7 窗口的样式	81
6.7 工作区显示	82
6.7.1 与显示画面有关的消息	83
6.8 有滚动条的窗口	83
6.8.1 滚动条	83
6.8.2 滚动条的范围	85
6.8.3 滚动块的位置	85
6.8.4 滚动条的消息	85
6.8.5 滚动工作区	86
6.9 CWnd与句柄	86
6.10 Window Text	87
第 7 章 快速创建MFC应用程序	89
7.1 基础知识	89
7.1.1 对话框与一般窗口的差别	89
7.1.2 控件	92
7.1.3 控件的通知消息	93
7.2 示例: 编辑器与列表框的字符串交换	93
7.2.1 建立工程文件	94
7.2.2 对话框资源编辑器	95
7.2.3 设计对话框的外观	97
7.2.4 响应各种事件消息	100
7.2.5 运行的细节	105
7.3 信息框	109
7.3.1 示例	111
第 8 章 键盘、鼠标与时间	112
8.1 信息与输入焦点	112
8.1.1 改变输入焦点	112
8.1.2 与输入焦点有关的信息	113
8.2 键盘的信息	113

8.2.1	ASCII码、虚拟键码与扫描码.....	114
8.2.2	信息	114
8.3	鼠标	116
8.3.1	鼠标信息	116
8.3.2	更换鼠标的光标	118
8.3.3	显示等待光标	119
8.3.4	取得鼠标的控制权	119
8.4	特殊的状况	120
8.4.1	中文窗口下的文本输入	120
8.4.2	Windows 95 之后的新按键.....	121
8.5	示例：打字模拟程序	122
8.5.1	捕获消息	123
8.5.2	文本光标	123
8.5.3	鼠标光标	125
8.5.4	显示键入的字符	125
8.6	计时器消息与时间	130
8.6.1	查询系统时间	131
8.7	设计鼠标光标	131
8.7.1	绘制手掌型光标	132
8.7.2	光标的热点	132
8.7.3	更换鼠标光标	132
第 9 章	菜单与快捷键	133
9.1	菜单	133
9.1.1	命令消息	134
9.1.2	使用菜单	137
9.1.3	系统菜单	137
9.1.4	弹出式菜单	138
9.1.5	CMenu的成员函数	139
9.1.6	示例	143
9.2	快捷键	149
9.2.1	快捷键的工作方式	149
9.2.2	编辑快捷键表	149
9.2.3	示例	150
第 10 章	文档、视图与框架窗口	153
10.1	“文档/视图”(Document/View)模型	153
10.1.1	“文档”与“视图”	153
10.1.2	单文档界面与多文档界面	154

10.2	Document/View与框架窗口的关系	156
10.2.1	一个标准的 SDI 示例	157
10.2.2	编辑字符串表 (String Table)	161
10.3	CDocument 类	167
10.3.1	View 的管理	167
10.3.2	建立或打开文档	168
10.3.3	储存文档	170
10.3.4	关闭文档	171
10.3.5	Serialize	171
10.3.6	应该被重载的函数	174
10.3.7	其他成员	175
10.4	CView 类	176
10.4.1	应该被重载的函数	176
10.5	CScrollView 类	177
10.5.1	CScrollView 的工作原理	177
10.5.2	CScrollView 的成员函数	177
10.6	SDI 框架窗口CFrameWnd	178
10.6.1	重要的成员函数	178
10.6.2	菜单的自动 enable / disable	179
10.7	示例: 拉线绘图	180
10.7.1	CList	180
10.7.2	鼠标的操作	181
10.7.3	Serialize	181
10.7.4	更换鼠标光标	182
10.7.5	其他细节	182
10.7.6	改用 CScrollView	192
10.8	工具条与状态条	193
10.8.1	建立工具条	194
10.8.2	编辑工具条资源	196
10.8.3	建立状态条	196
10.8.4	在状态条中显示时间	197
10.8.5	工具条与状态条的开关	198
10.8.6	状态条与 Tooltip 中的辅助说明文字	198
10.9	MDI	202
10.9.1	一个标准的MDI示例	203
10.9.2	排列MDI子窗口	211
10.9.3	管理 MDI 子窗口	211
10.9.4	CMDIFrameWnd 的自动功能	212
10.9.5	Window 选项	212

10.9.6 最近使用的文档MRU	213
10.9.7 与 MDI 有关的消息	213
10.10 MDI化的LineArt	214
10.11 多视图类的MDI	226
10.12 分裂式窗口(Splitter Window)	236
10.12.1 内含“动态分裂窗口”的SDI标准示例	237
10.12.2 内含“静态分裂窗口”的SDI示例	240
10.13 更多不同的View类	249
10.13.1 另一类对话框 CFormView	250
10.13.2 可以直接编辑文字的 CEditView	250
第 11 章 文档视图模式与AppWizard	258
11.1 SDI工程文件	258
11.1.1 使用AppWizard建立工程文件	258
11.1.2 新工程文件中有哪些类	261
11.1.3 用ClassWizard重做SDI版的LineArt	261
11.2 MDI工程文件	268
11.2.1 使用AppWizard建立MDI工程文件	268
11.2.2 新工程文件中有哪些类	269
11.3 不用文档视图模式的工程文件	270
第 12 章 控件	272
12.1 静态类	272
12.1.1 CStatic::Create()	272
12.1.2 样式	272
12.1.3 成员函数	274
12.1.4 CStatic 的通知消息(Win95)	274
12.2 按钮	275
12.2.1 CButton::Create()	275
12.2.2 样式	275
12.2.3 成员函数	278
12.2.4 通知消息	279
12.2.5 示例: 计算器	280
12.3 滚动条 CScrollBar	292
12.3.1 CScrollBar::Create()	293
12.3.2 样式	293
12.3.3 成员函数	293
12.3.4 通知消息	295
12.3.5 示例: Multithread 动画	296

12.4	列表框	304
12.4.1	样式	305
12.4.2	成员函数	306
12.4.3	通知消息	309
12.5	编辑控件	310
12.5.1	样式	310
12.5.2	成员函数	310
12.5.3	通知消息	313
12.6	组合框	313
12.6.1	样式	314
12.6.2	成员函数	314
12.6.3	通知消息	318
12.7	信息交换与检查	318
12.7.1	简单的示例	319
12.7.2	常见的 DDX 函数	323
12.7.3	常见的 DDV 检查函数	324
12.7.4	在ClassWizard中增加DDX/DDV选项	325
第 13 章	设备上下文与基本绘图工具.....	327
13.1	概述	327
13.2	CDC	327
13.2.1	建立与删除 CDC 对象	327
13.2.2	CDC 的内含资源对象	329
13.2.3	查询DC的相关信息	330
13.2.4	CDC 的其他成员函数	332
13.3	CWindowDC 与 CClientDC	333
13.3.1	CWindowDC 的构造函数	333
13.3.2	CClientDC 的构造函数	333
13.4	画笔与画刷	333
13.4.1	画笔	333
13.4.2	画刷	335
13.5	基本绘图函数	336
13.5.1	点	336
13.5.2	线	337
13.5.3	弧线	337
13.5.4	椭圆	339
13.5.5	矩形	340
13.5.6	饼图	340
13.5.7	多边形	341

13.6 色彩与调色板	342
13.6.1 显卡的概念	342
13.6.2 16色与 256 色模式	342
13.6.3 32K, 64K以及16M 色模式	343
13.6.4 逻辑调色板和硬件调色板	343
13.6.5 常用来表示“颜色”的结构与宏定义	345
13.7 使用调色板	347
13.7.1 建立调色板对象	347
13.7.2 将调色板对象指派给 DC	347
13.7.3 “实现”(Realize)调色板	347
13.7.4 画图	347
13.7.5 响应系统消息	347
13.7.6 调色板的其他成员函数	348
13.8 字体与文本输出	349
13.8.1 基本知识	349
13.8.2 如何描述“字体”	350
13.8.3 CFont 类	353
13.8.4 多样的文字输出方式	354
13.8.5 文字输出的格式与属性	358

第 14 章 位图.....363

14.1 简介	363
14.1.1 设备独立位图DIB	363
14.1.2 设备相关位图DDB.....	364
14.2 CBitmap类	364
14.2.1 建立 DDB.....	364
14.2.2 CBitmap 成员函数.....	365
14.2.3 与 DDB 有关的函数.....	366
14.3 设备无关位图类 DIB	367
14.3.1 DIB 的结构	367
14.3.2 BITMAPFILEHEADER	367
14.3.3 BITMAPINFO.....	368
14.3.4 BITMAPINFOHEADER	368
14.3.5 RGBQUAD	369
14.3.6 BMP 文件的格式.....	369
14.3.7 Win32 API 和 DIB 有关的函数	370
14.3.8 DIB 与 DDB 的转换.....	371
14.4 处理 DIB 的示例.....	371
14.4.1 自己设计 CDib 类	371

14.4.2 主程序	383
14.4.3 多页对话框	395
第 15 章 打印与坐标系	401
15.1 打印	401
15.1.1 设定打印机的状态	401
15.1.2 通过CView 打印	403
15.1.3 示例	406
15.2 比例、原点与方向	409
15.2.1 如何改变映射模式	410
15.2.2 理想与现实	410
15.2.3 示例	411
15.3 更大的灵活性	415
15.4 打印预览	417
第16章 通用控件	418
16.1 概述	418
16.1.1 通用控件的种类	418
16.1.2 通用控件的共性	421
16.1.3 通用控件的通知消息	421
16.1.4 通用控件都会产生的通知消息	422
16.2 CSliderCtrl	422
16.2.1 CSliderCtrl 的样式	423
16.2.2 CSliderCtrl的成员函数	423
16.2.3 CSliderCtrl 的通知消息	425
16.3 CSpinButtonCtrl	425
16.3.1 CSpinButtonCtrl 的样式	426
16.3.2 CSpinButtonCtrl 的用法	426
16.3.3 CSpinButtonCtrl 的其他成员函数	426
16.3.4 CSpinButtonCtrl 的通知消息	427
16.4 CHeaderCtrl	427
16.4.1 CHeaderCtrl 的样式	427
16.4.2 如何使用 CHeaderCtrl	427
16.4.3 其他的成员函数	429
16.4.4 CHeaderCtrl 的通知消息	429
16.5 CAnimateCtrl	429
16.5.1 CAnimateCtrl 的样式	430
16.5.2 CAnimateCtrl 的成员函数	430
16.5.3 CAnimateCtrl 的通知消息	430

16.5.4 示例	430
16.6 CProgressCtrl	432
16.6.1 CProgressCtrl 的用法	432
16.6.2 示例	432
16.7 CtreeCtrl, CTreeView与CImageList	433
16.7.1 使用 CImageList	433
16.7.2 CTreeCtrl 的样式	434
16.7.3 CTreeCtrl 的用法	435
16.7.4 其他成员函数	437
16.7.5 CTreeCtrl 的通知消息	439
16.7.6 CTreeView	441
16.7.7 示例: 显示磁盘的树状目录	441
16.8 CListCtrl 与 CListView	453
16.8.1 CListCtrl 的样式	455
16.8.2 CListCtrl 的用法	455
16.8.3 CListCtrl 的成员函数	458
16.8.4 CListCtrl 的通知消息	462
16.8.5 CListView	463
16.8.6 示例: 显示目录中的文件列表	463
16.9 示例: 程序管理器	475
第 17 章 使用ActiveX控件	479
17.1 ActiveX控件的由来	479
17.1.1 面向组件的程序设计 (Component-Oriented Programming)	480
17.2 使用ActiveX控件	480
17.2.1 安装 ActiveX 控件	481
17.2.2 MCI32.OCX 简介	482
17.3 设计ActiveX容器	482
17.3.1 启动ActiveX	482
17.3.2 插入 ActiveX 控件	483
17.3.3 在对话框中使用 MCI 控件	485
17.3.4 在程序中控制MCI控件	487
第 18 章 进程和线程	489
18.1 Win32 的内存管理	489
18.1.1 Win32 的内存分配函数	491
18.2 Process 简介	494
18.2.1 产生 Child Process	494
18.2.2 与 process 有关的信息	497

18.2.3 结束 process	498
18.3 Thread 简介	498
18.3.1 产生 Child Thread	499
18.3.2 Thread的信息	499
18.3.3 终止Thread	500
18.4 同步 (Synchronization)	500
18.4.1 Critical Section	502
18.4.2 等待对象的状态改变	505
18.4.3 Mutex	507
18.4.4 Event	509
18.5 进程间通信IPC	512
18.5.1 共享内存与文件映射	513
18.6 CWinThread	516
18.6.1 GUI Thread 与非GUI Thread	516
18.6.2 利用CWinThread产生Thread的流程	516
18.6.3 其他的CWinThread成员	517
第 19 章 Profile与Registry	518
19.1 什么是Profile	518
19.2 什么是 Registry	519
19.2.1 建立新的key	520
19.2.2 打开已存在的entry	521
19.2.3 写入信息	522
19.2.4 读取信息	522
19.2.5 删除key或entry	523
19.2.6 关闭Key	523
附录 Internet上的MFC资源	524

第1章 综 述

初学者常常问这样的问题：是学 C++ 好还是学 Visual C++ 好？

其实这是个很奇怪的问题，就好比在问：是学开车好还是学开福特车好？开车是一种普遍的技术，用这个技术可以控制福特车或捷达车，这在技术上并没有很大的差别，顶多只是一些开关位置的不同而已。同样的道理，C++ 是一种程序语言，现在已经成为 ISO 标准，是一种普遍的技术。而 Visual C++ 只是一种协助用户使用 C++ 语言设计软件的工具。

类似的工具还有很多，比如 Inprise 公司的 Borland C++，Borland C++ Builder，Watcom 公司的 Watcom C++，甚至还有免费的 GNU C++，这些都是协助用户使用 C++ 设计软件的工具。现在有了一个问题，既然可以使用免费的 GNU C++，为什么还要花钱买昂贵的 Visual C++ 呢？

首先，Visual C++ 有功能强大的集成开发环境，在此环境中用户可以方便地进行项目管理、编写 C++ 源程序、编译、改错，而这些 GNU C++ 都不具备。更重要的是，Visual C++ 提供了一套 Microsoft Foundation Classes（简称为 MFC）的程序类库（classes library），现在这套由微软开发的程序类库已经成为设计窗口应用程序的准工业标准了。

标准 C 语言程序库提供了 400 种库函数，如 printf()，fopen() 等等，这些函数可以实现文件存取、数学计算、字符串处理等功能。MFC 则包含了上百种事先用 C++ 写好的类，将这些类组合起来，可以编写出功能强大的 Windows 应用软件。

因为 MFC 是一套用 C++ 编写的程序类库，因此不只 Visual C++ 提供了这套程序库，Borland C++，Watcom C++ 也可以提供。也就是说，只要学会 MFC 的用法，就可以在 Borland C++ 或者 Watcom C++ 上使用 MFC 设计窗口应用程序。

除了 MFC 之外，其他厂商也有类似的产品，例如 Borland C++ 的 OWL (Object Window Library)，Borland C++ Builder 的 VCL。这些也是用 C++ 设计的程序库，可以协助用户快速地开发窗口应用程序。这些程序库各有所长，可以用于不同的方面。MFC 的最大优势是能抢先支持各种新功能，如 ActiveX 等，但是在面向对象技术方面则略逊一筹，比如 MFC 的类非常难以实现多重继承。

1.1 本书结构

本书的目的是讲述如何用 MFC 来设计窗口应用程序。在阅读本书之前，基本的要求是必须学会 C++ 程序语言，而且还要能操作 Windows，如果在面向对象及数据结构方面有深入的了解更好。

第 2 章简单地回顾一些 C++ 语言的重点；第 3 章介绍 MFC 程序库中与窗口无关的基础类；第 4 章讲述窗口程序设计的基本概念。

第 5 章将实际动手设计一个最简单的窗口应用程序。接下来在第 6 章详细地介绍窗口的类 CWnd。第 6 章后半部分的内容稍微有点深入，初学者可以先跳过不看。第 7 章讲述

用 Visual C++ 的 AppWizard 及 ClassWizard 设计以对话框为主的应用程序，并且在此处介绍 Windows 的子控制。第 8 章讲述输入设备：键盘及鼠标的相关用法。第 9 章介绍控制菜单的方法。第 10 章是本书最重要的一章，介绍 MFC 的文档和视图结构，请务必搞清楚。第 11 章介绍用 AppWizard 辅助设计文档、视图结构软件的方法。第 12 章是第 7 章的延伸，详细地介绍了各种子控制的用法，初学者可先跳过不看。

第 13 章的主题是绘图，所有 Windows 绘图的知识都可以在这里找到。第 14 章介绍处理位图的方法。在该章后半部分，笔者设计了一个可以处理 Windows 常见的*.BMP 文档的类，十分常用，请务必仔细阅读。由于在屏幕上显示图形有时不够大，还需要把图片打印出来，因此第 15 章介绍打印及控制打印图形尺寸的方法。

第 16 章介绍在 Windows 95 中才出现的通用子控制，该章的示例就是逐步建立一个类似文件管理器的应用程序。第 17 章讲述如何在程序中使用流行的 ActiveX 子控制。第 18、19 章是比较高级的论题，分别讨论 Win32 模式下的内存管理、进程和多线程系统以及 registry 系统，初学者可跳过不看。

1.2 本书约定

File|Open 指菜单 File 中的选项 Open，以垂直线分隔这些选项。

01 void main()

02 { 程序代码列表左侧的数字是行号。

03 }

int getch(); 函数的原型(prototype)，即函数的名称、返回值类型、参数。

1.3 特殊说明

出于价格方面的考虑，本书没有附带光盘，但本书所涉及到的程序代码和共享软件，均放在本公司的网站上，地址为：<http://www.kehaipzbook.com>，可以免费拷贝。

第2章 C++重点回顾

如果你对 C++ 程序语言还不熟悉，那么本章内容刚好适合你。因为笔者的前几本书中都假设读者对 C++ 有相当程度的了解，不过后来发现写信来问 C++ 问题的人远比对内容有疑问的人多得多。因此本书特意增加了这一章，介绍一些 C++ 语言的重点概念。

2.1 类、对象、实例

这 3 个名词常常困扰很多人，因此需对它们进行重点说明。类就是种类、品种，比如说狗，就是泛指长了一条尾巴，会吠会咬人的哺乳动物，不过每一条狗都是独立、有个性的小生命，有的狗叫小花，有的叫小黄，有的叫来福。因此，狗指的是一种动物的类，而小花、小黄、来福则是狗的实例。在大多数情况中，实例指的就是对象。

2.2 构造函数 Constructor

Constructor 在本书中统一称为构造函数。创建实例时，电脑会自动跳到该实例的构造函数去执行。通常，构造函数中的内容都是和初始化实例相关的程序代码。

不过，构造函数也会有一些奇怪的用法。比如，类可能会有数个父类，而父类也有它自己的构造函数，因此子类必须调用父类的构造函数，以便对从父类继承的部分进行初始化。例如下面这段程序：

```
01 class Parent
02 {
03     public:
04         int    parent_var;
05
06         Parent ()           { parent_var = 10; }
07         Parent ( int init ) { parent_var = init; }
08 };
09
10 class Child : public Parent
11 {
12     public:
13         int    child_var;
14
15         Child ()           { child_var = 23; }
16         Child ( int i ) : Parent ( i ) { child_var = 24; }
17         Child ( int i, int j )
18             : Parent ( i ), child_var ( j )      {}
19 };
20
```