Designed for Microsoft Windows NT Windows 95

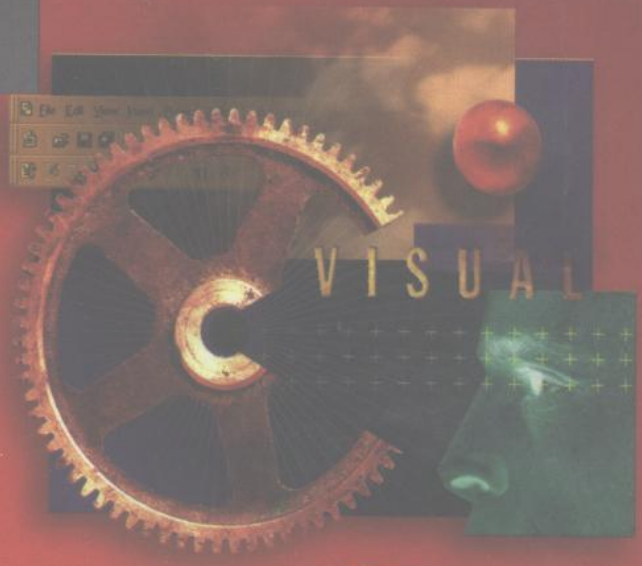
CD-ROM included

Microsoft® Visual C++®

使用指南

Version 5.0

Microsoft Programming Series



Developer Studio 和 Microsoft Visual C++ 编程工具的完整、实用的使用指南

[美] Beck Zaratian 著
詹津明 杨 欣 译
沈纪新 审校

清华大学出版社
http://www.tup.tsinghua.edu.cn

Microsoft Press

Microsoft Visual C++
使 用 指 南

[美] Beck Zaratian 著
詹津明 杨 欣 译
沈纪新 审校

清 华 大 学 出 版 社

(京)新登字 158 号

Microsoft Visual C++ 使用指南

Microsoft Visual C++ Owner's Manual

Beck Zaratian

北京市版权局著作权合同登记号 图字 01-98-0187 号

图书在版编目(CIP)数据

Microsoft Visual C++ 使用指南/(美)萨拉田(Zaratian, B.)著;詹津明,杨欣译.—北京:清华大学出版社,1999.3

书名原文:Microsoft Visual C++ Owner's Manual

ISBN 7-302-03316-1

Ⅰ.M… Ⅱ.①萨… ②詹… ③杨… Ⅲ.C 语言-程序设计-指南 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(1999)第 00804 号

出版者:清华大学出版社(北京清华大学校内,邮编 100084)

http://www.tup.tsinghua.edu.cn

责任编辑:张善余

印刷者:清华大学印刷厂

发行者:新华书店总店北京发行所

开 本:787×960 1/16 印张:27.75 字数:620 千字

版 次:1999 年 4 月第 1 版 1999 年 4 月第 1 次印刷

书 号:ISBN 7-302-03316-1/TP·1782

印 数:0001~8000

定 价:65.00 元(含光盘)

译 者 序

“公欲善其事，必先利其器”。做其它事情如此，软件开发自然也不例外。复杂软件项目的完成质量，一方面取决于开发者对于待解决的问题认识的深刻程度，另一方面则更多地取决于对选用的开发工具特性的认识程度。对开发工具认识越是深刻，越是全面，就越能发挥其长处，避开其短处，从而提高软件开发的质量。

这本《Visual C++ 使用指南》就是这样一本书。它最大的特点就在于把介绍的重点放在了 Visual C++ 这一开发工具“本身”的特性上，而不在于如何使用 Microsoft C++ 编程语言或 Microsoft 基本类库。这也是本书不同于其它介绍 Visual C++ 书籍的最显著的特点。

书中，作者以其丰富的开发经验及对 Visual C++ 娴熟地把握，向读者循序渐进地介绍了 Visual C++ 的一些最重要的特征，如从最基本的开发环境本身及应用程序向导的介绍到文本及资源编辑器的使用，从对类向导的讨论到 Active X 控制的使用，从调试程序及编译器的优化到开发环境的定制等，几乎覆盖了关于 Visual C++ 本身的全部重要内容。相信本书对于读者掌握 Visual C++ 本身的特性，并进而加深对其它特性的理解会有极大的帮助。同时，作者“以例子展示特性”的叙述风格也使得书中的内容更生动、更易理解，完全没有了技术资料中常见的枯燥乏味的缺点。

正是由于上述原因，我们决定把本书翻译出来介绍给国内的读者。虽然我们尽了最大的努力，但缺点与错误在所难免，希望得到读者的指正。

目　录

第二部分 编 辑 器

第三部分 编 程 辅 助

第四部分 高 级 话 题

第五部分 附 录

第0章 绪 论

本书是关于 Microsoft Visual C++ 的,不是关于 C++ 语言和 MFC 库,而是 Visual C++ 本身。

当然,Visual C++ 有它自己的手册——联机帮助。该帮助的内容庞大,使你相信你想要知道的东西一定在某个地方。但这也是联机帮助的问题所在:当你知道你要寻找什么时,它才最有用。本书是为了补充联机帮助,而不是要代替它。帮助系统和书籍的目的和风格都大不相同,彼此不能替代。一个提供信息,另一个则教你学习。一个有广度,另一个有深度。你从帮助系统能知道如何,却不能知道为什么。

本书要帮助读者成为 Visual C++ 的熟练使用者。它由逻辑上循序渐进的资料组成,演示整体的各部分之间如何交互,用例子代码进行说明。你可以坐在你最喜欢的地方阅读它,这是联机帮助做不到的。但是,联机帮助提供了快速和广度上的帮助。帮助文本覆盖了 Visual C++ 的每个角落,而本书只讲一些基础的东西。可以从本书学起,在你有了牢固的基础和更有经验之后,再使用联机帮助。你越熟悉它,联机帮助对你越有用。

Visual C++ 和 C++ 及 MFC 库是紧密联系在一起的,你不可能只谈 Visual C++ 而对 C++ 和 MFC 避而不谈。本书中提供了很多代码段和程序,它们是为了解释 Visual C++ 的某些特点。代码必须有注释——否则它就没用了——而对这些例子程序的讲解肯定要涉及到 C++ 和 MFC 的技术。但它们对如何使用编译器这个主题不会有干扰。有一些很好的书讲解了 C++ 编程和 MFC 库。

本书讲的是 Visual C++ 版本 5,但使用更早版本的读者也能从中受益。Visual C++ 的某些方面比上一版有了较大的变化,但很多地方几乎没有什么变化。现在的 Visual C++ 是一个薄薄的包装,有一些广告和印刷资料,还有一两张 CD-ROM。但你会发现 Visual C++ 中有大量的资料。我把它称为"编译器"(只是因为没有更好的名字)。除了编译器之外,Visual C++ 还提供了链接器、make 工具、调试器、文本编辑器、资源编辑器、开发环境、Microsoft 基本类库(MFC)、运行时库、几千行的源代码和其它一些东西。重复一下,本书不会讲述所有的东西。我的目的是要帮你掌握 Visual C++,而不是用细节淹没你。

0.1 你应该知道的知识

这种类型的书应该从学习曲线的零点之上的某处开始。起点太低，讨论就要涉及很多基本的解释，起点太高又会吓跑很多读者。本书的要求不太高。我假定读者已经熟悉 C 和 C++ 编程语言，以前为 Windows 编过程，至少听说过 MFC。你不必是一个专家，但如果你了解基本的概念（如指针、类和消息），就很容易理解本书的内容和例子代码。幸运的是，对编译器来说并不抽象，它只是一种软件。

0.2 Visual C++ 历史简介

Visual C++ 最初的根在于 Borland 而不是 Microsoft。一些读者可能还记得 Turbo Pascal，它在 DOS 下引入了集成开发环境（IDE）的概念。IDE 是指编辑器和编译器一起工作，它们能从同一个地方访问。你可以在编辑器中编写源代码，按下 Compile 按钮来启动编译器，当它发现一个错误时就把编辑器的光标停在错误的语句上，等你来修改。它就是向程序员提供一个不需要离开的环境。

C 语言在那时（1987 年）引起了大家的注意，从 Turbo Pascal 发展到了 Turbo C。Microsoft 开发了一个类似的产品，名为 Quick C。Quick C 作为独立产品出售，但它也是 Microsoft 的名为 BigC 的 C 编译器的一部分。那时，Big C 的版本是 5.0。它的竞争对手的名字有的现在看来有点儿古怪：Computer Innovations，Datalight，Lattice，Manx。其他一些幸存了下来，著名的有 Borland 和 Watcom（现在是 PowerSoft）。它们的优秀产品现在继续同 Microsoft 竞争。

把 Quick C 和 Big C 放在一起的目的是让程序员可以在 Quick C 的方便的 IDE 中编写代码。Quick C 的编译很快，主要是因为它对优化的处理很简单。当在 Quick C 中调试和运行完一个程序后，程序员可以用 Big C 创建一个发行版本。Big C 对代码优化的处理要认真得多。当用 Big C 编译后，程序的大小常常会减少百分之十五或更小。

Quick C 和 Turbo C 引导很多人开始 C 编程，但他们没有得到开发人员长久的喜爱。一个原因是他们的编辑器不是太好。（Quick C 的编辑器后来集成进了 Microsoft QuickBasic，现在还作为 Microsoft Windows 95 的 DOS 编辑器 Edit.com 存在。）另一个问题是 DOS 的 IDE 占用了很多内存，为开发时程序的运行留下了很少的空间。你不得不退出 IDE 来运行和调试你的程序。很多使用 Quick C 的程序员只使用它的命令行版本。

但这时出现了 Windows 3.0。Windows 3.0 和 3.1 为个人计算机引入了真正的 IDE。内存的限制消失了。如果要为 Windows 开发程序，很自然地要使用 Windows 环境。显然，在 Windows 中为 Windows 开发程序能产生更好的产品。

出乎很多人的意外，Microsoft 把它的努力集中在内部的 C 编译器而不是用户界面上。当版本 7.0 出现时，它仍然是一个基于 DOS 的产品，可以在 Windows 中的 DOS 框中运行，或

者同扩展内存管理器一起工作。作为一种让步，版本 7.0 提供了一个名为 Programmer's Workbench 的字符模式的 IDE。它同今天的标准相比似乎很笨拙，但它是从 Quick C 自然演化过来的。它的菜单的很多命令现在仍然使用，如 New，Open，Save As，Build 和 Open Project。

版本 7.0 对编程最大的贡献不是它的 IDE，而是它支持 C++。Microsoft 第一次强调了它的编译器的双重特性。支持不只是使编译器认识 C++ 的新命令，它还引入了 Microsoft 基本类库的版本 1.0 及其源代码。如果没有这些已经写好的类，对于 Windows 编程，C++ 就不会像今天一样普遍了。

在下一个版本中，Microsoft 抛弃了产品中与 DOS 相关的大部分。Microsoft C/C++ 8.0 支持真正的 Windows IDE，它作为 Visual C++ 1.0 被大家所熟悉。名字是从早先的 Visual Basic 中继承来的，但这两个产品却很不同。Visual Basic 允许开发人员用鼠标多点几下，不需什么编程就能建立一个能工作的 Windows 程序，而 Visual C++ 只通过名字 wizard 的特殊的动态链接库创建了初始的代码。wizard 为开发节省了很多重复的前端工作。

在 Visual C++ 1.5 之后，Microsoft 决定不再为支持 16 位编程努力了。Visual C++ 2.0 还支持 16 位，但 Visual C++ 5.0 只创建 32 位的程序。没有 Visual C++ 3.0。发行版本号从 2 跳到了 4 是为了使 Visual C++ 与 MFC 相一致，但这引起了一些小小的混淆。

0.3 本书的内容

本书分为 5 个主要部分，每部分讨论了有关 Visual C++ 和它的开发环境的一个专题。第 3 章之前的讨论是比较基本的，其中包括了文本编辑器。这样就确保了每位读者（不管是新手还是专家）都能够使用 Visual C++ 开发环境和在文本编辑器中编写代码。从第 4 章开始，讨论就越来越技术化了。

0.3.1 第 1 部分——基础

我们称为 Visual C++ 的大部分实际是它的开发环境，被称为 Microsoft Developer Studio。二者的区别并不重要，通常它们是可以互相替换的。但你在学会 Developer Studio 之后才能有效地使用 Visual C++。（当你首次启动 Developer Studio 时，会出现一个画面含有被称为 Visual Studio 的东西。Visual Studio 是 Microsoft 的开发工具系列的总称。它同 Developer Studio 并不一样，所以在本书中你可以不用管什么 Visual Studio。）

第 1 章是 Developer Studio 的初步知识，它介绍了工作在环境中时会遇到的主要窗口，并解释了如何使用 Visual C++ 的联机帮助系统。

第 2 章介绍了 AppWizard，它是为使用 MFC 的一般 Windows 应用程序创建起初文件的 Visual C++ wizard 程序。在本书中，我们将使用 AppWizard 来创建一些例子程序。

0.3.2 第2部分——编辑器

Developer Studio 提供了 3 种不同的编辑器——第 1 个用来创建文本源代码,第 2 个用于菜单和图形文件,第 3 个用于对话框。每个编辑器都专门有一章讨论。第 3 章介绍了文本编辑器。该章介绍了重要的菜单命令,打开文本文档的快捷方式和宏。

第 4 章介绍了图形编辑器,它用于创建包括菜单、位图、图标和工具栏的资源数据。本章还含有一个名为 DiskPie1 的例子程序。每一节都先介绍了如何使用图形编辑器来创建一个特定的界面元素,如菜单和工具栏等,再演示如何将该元素加入到 DiskPie1 程序中。到本章的结束,该程序就成为了一个有用的工具,它以饼图的形式显示磁盘和内存的使用情况。

第 5 章介绍了对话框编辑器,显示了如何使用 Visual C++ 来设计对话框和建立像 Windows Character Map 和 Phone Dialer 工具那样的基于对话框的应用程序。本章也介绍了几个例子。

0.3.3 第3部分——编程辅助

第 3 部分中的章节介绍了如何使用两个 Developer Studio 中的两个基本工具来加速程序的开发。第 6 章介绍了 ClassWizard,它很难讲清楚但易于使用。当开发 MFC 程序时,你会发现 ClassWizard 对于创建和维护类是非常有用的。

第 7 章中介绍的 Gallery 提供了很多可增加的部件,你只要单击几下鼠标就能把它们加入到你的项目中。第 7 章还演示了如何为 Gallery 创建你自己的部件。

0.3.4 第4部分——高级专题

第 8 章介绍了 ActiveX 控制和如何在程序中使用它们。第 9 章则介绍了如何编写 ActiveX 控制。一个名为 Tower 的例子一步步地介绍给你如何用 MFC 编写 ActiveX 控制。结果能被嵌入到任何支持 ActiveX 控制的程序中。

第 10 章介绍了调试器的基本功能,它是 Visual C++ 最完美的部件之一。该章介绍了调试的内部情况,调试器窗口和工具栏,而后用调试器来改正一个例子程序中的一个隐藏的错误。

在程序调试完之后,就应该打开编译器优化来创建一个发行版本。第 11 章中介绍了编译器优化中很难理解的一些问题,告诉你很多 Visual C++ 优化开关的确切含义和使用目的。

读到第 12 章之前,你应该已经熟悉了 Developer Studio,知道自己喜欢什么和想要修改什么。该章介绍了如何定制环境来满足你自己的口味。它还通过例子解释了如何编写能无缝地集成到 Developer Studio 中的宏和工具。

0.3.5 第5部分——附录

附录A给出了一张标准的ASCII和ANSI字符表,其中包含了字符的八进制表示。

附录B介绍了ClassWizard支持的MFC类。

附录C介绍了Microsoft Visual Basic Scripting Edition,即VBScript。Developer Studio 5.0中集成了VBScript作为宏语言,所以如果你从没用过VBScript,介绍一下是很有用的。虽然在Developer Studio中记录宏不需要VBScript的知识,但你可以只通过VBScript编程来创建一个通用的宏。

0.4 例子代码

本书中的几乎所有例子程序都是用C++写的,并使用了MFC。(两个例外是第4章中的光标演示程序和第12章中的基于控制台的小工具。)但对于书中的一些代码段,我使用了C。我发现在解释编程思想时,C++不如C那么简明,而且C还是今天的编程人员的一种通用语言。从理论上说,C++程序员了解C,反过来就不一定了。另一方面,在演示MFC程序时,C就没什么用处了。当我认为某个概念很重要时,就会给出等价的C和C++代码。

本书中的很多章的内容最好用例子来解释,我尽力使用那些有趣、有用和解释性的例子程序。一些程序是用AppWizard创建的,另一些不是,这样就尽可能地模拟了实际编程的情况。几乎每个程序在书中都有详细的讨论。书中还包含了它们的源代码,你就不需要在编辑器中打开源文件来看它们了。程序力求简明,所以我包含了很少的错误检查代码。

我假定你阅读本书是因为你拥有(或至少是好奇)Visual C++,它是专为Win32开发的产品。因此,所有的例子代码和几乎所有的讨论都是针对Win32的。例子程序是在Windows 95下创建的,但大部分也在Windows NT 3.5下测试过。

0.5 附带的CD

所有例子程序的项目文件都在本书附带的CD中。要把所有的项目拷贝到你的硬盘上,可以按下面的步骤运行Setup程序:

1. 单击Windows任务条上的Start按钮,选择Run命令。
2. 在Run对话框中输入"d:setup",这里d代表你的CD-ROM驱动器的盘符。

Setup程序把大约2MB的文件从CD拷贝到你的硬盘上,把它们放在一个名为VC++ Owner's Manual(或你指定的一个名字)的子目录下。运行Setup完全是可选的,如果你愿意的话,可以直接从CD上检索这些文件。你会在Code子目录中找到所有的文件。

下面的子目录给出了程序所在的章号和项目名。例如,子目录Chapter.05 MfcTree下包含了第5章中给出的MfcTree程序的所有文件。每个项目目录下还有一个名为Release的子

目录，其中含有程序的可执行文件，所以你不用建立程序就可以运行例子程序。如果你要自己建立例子程序，就在启动 Developer Studio 之后，选择 File 菜单中的 Open Workspace 命令。在你的硬盘上找到项目文件夹，再双击项目的 DSW 文件。

例子程序的项目名都保持在 8 个字符之内。这是为了方便那些仍喜欢用基于 DOS 的文本编辑器的读者，这些编译器不认识长文件名。一些老的 CD 驱动器对长文件名也有问题。

0.6 一些定义

在进一步讨论之前，我们需要先定义一些词汇，如建立(build)，项目(project)，目标(target)和应用程序框架(application framework)。我们在以后的章节中要用到它们。

建立意味着编译和链接，把一组源代码变成可执行程序。我们编译源文件，链接目标文件，建立项目。项目有两个相关的含义。它可以指最终的项目，即你建立的程序，但更确切地是指创建程序的一组文件，包括源文件、预编译头文件、资源文件、图形文件和其它需要使用的文件。Developer Studio 允许你一次打开一个项目，你就可以访问该项目的所有文件，并编译、建立和调试它们。每个项目可以包含任意数目的子项目。例如，你可能开发一个应用程序作为主项目，同时维护一个动态链接库作为子项目。

当你建立一个项目时，你创建的应用程序是发行和调试这两种版本之一。Visual C++ 有时使用目标这个词来指建立的类型。项目的发行目标是你给最终用户的可执行程序。调试目标是你在开发程序时使用的可执行程序。项目的设置(被称为配置)决定了可执行文件的类型。

MFC 库是通过把 Win32 API 表示为一组类对象来使 Windows 编程更容易。使用 MFC 的程序可以利用已经测试过的代码作为程序框架来处理很多任务，否则这些都需要自己来完成。唯一的代价是它使得可执行文件的大小可能要大一些，而且有一些局限性。框架通过类来指导程序的结构而不是细节。不过，MFC 并不严重地限制程序员的创造性，这已经被使用 MFC 的各种各样的 Windows 程序所证明。

这里还要讲关于文件夹的一个问题。缺省时，Visual C++ 把它的文件放在一个名为 Program Files 的文件夹中。当引用某个 Visual C++ 的子文件夹时，如 DevStudio \ SharedIDE \ Bin，我不在路径名前加上 Program Files。因为路径名已经很长了，而且文件夹名字中的空格也可能会让人以为“Program”不是路径的一部分。

0.7 进一步的阅读

这里有几本书我认为值得使用 Visual C++ 的程序员阅读一下，它们恰好都是 Microsoft Press 出版的。