循序渐进学用 Visual C++5.0

王海云 张颖峰 编著

海洋出版社

循序渐进学用 Visual C++5.0

王海云　张颖峰　编著

海洋出版社

1998年·北京

内 容 简 介

本书全面介绍了运用微软的优秀开发工具 Visua C++ 5.0 进行程序设计的基本知识。内容涉及 Windows 编程基本概念、Visual C++ 集成开发环境 Visual Studio 的使用、MFC 基础类库、用户界面技术、文档/视结构、用户模块设计、利用 ODBC 和 DAO 进行数据库编程，多媒体编程、多线程和串行通信编程技术。在附录中还介绍了 Visual C++ 程序的调试、Borland C++ 程序向 Visual C++ 程序的移植以及有关 Visual C++ 的一些热门网址。

本书内容翔实、深入浅出、实用性强，是 Visual C++ 初学者很好的入门教材，也适用于有经验的 VC 程序员参考使用。

图书在版编目(CIP)数据

循序渐进学用 Visual C++ 5.0/王海云等编著. －北京：海洋出版社，1998.9

ISBN 7-5027-4596-3

Ⅰ. 循… Ⅱ. 王… Ⅲ. C 语言 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(98)第 19118 号

海洋出版社 出版发行

(100081 北京市海淀区大慧寺路 8 号)

北京媛明印刷厂印刷 新华书店发行所经销

1998 年 9 月第 1 版 1998 年 9 月北京第 1 次印刷

开本：787×1092 1/16 印张：25.75

字数：600 千字 印数：1～3000 册

定价：42.00 元

海洋版图书印、装错误可随时退换

前　言

欢迎进入 Visual C++ 的美妙世界。如果你打算致力于 Windows 95/NT 编程，那么也许本书将会结束你在书架前的徘徊。因为本书是针对微软公司最新发布的 Visual C++ 5.0 的，具有内容新颖、概念清晰、实用性强、适合中国读者等特点。

现在流行的 Windows 软件开发工具主要包括 Visual Basic、Delphi、Visual FoxPro 和 Visual C++ 等，到底应该选择哪一种呢？Windows 编程是一个复杂的过程，所以程序员希望得到一种能够掩盖其复杂性的易于使用的开发工具，在这一点上，上述几种软件都能满足要求。另一方面，正因为 Windows 编程是复杂的，所以程序员不能停留在简单语句的水平上，而应该深入研究下去，练就"绝活"，只有这样，才能编出让老板刮目相看，令同事赞叹不已的程序来。从这一点来说，Visual C++ 将是你的最佳选择。

Windows 本身是汇编语言和 C/C++ 的产物，它的编程接口也是 C 语言的，主要包括成百上千的 API 函数，所有的 Windows 开发工具都是建立在这些 API 基础之上的。单从 Windows 与 C/C++ 的密切关系来看，就没有理由不使用 Visual C++。

本书的内容主要是围绕 MFC 展开的。MFC(微软基础类库)是 Visual C++ 的精华所在，它提供了一个功能强大的 C++ 类库，在 AppWizard 和 ClassWizard 的帮助下，程序员可以用 MFC 类库方便快速地设计出功能强大的应用程序。用 MFC 编写的程序具有性能优良、可靠性高、可移植性好等特点。

MFC 的一个突出特点是"能上能下"，它一方面封装了 Windows 编程的复杂性，另一方面，它提供了适应和掌握这种复杂性的捷径。例如，MFC 类的许多成员函数与底层的 Windows API 函数同名，并且函数参数和功能也类似。因此，随着对 MFC 的不断使用和深入研究，程序员将逐渐接触到 Windows 的核心，从而真正成为 Windows 编程的高手。MFC 在 Windows 编程中的地位，就像当年 C 语言在 DOS 编程中的地位一样。

本书既不是简单的操作手册，也不是长篇累牍的大全，而是实用性强、内容充实的技术型书籍。本书的两位作者具有多年的 Visual C++ 编程经验，他们知道读者需要什么东西。通过学习本书，初学者将迅速掌握一些 Win 32 编程的实用技术。而使用过 Visual C++ 的用户也会在书中找到许多 Visual C++ 5.0 方面的新内容，并了解到一些最新热点技术，如多线程编程、ODBC 和 DAO 编程、多媒体编程等等。

本书的另一大特点是概念清晰。作者不但告诉读者应该怎么做，而且告诉读者为什么这样做。本书从基本概念讲起，由浅入深、条理清楚，对一些关键技术作出了详尽的分析和说明。例如，本书用较大篇幅，对 MFC 实用而复杂的文档/视图技术、对话框和控件技术等进行了深入、彻底的剖析，这正是广大读者所急需的知识。通过学习本书，读者将从概念上理解和掌握 Windows 编程技术。有了这些概念和思路，要想再学习别的像 VB 或 Delphi 这样的开发工具，那将是轻而易举的。

本书的作者不但经验丰富，而且重要的是按照中国人的思路写出来的，因此本书符合国内读者的阅读和思维习惯。本书也考虑到国内的一些实际情况。例如，考虑到国内很多人

都是 Borland C++的用户，在本书的附录中介绍了从 Borland C++ OWL 向 Visual C++ MFC 过渡的方法。

本书首先介绍了 Windows 和面向对象编程的基本概念和 Visual C++ 5.0 的概况。然后逐步深入，介绍了各种界面编程技术，包括窗口、菜单、工具条和状态栏、对话框和控件、文档/视结构、GDI 绘图、鼠标和键盘处理等等。在书的后半部分介绍了一些实用的高级技术，包括创建用户模块、用 ODBC 和 DAO 进行数据库编程、多媒体技术以及多线程和通信程序设计等。在本书的附录中，还介绍了从 OWL 迁移到 MFC 的方法、Visual C++在 Internet 上的热门站点以及 VC 程序的调试方法和常见错误处理。

全书共有 12 章，第 1,2,3,7,8,9 章和附录由王海云编写，第 4、5、6、10、11、12 章由张颖峰编写。书中所有程序可以从 http://account. njupt.edu.cn\vcbook.htm 处下载。在本书编写过程中，得到胡建彰教授和刘涛同学的支持、帮助和鼓励，在此向他们表示衷心感谢。

由于时间仓促，编者水平和能力有限，缺点和错误在所难免，恳请读者赐教，不胜感谢。

编　者

1998 年 7 月于南京

目　次

第 1 章　Windows 编程和面向对象技术

Microsoft Windows 是一个基于 Intel x86 微处理芯片个人计算机上的具有图形用户接口的多任务和多窗口的操作系统，它是对 MS－DOS 操作系统的扩展和延伸。与 MS－DOS 操作系统相比，它有许多优越之处：首先，它提供了比 MS－DOS 字符界面更为直观、友好的图形用户界面；其次，它能一次运行多个程序，方便用户操作，提高了机器的利用率；再次，Windows 环境下的应用程序具有一致的外观和用户接口，用户只需要熟悉其中一两个程序，就可以触类旁通学会使用别的 Windows 应用程序。另外，Windows 还具有更好的虚拟内存管理和设备无关特性等等。由于 Windows 具有以上突出优点，Windows 平台上的软件开发和程序设计已成主流。本章首先介绍 Windows 发展简史，然后分析 Windows 操作系统的特点以及 Windows 程序设计的关键概念，最后介绍 Windows 程序设计的过程及开发工具。

1.1　Windows 发展历史

Windows 起源可以追溯到 Xerox 公司进行的工作。1970 年，美国 Xerox 公司成立了著名的研究机构 Palo Alto Research Center(PARC)，从事局域网、激光打印机、图形用户接口和面向对象技术的研究，并于 1981 年宣布推出世界上第一个商用的 GUI(图形用户接口)系统：Star 8010 工作站。但如后来许多公司一样，由于种种原因，技术上的先进性并没有给它带来它所期望的商业上的成功。

当时，Apple Computer 公司的创始人之一 Steve Jobs，在参观 Xerox 公司的 PARC 研究中心后，认识到了图形用户接口的重要性以及广阔的市场前景，开始着手进行自己的 GUI 系统研究开发工作，并于 1983 年研制成功第一个 GUI 系统：Apple Lisa。随后不久，Apple 又推出第二个 GUI 系统 Apple Macintosh，这是世界上第一个成功的商用 GUI 系统。当时，Apple 公司在开发 Macintosh 时，出于市场战略上的考虑，只开发了 Apple 公司自己的微机上的 GUI 系统，而此时，基于 Intel x86 微处理器芯片的 IBM 兼容微机已渐露峥嵘。这样，就给 Microsoft 公司开发 Windows 提供了发展空间和市场。

Microsoft 公司早就意识到建立行业标准的重要性，在 1983 年春季就宣布开始研究开发 Windows，希望它能够成为基于 Intel x86 微处理芯片计算机上的标准 GUI 操作系统。它在 1985 年和 1987 年分别推出 Windows 1.03 版和 Windows2.0 版。但由于当时硬件和 DOS 操作系统的限制，这两个版本并没有取得很大成功。此后，Microsoft 公司对 Windows 的内存管理、图形界面做了重大改进，使图形界面更加美观并支持虚拟内存。Microsoft 于 1990 年 5 月份推出 Windows3.0 并一炮打红。这个“千呼万唤始出来”的操作系统一经面世便在商业上取得惊人的成功：不到 6 周，Microsoft 公司销出 50 万份 Windows3.0 拷贝，打破了任何软件产品的 6 周销售记录，从而一举奠定了 Microsoft 在操作系统上的垄断地位。

一年之后推出的 Windows3.1 对 Windows 3.0 作了一些改进，引入 TrueType 字体技术，这是一种可缩放的字体技术，它改进了性能；还引入了一种新设计的文件管理程序，改进

了系统的可靠性。更重要的是增加对象链接及嵌入技术(OLE)和多媒体技术的支持。Windows3.0 和 Windows3.1 都必须运行于 MS－DOS 操作系统之上。

随后,Microsoft 借 Windows 东风,于 1995 年推出新一代操作系统 Windows95(又名 Chicago),它可以独立运行而无需 DOS 支持。Windows95 是操作系统发展史上一个里程碑式的作品,它对 Windows3.1 版作了许多重大改进,包括:更加优秀的、面向对象的图形用户界面,从而减轻了用户的学习负担;全 32 位的高性能的抢先式多任务和多线程;内置的对 Internet 的支持;更加高级的多媒体支持(声音、图形、影像等),可以直接写屏并很好地支持游戏;即插即用,简化用户配置硬件操作,并避免了硬件上的冲突;32 位线性寻址的内存管理和良好的向下兼容性等等。以后我们提到的 Windows 一般均指 Windows95。

1.2 Windows 操作系统特点

Windows 之所以取得成功,主要在于它具有以下优点:

· 直观、高效的面向对象的图形用户界面,易学易用:

从某种意义上说,Windows 用户界面和开发环境都是面向对象的。用户采用"选择对象－操作对象"这种方式进行工作。比如要打开一个文档,我们首先用鼠标或键盘选择该文档,然后从右键菜单中选择"打开"操作,打开该文档。这种操作方式模拟了现实世界的行为,易于理解、学习和使用。

· 用户界面统一、友好、漂亮:

Windows 应用程序大多符合 IBM 公司提出的 CUA (Common User Acess)标准,所有的程序拥有相同的或相似的基本外观,包括窗口、菜单、工具条等。用户只要掌握其中一个,就不难学会其他软件,从而降低了用户培训学习的费用。

· 丰富的设备无关的图形操作:

Windows 的图形设备接口(GDI)提供了丰富的图形操作函数,可以绘制出诸如线、圆、框等的几何图形,并支持各种输出设备。设备无关意味着在针式打印机上和高分辨率的显示器上都能显示出相同效果的图形。

· 多任务:

Windows 是一个多任务的操作环境,它允许用户同时运行多个应用程序,或在一个程序中同时做几件事情。每个程序在屏幕上占据一块矩形区域,这个区域称为窗口,窗口是可以重叠的。用户可以移动这些窗口,或在不同的应用程序之间进行切换,并可以在程序之间进行手工和自动的数据交换和通信。

虽然同一时刻计算机可以运行多个应用程序,但仅有一个是处于活动状态的,其标题栏呈现高亮颜色。一个活动的程序是指当前能够接收用户键盘输入的程序。

1.3 Windows 应用程序设计的特点

如前所述,Windows 操作系统具有 MS－DOS 操作系统无可比拟的优点,因而受到了广大软件开发人员的青睐。但是,熟悉 DOS 环境下软件开发的程序员很快就会发现,Windows编程与 DOS 环境下编程相比有很大的不同。Windows 要求以一种全新的思维方

式进行程序设计，主要表现为以下几点：

1.3.1　事件驱动的程序设计

传统的 MS－DOS 程序主要采用顺序的、关联的、过程驱动的程序设计方法。一个程序是一系列预先定义好的操作序列的组合，它具有一定的开头、中间过程和结束。程序直接控制程序事件和过程的顺序。这样的程序设计方法是面向程序而不是面向用户的，交互性差，用户界面不够友好，因为它强迫用户按照某种不可更改的模式进行工作。它的基本模型如图 1.1 所示。

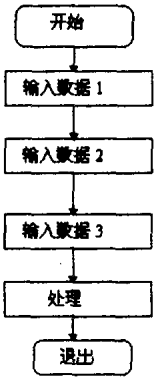

图 1.1　过程驱动模型

事件驱动程序设计是一种全新的程序设计方法，它不是由事件的顺序来控制，而是由事件的发生来控制，而这种事件的发生是随机的、不确定的，并没有预定的顺序，这样就允许程序的的用户用各种合理的顺序来安排程序的流程。对于需要用户交互的应用程序来说，事件驱动的程序设计有着过程驱动方法无法替代的优点。它是一种面向用户的程序设计方法，它在程序设计过程中除了完成所需功能之外，更多的考虑了用户可能的各种输入，并针对性的设计相应的处理程序。它是一种“被动”式程序设计方法，程序开始运行时，处于等待用户输入事件状态，然后取得事件并作出相应反应，处理完毕又返回并处于等待事件状态。它的框图如图 1.2 所示。

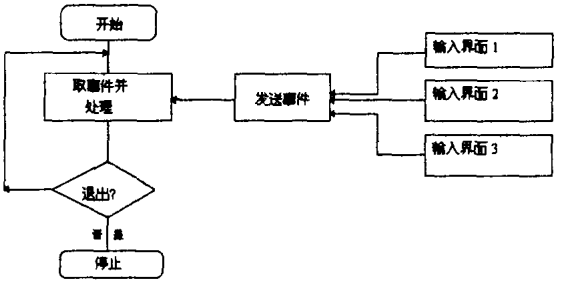

图 1.2　事件驱动程序模型

在图中，输入界面 1～3 并没有固定的顺序，用户可以随机选取，以任何合理的顺序来输入数据。

1.3.2 消息循环与输入

事件驱动围绕着消息的产生与处理展开，一条消息是关于发生的事件的消息。事件驱动是靠消息循环机制来实现的。

消息是一种报告有关事件发生的通知。

消息类似于 DOS 下的用户输入，但比 DOS 的输入来源要广，Windows 应用程序的消息来源有以下四种：

(1)输入消息：包括键盘和鼠标的输入。这一类消息首先放在系统消息队列中，然后由 Windows 将它们送入应用程序消息队列中，由应用程序来处理消息。

(2)控制消息：用来与 Windows 的控制对象，如列表框、按钮、检查框等进行双向通信。当用户在列表框中改动当前选择或改变了检查框的状态时发出此类消息。这类消息一般不经过应用程序消息队列，而是直接发送到控制对象上去。

(3)系统消息：对程序化的事件或系统时钟中断作出反应。一些系统消息，像 DDE 消息(动态数据交换消息)要通过 Windows 的系统消息队列，而有的则不通过系统消息队列而直接送入应用程序的消息队列，如创建窗口消息。

(4)用户消息：这是程序员自己定义并在应用程序中主动发出的，一般由应用程序的某一部分内部处理。

在 DOS 应用程序下，可以通过 getchar()、getch()等函数直接等待键盘输入，并直接向屏幕输出。而在 Windows 下，由于允许多个任务同时运行，应用程序的输入输出是由 Windows来统一管理的。

Windows 操作系统包括三个内核基本元件：GDI，KERNEL，USER。其中 GDI(图形设备接口)负责在屏幕上绘制像素、打印硬拷贝输出，绘制用户界面包括窗口、菜单、对话框等。系统内核 KERNEL 支持与操作系统密切相关的功能：如进程加载，文本切换、文件 I/O 以及内存管理、线程管理等。USER 为所有的用户界面对象提供支持，它用于接收和管理所有输入消息、系统消息并把它们发给相应的窗口的消息队列。消息队列是一个系统定义的内存块，用于临时存储消息；或是把消息直接发给窗口过程。每个窗口维护自己的消息队列，并从中取出消息，利用窗口函数进行处理。框图如下：

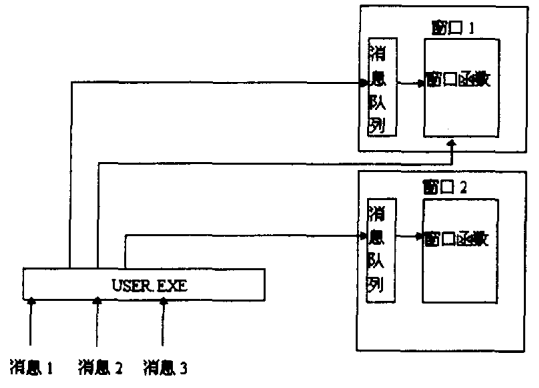

图 1.3 消息驱动模型

1.3.3　图形输出

Windows 程序不仅在输入上与 DOS 程序不同，而且在程序输出上也与 DOS 有着很大不同，主要表现为以下几方面：

(1)DOS 程序独占整个显示屏幕，其他程序在后台等待。而 Windows 的每一个应用程序对屏幕的一部分进行处理。DOS 程序可以直接往屏幕上输出，而 Windows 是一个多窗口的操作系统，由操作系统来统一管理屏幕输出；每个窗口要输出内容时，必须首先向操作系统发出请求(GDI 请求)，由操作系统完成实际的屏幕输出工作。

(2)Windows 程序的所有输出都是图形。Windows 提供了丰富的图形函数用于图形输出，这对输出图形是相当方便的，但是由于字符也被作为图形来处理，输出时的定位要比 DOS 复杂得多。

比如，在 DOS 字符方式下，我们可以写出如下程序用于输出两行文字：

```
printf("Hello, \n");
printf("This is DOS program. \n");
```

而在 Windows 下要输出这两行文字所做的工作要复杂得多。因为 Windows 输出是基于图形的，它输出文本时不会像 DOS 那样自动换行，而必须以像素为单位精确定位每一行的输出位置。另外，由于 Windows 提供了丰富的字体，所以在计算坐标偏移量时还必须知道当前所用字体的高度和宽度。

(3)Windows 下的输出是设备无关的。在 DOS 下编写过 Foxpro 程序的读者常常会有这样的体会，在编写打印报表程序时，要针对不同的打印机在程序中插入不同的打印控制码，用以控制换页、字体设置等选项。这样的程序编写起来繁琐，而且不容易移植(因为换一台不同型号的打印机就要重新修改程序)。而 Windows 下的应用程序使用图形设备接口(GDI)来进行图形输出。GDI 屏蔽了不同设备的差异，提供了设备无关的图形输出能力，Windows 应用程序只要发出设备无关的 GDI 请求(如调用 Rectangle 画一个矩形)，由 GDI 去完成实际的图形输出操作。对于一台具有打印矩形功能的 PostScript 打印机来说，GDI 可能只需要将矩形数据传给驱动程序就可以了，然后由驱动程序产生 PostScript 命令绘制出相应的矩形；而对于一台没有矩形输出功能的点阵打印机来说，GDI 可能需要将矩形转化为四条线，然后向驱动程序发出画线的指令，在打印机上输出矩形。当然，这两种输出在用户看来并没有什么区别。Windows 的图形输出是由图形设备接口(GDI)来完成的，GDI 是系统原始的图形输出库，它用于在屏幕上输出像素、在打印机上输出硬拷贝以及绘制 Windows用户界面。

GDI 提供两种基本服务：创建图形输出和存储图像。GDI 提供了大量用于图形输出的函数，这些函数接收应用程序发出来的绘图请求、处理绘图数据并根据当前使用设备调用相应的设备驱动程序产生绘图输出。这些绘图函数分为三类：一是文字输出，二是矢量图形函数，用于画线、圆等几何图形，三是光栅(位图)图形函数，用于绘制位图。

GDI 识别四种类型的设备：显示屏幕、硬拷贝设备(打印机、绘图机)、位图和图元文件。前两者是物理设备，后两者是伪设备。一个伪设备提供了一种在 RAM 里或磁盘里存储图像的方法。位图存放的是图形的点位信息，占用较多的内存，但速度很快。图元文件保存的是 GDI 函数的调用和调用参数，占用内存较少，但依赖于 GDI，因此不可能用某个设备来创