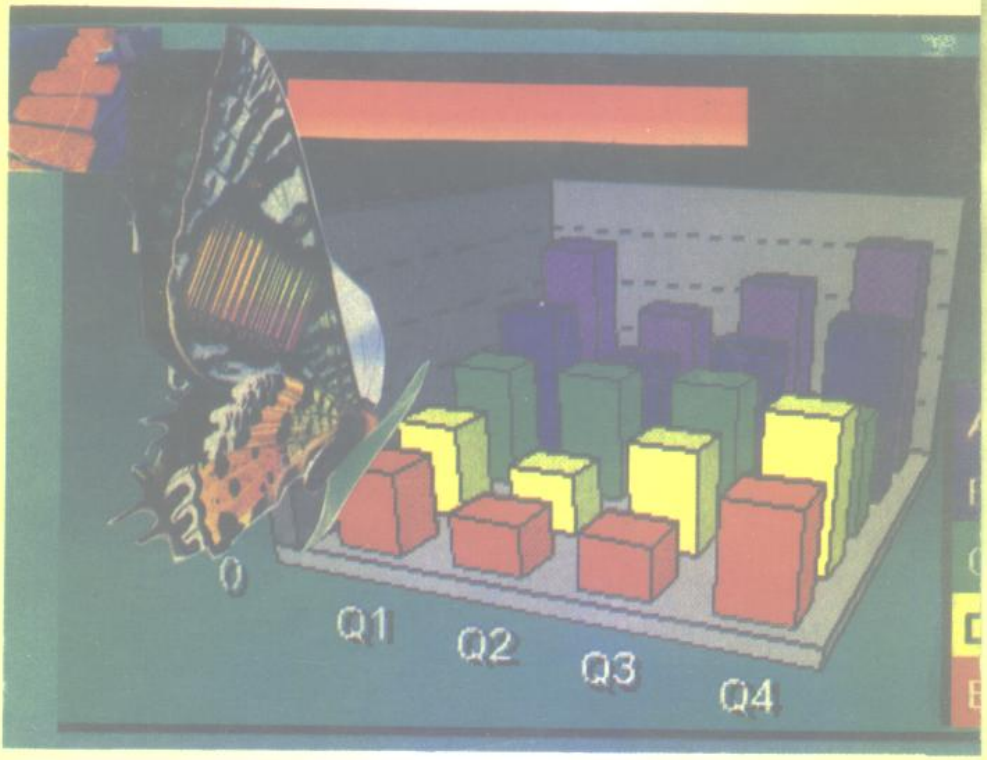
《现代电子技术》增刊

Windows与Windows NT
程序设计

何自强
顾垒 著



陕西电子杂志社

NT

TP316
H38

384341

Windows 与 Windows NT 程序设计

何自强　顾垒　著

陕西电子杂志社

384341

windows 与 windows NT 程序设计
何志强 顾 垒 著
陕西电子杂志社出版
陕西广播电视印刷厂印刷
* *
开本 787×1092 1/16 印张:26 字数:624 千字
1995 年 3 月第 1 版 1995 年 3 月第 1 次印刷
印数:1—5000 册
定价:22.00 元

TS191/2003

内 容 简 介

本书讨论了 Windows 程序设计的基本原理，内容包括对 Windows 编程要素的描述和示例，以及对 Windows 中几个最基本的编程问题的解答。本书也介绍了怎样编写 Windows-NT 应用程序，并重点讨论了 WindowsNT 中的控制台和多任务。为了说明 C++语言是如何对 Windows 编程提供支持的，本书最后两章还介绍了两个重要类库的用法：Microsoft 基础类库（FCL）和 Borland ObjectWindows 环境。

读完本书之后，读者就能够对所有常见的 Windows 操作进行编程，而且在编写自己的应用程序或进一步探索 Windows 方面将不会再有困难。

本书可作为大中专院校师生的教材，也可供 Windows，MS C/C++以及 Borland C++程序员参考。

前　言

Windows 是使个人计算机迈入下一个世纪的操作系统，它非常灵活，功能也很强大。它也是现有的、经过最彻底的深思熟虑而设计出的操作系统之一，而且它的各个子系统在逻辑上是一致的。因此，一旦掌握了它的要点并且生成了可重用的代码，用它编程就成为一种乐趣了。事实上，如果读者曾在诸如 DOS 之类的系统下编过程序的话，就会发现 Windows 对于一致性的重视是一个显著的改进。

在 Windows 中编程是很复杂的。就编程这一点而言，无论用什么标准来看，Windows 都是一个复杂的操作系统，其中包含了大量的子系统，而且有时会使那些看来非常容易的任务变得极为复杂。因此，本书不会有论题短缺之虞。但是，尽管如此，本书仍主要侧重于 Windows 中那些最基本的和最常用的子系统。特别是，它将解释如何生成一个 Windows 框架程序，如何管理消息，如何生成菜单和消息框，如何使用对话框（包括通用对话框），如何管理图形和文本，以及如何访问文件。

本书也解释了怎样编写 WindowsNT 应用程序，并且用两章的篇幅来探讨与 Windows NT 有关的主题：控制台和多任务。此外，关于 WindowsNT 的“标注”遍布全书各处，它们解释了 16 位 Windows 与 Windows NT 之间的具体差异，也为那些正在向 WindowsNT 移植程序的程序员提供了一个快速参考。注意到基于 C++的类库正变得越来越流行，所以本书也介绍了两个最重要的类库：Microsoft 基础类库以及 Borland ObjectWindows 类库。

本书适合于任何一个想要学习 Windows 编程的程序员阅读。它并不假设读者曾编写过 Windows 程序，然而，它确实假设读者是一位熟练的 C/C++程序员。如果读者是一位 C 语言初学者，我们建议先用一些时间去学习它，因为许多 Windows 结构都用到了复杂的 C 程序设计技巧。特别是，读者应对如何使用指针、结构及联合相当熟悉。除了第十五章和第十六章之外，本书中的其他代码都是 C 语言代码，而没有使用 C++特征。第十五章和第十六章使用了 C++，因为它们讨论的是由 Microsoft 和 Borland 公司提供的 C++类库。

尽管本书不假定读者熟悉操作系统原理的基本内容，但若读者具有操作系统基础方面的背景知识或经验，那么学习编写 Windows 程序将会容易一些，因为 Windows 毕竟太复杂了。

最后要注意的是，如果读者从未编写过 Windows 风格的程序，则请耐心些。Windows 风格的程序与读者过去编写的 C 程序是极不相同的。不管怎样，到了第四章末尾，Windows 程序的整体结构就会很清楚了，生成一个 Windows 程序也不会再有困难。

本书中的程序是用 Microsoft C/C++编译器（7.0）及 Borland C/C++编译器（3.1）编写、编译并调试的。读者需要这两种编译器之一，或者其他可用来生成 Windows 兼容目标代码的 C/C++编译器。有关 Windows NT 的例子是用 Microsoft Windows NT Developer's Kit C/C++编译器编译的。

第一章到第四章由何自强和顾垒执笔，第五章到第七章由李小明和乾志绚执笔，第八章由莫桂华执笔，第九章到第十三章由韩晶、吴兔和徐小夏执笔，最后三章由江涛执笔。许鑫为本书的机算机录入付出了辛勤的劳动，在此一并致谢。

此为试读，需要完整PDF请访问：www.ertongbook.com

由于时间仓促，加上 Windows NT 中的一些概念尚未形成标准，对书中的不足，请读者指正。

一九九四年十二月
作者于西安

目　录

此为试读,需要完整PDF请访问: www.ertongbook.com

第一章　Windows 编程要素简介

本章讨论 Windows 程序设计，其主要内容如下：首先，以一般方式讨论什么是 Windows？应用程序怎样与它进行交互？每个 Windows 应用程序必须遵循哪些原则？其次开发了一个应用程序框架，这一系列中所有其他 Windows 程序都可以此为基础来开发。正如读者将要看到的，所有 Windows 程序都有某些公共特征，而框架程序中所包含的正是这些公共特征。

1.1　Windows 环境简介

Windows 究竟是什么？在某种程度上，这要看读者是一个终端用户还是一个程序员。从用户的观点来看，Windows 是一个“外壳”，用户可通过它与程序进行交互。然而，从程序员的观点来看，Windows 是一个面向图形的多任务操作系统，它由几百个 API（应用程序接口）函数组成，并支持一种特殊的应用程序设计方法。从程序员的观点来看，Windows 是提供各种相互关联的服务的巨大工具箱。当正确使用的时候，它允许产生共享同一界面的多个应用程序。

Windows 的目标是，一个对该系统仅有基本的熟悉程度的人能够坐下来实际运行任何应用程序，而无需任何先期训练。从理论上讲，如果用户能够运行一个 Windows 程序，他就能够运行所有的 Windows 程序。当然，实际上最有用的程序仍然要求用户受过某种训练，以便更有效地使用这些程序，但至少这些训练可被局限在程序做什么而不是用户应怎样与之交互上。事实上，在一个 Windows 应用程序中，许多代码是为支持用户界面而编写的。

对读者来说，非常重要的是要理解：不是每一个在 Windows 下运行的程序都必须提供一个 Windows 风格的界面，而只有那些利用了 Windows 功能的程序看起来和感觉起来才像一个 Windows 程序。尽管读者可以不理会基本的 Windows 程序设计原则，但最好为此找到足够的理由，因为这些程序的用户可能会因为程序看起来不像一个 Windows 程序而感到恼火。如果读者正在为 Windows 编写应用程序，那么就应遵循普遍接受的 Windows 程序设计原则。

如前所述，Windows 是面向图形的，这意味着它提供了一个图形用户界面。图形硬件和视频模式各不相同，但这些差异都由 Windows 来处理。这意味着，在大多数情况下，用户程序不必关心正在使用的是何种图形硬件或视频模式。

1.1.1　Windows 与 Windows NT

正如读者可能知道的，现在有两种不同的 Windows，一种是 16 位的 Windows，它在 DOS 下运行，且支持 16 位程序设计环境，这就是通常使用的那种 Windows。16 位 Windows

的当前版本是 3.1 版。另一种 Windows 系统是 Windows NT，它是一个独立的操作系统，支持完整的 32 位编程环境。Windows NT 是相当新的，而且在写作本书时它尚未普遍使用。

从编程的角度来看，Windows 3.1 程序和 Windows NT 程序之间的差异相对而言是较小的。本书中第一章到第十二章的例子是设计来在 Windows 3.1 下运行的。Windows NT 编程及实例在第十三章到第十五章讨论。本书侧重于 16 位 Windows 的原因是，在写作本书时，Windows 3.1 是更为常用的编程环境。虽然如此，有关 Windows 3.1 和 Windows NT 的特殊区别在全书中每一适当位置都作了标注。

1.1.2 桌面模型

虽然也有若干例外，但基于窗口的用户界面的要点在于在屏幕上为用户提供一个桌面的“等价物”。在桌面上，用户会发现几张不同的“纸”，一张叠着一张。通常，最上一页下面的各页也有部分可见。在 Windows 中，桌面的等价物就是屏幕，而纸张的等价物就是屏幕上的窗口。在桌面上，用户可能会移动纸张，也许会把某张纸移到顶部，或者改变另外一张纸，使其一部分可见。Windows 也允许对其窗口作同样的操作。当选择了某个窗口时，可使它成为“当前窗口”，这意味着把它放在所有其他窗口的上面。可以放大和缩小窗口，也可以移动它。简言之，Windows 允许用户以控制桌面的方式来控制屏幕。

1.1.3 鼠标接口

与 DOS 不同的是，Windows 允许用户使用鼠标来进行几乎所有的控制、选择和绘图操作。当然，说它“允许”使用鼠标是过于保守的说法。事实上，Windows 界面是为鼠标设计的，但它也允许使用键盘。尽管一个应用程序对鼠标置之不管是可能的，但它在这样做的同时，也就违反了 Windows 的基本设计原则。

1.1.4 图标和图形/图像

Windows 允许（但不是要求）使用图标和位图（bit－map）图形/图像。使用图标和图形/图像的理论依据简单得像一句格言：图片胜过千言万语。

图标是一个较小的符号，代表一些功能或程序。可通过将鼠标移到图标上并双击按钮来激活它。图形/图像通常被用来将信息快速传递给用户。

1.1.5 菜单和对话框

除了标准窗口之外，Windows 也提供了用以完成特殊目的的窗口，其中最常用的窗口就是菜单和对话框。简单地说，菜单就是其中含有许多项的特殊窗口，用户可从中选择所需的项。但是，在建立 Windows 程序时，并不是必须提供菜单选择，程序员可以利用内置的菜单功能简单地建立一个标准菜单。

对话框所提供的交互操作比菜单要复杂。例如，应用程序可利用对话框来输入文件名。

1.2 Windows 怎样与用户程序进行通信

当为别的操作系统编写程序时，通常是由用户程序为了与操作系统进行交互而作初始化工作。例如，在一个 DOS 程序中，通常是由这个程序来完成诸如请求输入/输出这类事情。不同之处在于，以“传统方式”编写的程序是调用操作系统，而不是由操作系统调用该程序，然而 Windows 在大多数场合是以相反的方式工作的，即由 Windows 调用用户程序，该过程是如下运作的：一个 Windows 程序一直等待，直到 Windows 发送一条“消息”给它，消息是由 Windows 调用的一个特殊函数传送给用户程序的。一旦收到一条消息，用户程序便被期待作出适当的动作。当对一条消息作出响应时，用户程序可以调用一或多个 Windows API 函数，而与此有关的初始化工作仍是由 Windows 完成的。

Windows 可能发送许多不同种类的消息给用户程序。例如，每当鼠标按钮在一个属于用户程序的窗口内被按下时，Windows 就会发送一条“鼠标按下”消息给用户程序；每当一个属于用户程序的窗口需要重画时，另一类消息会被发送给用户程序；当用户程序需要接受输入时，每当用户按下一个键，就有一条消息被发送给用户程序。请记住，对用户程序而言，消息是随机到达的，这就是 Windows 程序类似中断驱动程序的原因。程序无法知道下一条消息是什么。

最后，在处理之前，发送给用户程序的消息被存储在与用户程序相关的一个“消息队列”中。因此，不会由于用户程序正忙于处理另外的消息而导致消息被丢失。消息会在队列中等待，直到用户程序准备好处理它为止。

1.3 Windows 与多任务处理

如前所述，Windows 是一个多任务操作系统。作为一个多任务操作系统，在采用“非占先式多任务”这一点上，它是有些独特的。这意味着在系统中运行的每个程序都保留着对处理器的占用，直到放弃为止，这与其他操作系统通过采用基于时间片的占先式任务切换来实现的那一类多任务是根本不同的。在占先式任务切换中，操作系统简单地停止运行一个程序，并以一种轮流的方式转而去运行另一个程序。请记住，Windows 不是这样工作的。一个 Windows 程序必须主动放弃 CPU。

注意：不像 Windows 3.1，Windows NT 采用占先式多任务，并且大多数程序员认为这是一个重大的改进（Windows NT 多任务将在第十四章讨论）。尽管 Windows 与 Windows NT 采用两种不同形式的多任务，但程序员为它们编写程序的一般方法是相同的。

Windows 程序必须遵循的最重要的原则之一是：必须在暂停运行前将控制交回给 Windows。这样，Windows 才能将控制交给另一任务。如读者将要看到的，将控制返回给 Windows 通常是很容易的。不管怎样，请记住，一个程序独占处理器，从而有效地终止其他任务也是可能的。

此为试读，需要完整PDF请访问：www.ertongbook.com