Windows 高级编程

面向对象的思想、方法和实例

孙晓涛 等编著

Up to BC 5.0

All the Source Code Included

Create object-oriented Win 3.x, Win NT and Win 95 programs quickly and efficiently

西北工业大学出版社

Windows 高级编程

——面向对象的思想、方法和实例

孙晓涛　孙玉霖　施笑畏　编著

西北工业大学出版社

1997年10月　西安

(陕)新登字 009 号

【内容简介】 本书全面系统地介绍了使用OWL 2.5进行面向对象的Windows 95、Windows NT和Windows 3.x应用程序开发的原理和方法。OWL(Object Windows Class Library)应用技术是目前最为流行的几种面向对象Windows开发技术之一,它能够极大地简化程序设计过程,加快开发的进度。本书从最基本的Windows编程技术入手,循序渐进地介绍OWL的程序设计技术。全书内容简明,例题典型、丰富,实用性强,书中所有应用程序均以Borland C++ 4.5作为集成开发平台,并主要选用Windows 95作为应用系统,所有应用程序均已在Borland C++ 5.0集成环境下调试运行通过。本书可作为大专院校计算机专业及其他专业学生的教学用书或参考书,也可供从事计算机程序设计的工程人员和广大计算机爱好者参考阅读。

Windows 高级编程

——面向对象的思想、方法和实例

孙晓涛 等编著

责任编辑 王俊轩

责任校对 钱伟峰

*

(710072 西安市友谊西路127号 电话 8493844)

全国各地新华书店经销

西北工业大学出版社印刷厂印装

ISBN 7-5612-0981-9/TP·135

*

开本:787×1092 毫米 1/16 印张:31.125 字数:755 千字

1997年10月第1版 1997年10月第1次印刷

印数:1—6 000册 定价:38.00 元

前　言

计算机应用的发展可以说是日新月异，各种新技术新方法层出不穷，这就要求人们能够站在某些领域的前沿，把握当前和今后的新技术和发展方向。

在操作系统领域，随着 Windows 95 作为一种全新的完整的操作系统的推出和广泛应用，人们越来越认识到 Windows，尤其是 32 位的 Windows 95 等“完全”的操作系统，将是目前以至于将来很长一段时间内微机上的首选操作系统。编写准确无误、界面美观的窗口应用程序，已成为摆在软件开发人员面前的一项重要任务。

面向对象编程技术是进行窗口应用程序开发的最为有效的工具，是目前最为流行的软件设计思想和方法。它以其结构清晰、易于维护、可移植性好、查错能力强等诸多优点，成为目前最行之有效的软件开发技术。

本书是作者积多年 C＋＋应用程序设计的经验，经整理提炼后完成的。书中的内容力求简明、实用，对于概念、思想的论述力求简洁准确，对于实际的编程方法则予以较详细的说明。

书中列出了完备的例子程序，对每个例子程序都附有详尽的注解。所有的例子程序都在 Borland C＋＋ 5.0 集成开发环境下调试运行通过。本书主要以 Windows 95 应用程序作为例子程序的运行结果，读者也可以生成其它的 Windows 应用程序而无须改变例子源代码。

本书主要讲述使用 Borland 公司最新版的 OWL 2.5 进行面向对象的应用程序设计，但是读者将会看到，书中所述的思想和方法，完全可以应用于编写 Visual C＋＋以及 Java 等其它面向对象的应用程序。

本书的第一、三、十二章由孙玉霖编写，本书的第二、五章以及第六章和第十六章的部分小节由施笑畏编写，书中其它各章节由孙晓涛编写。书中各章例子程序的编写和调试工作由孙晓涛完成。

由于作者学识水平有限，书中的缺点和差错在所难免，诚恳地希望读者批评指正。

作　者

1997 年 4 月于西安

目　录

简　介

Windows 是一种具有窗口和多任务功能的操作系统。从 1990 年推出的 Windows 3.0，到 1992 年推出的 Windows 3.1，直至 1995 年推出的 Windows 95，它的发展非常迅猛。Windows 以其漂亮的用户界面、简便易学的操作、强大的功能而受到了广大的计算机用户的欢迎。

0.1　编写本书的目标

本书旨在指导读者学习和掌握 Windows 面向对象的高级编程方法，主要针对 32 位的 Windows 95、Windows NT 以及 16 位的 Windows 3.x。

用面向对象的方法开发 Windows 程序具有极大的优越性，它能够极大地简化程序设计过程。这主要得益于 Windows 软件开发工具提供的面向对象的手段(如 Borland C++提供了 Object Windows 类库)，它将多种窗口的产生和操作行为都封装于一个类中，因而应用程序就不必再对这些行为进行定义和描述。

另一方面，面向对象的方法涉及到许多新概念和新思想，对于许多初学者来说要很好地掌握这一方法是比较困难的。本书将较全面地给出面向对象的思想和方法，重点介绍使用 BorlandC++ 4.5 的 OWL 2.5 类库及其以上版本进行面向对象的 Windows 程序设计，并给出全面的、可靠的程序实例。

熟悉 DOS 编程的人，在初次接触到 Windows 编程时往往会不知所措，因此本书将以较短篇幅对 Windows 编程的基础知识予以介绍。考虑到一些读者对用 C 语言编写 Windows 程序的偏爱，本书的前两章采用 C 语言方式介绍 Windows 应用程序。

0.2　本书适用的操作系统

本书主要针对 Windows 95 编程，但书中论述的原理和方法同样适用于 Windows 3.x 和 Windows NT 系统。读者将会看到，Windows 95 编程方式与 Windows 3.x 和 Windows NT 的编程方式是相同的，在一个 Windows 系统下编写的应用程序几乎可以不改变地应用于另一个 Windows 系统。唯一要注意的一点是，虽然 16 位的 Windows 3.x 应用程序在 Windows 95 下可以很好地运行，但是也有极少的 32 位 Windows 95 应用程序却不能用于 16 位的 Windows 3.x 系统。但总的说来，本书论述的 Windows 编程的思想、原理以及利用 OWL 进行应用开发的方法、过程适用于目前所有的 Windows 系统。

0.3　本书所用的程序开发环境

本书中的所有代码都是在 Borland C++ 4.5 集成开发环境下经细心地编写、编译和测试通过的，所有代码在 Borland C++ 5.0 集成开发环境中也完全可以编译和运行通过。

之所以选择 BC(Borland C++之简)作为编程工具,是因为 BC 提供了丰富的 OWL 类库,以及强大的可视化资源编辑工具 Resource Workshop,它能够提供极其方便的手段以生成各种 Windows 资源。

读者也可以选择其它如 Visual C++ 等编程环境进行 Windows 程序设计,虽然在程序的编写格式上会有一些差别,但是编程的基本思想和概念,如消息响应和如何继承现有类库中类的功能等,都是完全相同的,可以将本书中的所有例子很方便地改写为 Visual C++程序。

本书大部分内容都是关于用 Borland C++ 4.5 提供的 OWL 2.5 类库和 Borland C++ 5.0 提供的更高版本的 OWL 类库进行面向对象程序开发的方法,书中提供了充分有效的例子程序。OWL 2.5 及其以上版本的类库与 Borland C++ 3.1 提供的 OWL 1.0 类库在规模、使用方法等方面有很大的差别:类库中类的数目大为增加,各个类的相互关系更为复杂;许多方法,包括基本方法如主窗口的生成和初始化、消息响应机制等都与 OWL 1.0 的有很大的差别。这些,都增加了初学者,包括那些曾经在 Borland 3.1 环境下用 OWL 1.0 进行面向对象程序开发的程序员,学习使用 OWL 2.5 编程的难度。在目前国内尚没有全面和细致地关于开发 Borland C++ 4.5 和 Borland C++ 5.0 面向对象应用程序的实用性指导书的情况下,本书将会给读者以实用有效和快速的帮助。

0.4 本书的读者

本书采用循序渐进的方法介绍 Windows 编程及面向对象编程的思想和方法,读者只要掌握 C 语言基本编程方法即可阅读此书。如果读者曾经使用 OWL 1.0 类库(挂在 Borland C++ 3.1开发环境中)或 OWL 2.x 类库(挂在 Borland C++ 4.0 以上版本)开发过 Windows 应用程序,则可以跳过本书前两章内容阅读,而不会影响系统、方便地掌握 Windows 的编程思想和方法。

本书可作为大专院校计算机专业及其他相关专业师生的教学用书或参考书,也可供从事计算机程序设计和软件开发的工程技术人员及广大计算机爱好者使用。

第一章　Windows 编程基础

本章重点：

- Windows 编程的特点
- 基本的 Windows 程序的框架
- 重要的 Windows 程序设计术语
- Windows 程序的命名规则
- 如何用 Borland C++集成环境开发 Windows 程序

本章的目的，是为了使初学者尽快地掌握 Windows 编程的基本思想和方法，在进行 Windows编程时摒弃传统的顺序和过程驱动的方法，掌握 Windows 消息驱动编程技术。

本章所论述的概念和思想是以后各章的基础，读者花较长的时间阅读本章会对以后的学习有较大的帮助。

1.1　Windows 编程的特点

在 DOS 系统下编程时，计算机的所有资源都归用户所有。程序的每一个动作也都由程序编制人员控制，如键盘操作、显示器管理、鼠标、串并口、内存等。程序员知道程序每一步在干什么，如用户什么时候按下了键盘或鼠标、怎样存储数据等。而对 Windows 系统，这一方式就行不通了，初学 Windows 编程的人往往对一个 Windows 程序很难理解，他们可能不知道程序中的某些部分是干什么用的，而且程序中的各个部分看起来似乎没有什么联系。Windows 是一个多任务、多窗口操作系统，它允许多个应用程序同时运行（请注意：计算机主机中只有一个 CPU，实际上在某一时刻还是只能做一件事，Windows 只是将 CPU 分配给同时运行的各个应用程序使用，而从一个应用程序窗口切换到另一个窗口的速度又非常快，所以感觉上好像是在同一时间运行的），如果一个应用程序一直占用着某些资源，其它应用程序就无法得到这些资源，也就无法实现多任务。因此，这些资源就由 Windows 来管理，包括显示器、键盘、鼠标、打印机和串行端口的管理，以及内存管理、程序执行和调度。应用程序不直接和这些硬件打交道，一旦有一个硬件动作，如键盘式鼠标操作，Windows 会将相应的消息传给相关的应用程序，由应用程序的窗口函数或消息响应函数（对应于 C++编程）来进行相应处理。

1.1.1　消息驱动思想

消息是一个有特定含义的 16 位整数（对于 32 位 Windows 系统，如 Windows 95，消息是 32 位整数），这个特定的含义由 Windows 规定（定义在 Window.h 中）。尽管可以使用数字值来对应于一个消息，如可以在 Windows 3.x 中用 0x0201 表示“鼠标的左键按下”，但在实际中很少这样做。Windows 用一个有意义的字符串表示一个消息，并将它们作为宏定义包含在 Window.h 中。上面提到的消息在 Window.h 中被定义为：

```
# define  WM _LBUTTONDOWN 0x0201
```

在以后的程序中，就可以用 WM _LBUTTONDOWN 来表示“鼠标左键按下”。这样表示，既可以增加程序的可读性，又可以增强程序的通用性。因为在 Window 95 中，对相同的事件如“鼠标按下”，采用的消息宏名与在 Window 3. x 下的完全相同，尽管它们的值不同（一个为 32 位，一个为 16 位），但只要应用程序以宏名来定义消息，程序就不需要作修改。

在 DOS 系统下，编写的程序通过函数调用等手段来调用操作系统，而不是操作系统来调用程序。DOS 程序通常使用顺序的方法编制，有明确的开始和结束，程序的每一个动作都是顺序执行的。

Windows 系统使用的却是另一种方法，是 Windows 系统调用应用程序。应用程序处于等待状态，直至由 Windows 发送给它一个消息（这个消息是由用户或其它应用程序或系统本身产生的）。这种由消息来控制动作的程序称为消息驱动程序。消息驱动程序是由随机事件产生的消息来控制的，而不是由顺序事件控制。消息驱动程序的流程如图 1 - 1 所示：

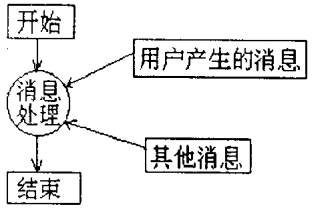

图 1 - 1　消息驱动程序的流程

消息驱动编程思想为程序员提供了许多便利。在整个应用程序中，程序员所完成的工作主要就是对消息进行响应。这种编程思想对于用户来说非常有用，用户不必按照传统的顺序方法编制程序的要求进行操作，这就是说，消息驱动的程序使用户可以控制程序的运行过程。

1.1.2　消息的来源

Windows 发送给应用程序的消息是很多的，可以将消息按来源分为大体的四种：用户产生的消息、其它应用程序产生的消息、应用程序自身产生的消息以及来自系统的消息。

用户产生的消息包括鼠标消息、键盘消息、记时器消息、列表控制消息、菜单消息、滚动杠消息等。最常碰到的消息之一是 WM _COMMAND 消息，它是由用户对菜单进行选择（包括用相应加速键进行选择），或是点击某一控制按钮产生的。

当两个应用程序同时运行且要相互通信时，一个应用程序会接到另一个应用程序发来的消息。例如，当它们需要进行数据交换时，一个应用程序会发出一个消息给另一个应用程序，收方接到消息后，即进行数据准备。待完成后，发出消息，通知接收数据方数据已准备就绪。这种通信方式，对于多任务下多个进程间的相互协调是很有用的。

第三种消息源自于程序自身。在传统的 DOS 程序设计中，通常是通过函数调用的方法将各部分结合在一起，而在以消息驱动的 Windows 程序中，程序设计人员应通过消息来将程序的各部分有机地组织起来。

第四种消息是 Windows 产生的消息，这些消息是在状态变化时产生的。例如，当窗口大小发生改变时，Windows 会产生一个 WM _SIZE 消息，告诉应用程序，窗口的大小发生了改变。

务必记住，对于应用程序，消息是随机到达的。

1.1.3　消息的传递

消息产生之后，由 Windows 系统将消息传送到相应的地方。Windows 内部有两类系统缓冲区：应用程序消息队列缓冲区和系统消息队列缓冲区。凡是鼠标和键盘产生的消息，都放在系统消息队列缓冲区中；应用程序产生的消息则放在应用程序消息队列缓冲区中。可以通过