Windows 3.1和Windows 95系列丛书

Borland C++ Object Windows 程序设计实例

周志国　洪定明等　编

北京航空航天大学出版社

Windows 3.1 和 Windows 95 系列丛书

Borland C++ ObjectWindows

程序设计实例

周志国　洪定明等　　编

北京航空航天大学出版社

图书在版编目(CIP)数据

Borland C++ Object Windows 程序设计实例/周志国等编著. —北京:北京航空航天大学出版社,1995.9

ISBN 7-81012-601-6

Ⅰ.B… Ⅱ.周… Ⅲ.C 语言-程序设计 Ⅳ.TP312C

中国版本图书馆 CIP 数据核字(95)第 11763 号

内 容 简 介

Borland 的 OWL 升级版本 2.X 进一步压缩了原来的 Windows API,功能增强支持 Windows 3.1,Windows 32s,Windows 95 和 Windows NT。本书介绍消息和事件驱动程序设计、图形输出(图形设备接口)和用户界面对象(菜单、窗口和对话框)的程序设计技术,阐述键盘和鼠标输入的编程处理。本书包括快速菜单、标准菜单和高级对话框等最新的界面设计技术。书中程序有很好的中文注释,所有菜单、对话框及提示都用中文。示例程序通用性好,适用于 Microsoft 中文 Windows 3.1 或在中文之星等支持之下的西文 Windows 3.1,也适用于 Windows 95。本书中的完整程序全部在 Borland C++ 4.0、4.5和5.0下通过试调。本书供用 Borland C++的 ObjectWindows Library(OWL) 2.X 类库编写 Windows 程序的 C++程序员学习使用。

●书　　名:**Borland C++ ObjectWindows 程序设计实例**
●编　　者:周志国　洪定明等　　编
●责任编辑:王小青
●责任校对:张韵秋
●出 版 者:北京航空航天大学出版社
●地　　址:北京市海淀区学院路 37 号(100083)　　电话:2015720(发行科电话)
●印 刷 者:铁道部十八局印刷厂
●发　　行:新华书店总店科技发行所
●经　　售:全国各地书店
●开　　本:787×1092　1/16
●印　　张:39.75
●字　　数:1014 千字
●印　　数:4000 册
●版　　次:1995 年 11 月第 1 版
●印　　次:1995 年 11 月第 1 次印刷
●书　　号:ISBN 7-81012-601-6/TP・182
●定　　价:50.00 元

前　言

在 Borland C++中包含两个部分，即 C 语言和C++语言。C 语言是C++语言的前身。C++语言是继承 C 语言发展而来的，是 C 语言的超集。对于习惯用 C 语言的程序员，到目前为止 Borland C++中集成了世界上功能最强、使用最方便，并符合 ANSI C 标准的 C 语言。同样，使用面向对象的语言来编程是当今软件世界的潮流，Borland C++中的面向对象的品质也是一流的。

所谓的 OWL 是指 ObjectWindows Class Library，即 Borland ObjectWindows 类库，它是 Borland C++中用C++编写 Windows 应用程序的类库。OWL 封装了 Windows API(应用程序编程接口)的函数、数据结构和宏，以面向对象的类提供给C++的程序员。

编写过 Windows 应用程序的读者，都有这样的体会：一个编写好的 Windows 应用程序是很容易使用的；然而，开发这样的程序并不是一件容易的事。为此许多程序员就不得不熟悉编写 Windows 应用程序所必需的 500 多个 Windows API 函数。所谓的 API 是指 Application Programming Interface，即应用程序编程接口。在 Windows API 中包括窗口管理、图形设备接口、系统服务和扩展库。窗口管理又包括消息、创建和管理窗口、显示和移动、输入处理、硬件处理、绘图、对话框、滚动、菜单、信息、系统、剪贴板、错误、插入符、光标、挂接、特性列表和矩形等的处理函数。图形设备接口又包括设备场境、画图工具、调色板、绘画属性、映像模式、坐标、区域、剪裁、线输出、椭圆和多边形、位图、设备无关位图、文本、字体、元文件、设备控制和打印机等的处理函数。系统服务又包括模块管理、内存管理、段、操作系统中断、任务、资源管理、字符串处理、原子管理和初始化文件等的函数。扩展库又包括公共对话框、动态数据交换管理、对象链接与嵌入、Shell、工具帮助、数据解压、系统资源紧张测试、文件安装、32 位内存管理、浮点仿真、屏幕保护等功能。Windows API 的功能丰富，令人眼花缭乱，编写程序时无从下手。

Borland 用来解决上面难题的方法是使用 OWL。可重用的类很容易掌握和使用。OWL 充分利用了C++提供的数据抽象的特性，它的使用简化了 Windows 编程。初级程序员可以像用《烹调全书》炒菜那样"照搬照抄"，有经验的C++程序员可以扩展其中的类或者把它们混合进自己的类层次中。

OWL 库提供用于管理 Windows 对象的类，并提供许多可以同时用于 MS - DOS 和 Windows 的通用类。比如，创建和管理文件、字符串、一致的存储和异常处理都是通用的类。

实际上，ObjectWindows Class Library 虚拟表示了每个 Windows API 的功能，包含了用于简化消息处理、诊断及所有 Windows 应用程序都需要重复处理的一般问题的高级代码。OWL 对 Windows API 函数的逻辑组合和增强的优点主要有以下九个方面。

- Windows API 的封装是完整的。OWL 库为所有的经常使用的 Windows API 性能提供支持，包括窗口函数、消息、控件、菜单、对话框、GDI(图形设置接口)对象(字体、画刷、画笔及位图)、对象链接以及多文档界面(MDI)。
- OWL 很容易学。Borland 做了最大的努力以保证 OWL 函数和相关的参数的名字与 Windows API 父类相一致。这极大地减少了有经验的 Windows 程序员想利用能

简化工作的OWL平台的优点时，可能带来的混乱。它也可以帮助初级Windows程序员利用OWL这个Windows API的超集来完成自己的工作。

- C++代码更加有效。当在小内存模式下编译OWL库中的类时，一个应用程序只耗用少许的额外RAM。一个OWL应用程序的执行速度几乎同使用标准Windows API C语言所编写的程序相等。大部分的OWL应用程序的运行速度仅比相应的Windows API C语言程序慢5%——考虑到OWL有助于缩短程序设计开发周期，这是非常值得的。
- OWL库提供了自动消息处理。ObjectWindows Class Library库消除了最易产生编程错误的来源，即Windows API的消息循环。OWL库被设计来自动处理每一条Windows消息。代替使用标准switch-case语句，每一条Windows消息被直接映射到一个进行处理的成员函数。
- OWL库允许自诊断。OWL库实现了自诊断的能力，这就意味着可以在一种易于理解的格式下把有关多个不同对象的信息转储到文件中去，并且验证一个对象的成员变量。
- OWL库具有一个稳定的体系结构。因为事先考虑到ANSI C的throw/catch标准，ObjectWindows Class Library已经实现了一个扩展的异常处理体系结构。这允许一个OWL对象可从标准错误状态恢复过来，这些标准错误包括"内存溢出"错误、无效选项、文件或资源加载问题，在体系结构中的每个成员同ANSI C建议的标准能提供向上兼容。
- OWL库提供动态对象类型。这一特别有用的性能把动态分配对象类型的确定延迟到运行时才进行。这样便可以不必担心一个对象的数据类型要进行处理。因为关于对象类型的信息在运行时被返回，程序员可以减少许多细节处理。
- OWL库可以和谐地同以C语言为基础的Windows应用程序共存。ObjectWindows Class Library最重要的特征是同以C语言为基础的使用Windows API的Windows应用程序共存的能力。在同一个程序中，程序员可以同时使用OWL类和Windows API调用。这样，就可按经验和需要在一个使用OWL的应用程序加进真正的C++面向对象代码。上面透明的环境成为可能，因为在两个体系结构使用了公共的命令协议。这就意味着OWL头文件、类型及全局定义与相应的Windows API名字不冲突。透明的内存管理也是达成上面和谐关系的关键因素之一。
- OWL库可以和MS-DOS应用程序一起使用。ObjectWindows Class Library是专门用来设计Windows应用程序的。然而，许多类提供了使用频繁的I/O文件和串操作所需要的对象。因此，这些通用类既可在Windows应用程序中使用，也可在MS-DOS应用程序中使用。

在理解每个类的成员和用法时，首先要联系本书前面所介绍的C++原理，清楚OWL类的体系结构。它包括继承和派生关系父类和子类之间有许多相似之处)、具有许多相同的数据和函数成员；具有相同父类的子类也有许多相似之处，具有许多同名的数据和函数成员，不同的是具体的数据内容。掌握这一点，在学习和理解OWL时，可以减少记忆量。

Borland C++从4.0开始，是一个很好用的Windows应用程序开发环境。新的IDE(集成开发环境)可在Windows中直接调试程序，工程管理比以前更方便。Borland C++配备了

Turbo Debugger for Win16 和 Turbo Debugger for Win32 用来调试设计 16 位和 32 位的 Windows 应用程序，增强了包括图标、光标、位图、对话框、菜单、热键和字体等的资源设计。特别是在 Borland C++ 4.0、4.5 和 5.0 中，资源设计已得到显著增强，用来设计 Windows 应用程序的界面。Borland C++在增强的 OWL 中封装了 Windows API，包括函数、数据结构和消息等，可以加快读者用C++开发 Windows 应用程序的速度，而减轻读者用C++开发 Windows 应用程序的难度。Borland C++能用来开发 Windows，Win32s，Windows 95 和 Windows NT 应用程序。

本书先介绍 C++的基本原理，为理解用 OWL 设计 Windows 应用程序的技术打好理论基础。然后介绍用 OWL 进行 Windows 消息和事件驱动程序设计、图形输出（图形设备接口）和用户界面对象（菜单、窗口和对话框）的程序设计技术，阐述键盘和鼠标输入的编程处理。本书包括快速菜单、标准菜单和高级对话框等最新的界面设计技术，书中程序有很好的中文注释，所有菜单、对话框及提示都用中文。示例程序通用性好，适用于 Microsoft 中文 Windows 3.1 或在中文之星等支持之下的西文 Windows 3.1。本书中的程序也适用于 Windows 95，通过设置生成的目标类型，可以生成 16 位和 32 位的 Windows 95 应用程序。完整程序全部在 Borland C++ 4.0、4.5 和 5.0 下通过试调。本书供学习使用 Borland C++的 ObjectWindows Library（OWL） 2.X 类库编写 Windows 程序的 C++程序员使用。

编　者

1995 年 1 月于北京

目　　录

第一章 C++概述

C++是一种面向对象(object oriented)的编程语言。在学习有关C++的专门知识前，应当先理解面向对象程序设计中的基础理论。

1.1 什么是面向对象程序设计

面向对象程序设计是一种编程的新方法。自从发明了计算机以来，程序设计的方法有了很大的变化，根本原因是为了适应不断增长的程序复杂性。例如，在计算机刚刚出现的时候，程序设计是通过二进制机器指令来完成的。如果程序长度在几百条指令之内，这种方法还可胜任。随着程序的增大，出现了汇编语言，程序员能用机器指令的助记符来处理更大更复杂的程序。由于程序进一步增大，出现了高级语言，程序员有了处理程序复杂性的工具和手段。广为流行的高级语言无疑是FORTRAN，但用FORTRAN编制的程序的内在清晰性与易理解性比较弱。

到了60年代，结构化程序设计诞生了。C和Pascal语言也正是使用了结构化的程序设计概念，才第一次有可能很容易地编写比较复杂的程序。但即使是使用结构化编程方法，只要工程达到一定的规模，也会对它无法控制。这是由于程序复杂性超过了程序员使用结构化编程技术所能处理的限度。在程序设计开发的每一里程碑上新方法被不断发明，以便能够适应工程复杂性不断增加。在此过程中，每一种新方法都保留了先前方法中的精华。今天，许多工程所具有的复杂性已使结构化方法也不能奏效。为解决这个问题，发明了面向对象的程序设计方法。

面向对象的程序设计吸取了结构化程序设计中的精髓，并且将它们与一些强有力的新概念融合起来，以一种新的方式来看待程序设计任务。面向对象程序设计方法将一个问题分解成该问题的各个相关部分的子模块，然后用语言将这些子模块翻译成被称为对象的自包含单元。

所有面对象程序设计语言都有三个共同特点，即对象、多态性及继承。

1.1.1 对象(object)

对象是面向对象程序设计中最重要的概念。简单地说，一个对象就是既含有数据又含有处理该数据的代码的一个逻辑实体。在一个对象中，有些代码或数据是为该对象私有的，即不能为对象之外的任何部分直接存取。用这种方法，对象就提供了防止程序中其它不相关成份、修改或不正确使用该对象私有部分的有效保护。将代码及数据以此方式联合在一起称为封装(encapsulation)。

简单地说，一个对象就是一个用户自定义的数据类型，读者也许对将包含了数据和代码的对象看成是一种数据结构有些不理解，但是，这种新思想正是面向对象的程序设计方法的核心。

1.1.2 多态性(polymorphism)

面向对象程序设计语言支持多态性。多态性意味着一个名字可被几个相关但多少有些不同的目的所使用。支持多态性的目的是允许一个名字能被用来说明一种具有通用性的操作,根据正在处理的数据,选择一具体实例的执行。例如,可能有一个定义了三种不同类型栈的程序。一个栈用于整型数,另一个用于浮点数,还有一个用于 long 型数。由于引进了多态性的概念,可以为这些栈建立三组称为 push()和 pop()的函数,编译程序将根据调用函数时伴随着的是什么类型的数据而选择正确的例程。在本例中,通用的概念就是将数据从栈中推入、推出。这些函数为每种类型的数据定义了怎样推入、推出的专门方法。

第一个面向对象的程序设计语言是解释型的,只在程序运行时支持多态性。但是,C++是一种编译语言,所以,既可以在程序运行时支持多态性,也可以在程序编译时支持多态性。

1.1.3 继承(inheritance)

继承是一对象获取另一对象某些性质的过程。继承的重要特性是支持类的概念。在现实生活中有这样的例子:红香蕉苹果是苹果类的一部分,苹果又是水果类的一部分,水果又是食品类的一部分。如果不使用类,那么每个对象就都得定义其所有的性质。但是,如果使用了类,则只需定义该对象区别于其类中其它对象的那些性质。它能继承那些和一般类共享的性质。正是这种继承机制才使得一对象能够成为某一般类的实例。

1.2 C++的一些基本原则

由于C++是C的一个超集,所以C程序隐含的也就是C++程序(但在ANSI C和C++之间有几点细微差异),这将使得很少一些C程序不能为C++编译器所编译(这些差异将在后面讨论)。这意味着可从编制看上去像C程序的C++程序。但是,这样做没有充分利用C++的长处,另外,虽然C++允许编制类似C的程序,可是大多数C++程序员却使用C++所独有的一种程序设计风格。由于有些C++程序可以很像C程序,在深入讨论C++细节之前,本节先介绍有关的一些性质,以一个例子作为开始。请看下面的C++程序:

```
#include <stdio.h>
#include <iostream.h>
main(void)
{
    int i;
    char str[80];
    cout << "C++是一种面向对象的语言\n";      // 这是一行注释
    /*  还可以使用原C的翻译  */
    printf("还可以使用函数 printf()\n");
    // 使用>>输入一个数
    cout << "键入一个数:";
```

```
    cin >> i;
    cout <<"数字是"<<i<<"\n";
    // 现在使用<<输出一个数
    cout << " \n";
      // 读入一个字符串
    cout << "键入一个字符串: ";
    cin >> str;
    // 打印出来
    cout << str;
    return 0;
}
```

该程序看上去与一般C程序大不相同。首先,包含了头文件iostream.h。该文件是为支持C++风格的I/O操作而定义的。

接着看上去不同的一行是:

```
cout << "C++是一种面向对象的语言\n";      // 这是一行注释
```

该行引入了两个C++的新性质。第一,语句

```
cout << "C++是一种面向对象的语言\n";
```

使得"C++是一种面向对象的语言"(后跟一回车换行)被显示在屏幕上。在C++中,<<的作用得到了扩展。它仍然是左移位操作符,但当用在本例中所示的地方时,它就是一输出操作符。字cout是与屏幕相关的一标识符(实际上,C++也类似于C,支持I/O重定向,但为便于讨论,可假定cout是指屏幕)。可用cout和<<来输出任一内部数据类型以及字符串。

需要指出的是,仍能使用printf()(如程序所示)或任何其它C的I/O函数。许多程序员只是感到使用cout<<更具有C++的味道。

跟在输出表达式后面的是一条C++注释。在C++中,注释可以由两种方式来定义。第一种注释与C程序相同,即在/*和*/之间的内容为注释。但是,在C++中,还可用//来定义单行注释。当使用//注释符时,其后的一切内容都被编译程序忽略直到行尾。一般来说,C++程序员在需用多行注释时,仍使用第一种注释,在只需一行注释时,则使用第二种。

第二,该程序提示用户输入一个数,该数用下面的语句从键盘上读入:

```
cin >> i;
```

在C++中,>>操作符仍然是右移位操作符。但是,当如上使用时,它还表示让i接受从键盘上输入的一个数值,标识符cin表示键盘。一般说来,可用cin>>来接收任何基本数据类型的输入。

虽然程序中没有表明,但仍能使用诸如scanf()这样的C输入函数,而不必用cin>>。但是,如前所述,许多程序员感到cin>>更具有C++的风格。

程序中另一值得注意的行是:

```
cout << "数字是" << i << "\n";
```

执行该行的结果是显示(如果i的值为100):

```
数字是100
```

后跟一回车换行。一般说来,可连续使用所期望的任意多个<<输出操作符。