Borland C++ for Windows

应用程序设计及实例

李舜酩 石宵宏 王好臣 编著

西安交通大学出版社

Borland C++ For Windows
应用程序设计及实例

李舜酩　石宵宏　王好臣　编著

西安交通大学出版社

内容提要

本书首先对 Borland C++基本技术作了简要介绍，其次对 Windows 的基本结构、基本特点及其应用程序设计的基本方法作了阐述和分析，然后以大部分的篇幅详细介绍了用 Borland C++ 3.1、4.0 版本对 Windows 各方面内容进行应用程序设计的指导思想、设计过程及所能达到的效果，且针对 Windows 各项功能，给出了若干应用程序设计实例。同时也对 Borland C++ 4.5 的一些功能作了介绍。本书具有语言通俗易懂、内容丰富实用、举例生动详实的特点。

本书是为那些已具备 Windows 基本的使用基础、C 语言基础知识的读者编写的。重点放在设计一些实例程序并对其进行分析上，以说明 Windows 和 Borland C++程序设计的基本思想和方法。书中所有实例都只是提供了应用程序设计的框架，以便于读者在此基础上设计并开发出自己感兴趣的更完善的应用程序。

本书适用于 Borland C++系列软件的应用与开发者、Windows 使用者。特别是对大中专学生的计算机应用程序设计水平的提高，具有较强的指导作用。本书亦可作为 Windows 应用程序设计与开发的培训班教材。

（陕）新登字 007 号

Borland C++ For Windows 应用程序设计及实例

李舜酩　石宵宏　王妍臣　编著

责任编辑　贺峰涛

组稿编辑　林　全

西安交通大学出版社出版发行

（西安市咸宁西路 28 号　邮政编码 710049）

西安电子科技大学印刷厂印装

各地新华书店经销

*

开本：787×1092　1/16 印张：19.5　字数：466 千字

1996 年 4 月第 1 版　1996 年 4 月第 1 次印刷

印数：1——5000

ISBN7-5605-0852-9/TP・137　定价：20.00 元

前　言

Windows 风暴席卷全球，把人们拥有的计算机世界涤荡出一个光辉灿烂的崭新面貌。人们不再为自己已拥有的计算机知识感到自豪，而是面临着一种新的思维方式和新的处理技术的挑战。迈进 Windows 的世界，可以说是海阔天空；如果畏惧不前，要在计算机应用与开发领域做出点成绩，将更加步履艰难。

Windows 有许多优点，这已为广大计算机用户所公认。怎样有效地利用其各方面的优点，开发出适合自己的应用程序，是许多用户所希望掌握的。在常用的几种开发语言中，Borland C++具有独特的优越性，它不但可以开发出完美的适合用户自己需要的应用程序，而且能够为大型的应用程序开发提供优秀的环境和资源。可以说 Borland C++是极其优秀的 Windows 应用程序开发工具。也正因如此，它在短短几年的时间里迅速发展，版本不断升级，已发展成为一个完善的系统。

本书在前两章中对 Borland C++的一些基本技术作了简单的介绍。至于 Borland C++的详细语法、规则、库函数以及类的解释，则不是本书的任务。实际上，阅读本书并不需要首先去掌握这些内容，相反的是当你读完本书并希望在 Borland C++的 Windows 应用程序开发方面做进一步的研究与探索时，你就会自然产生进一步掌握更详细内容的要求。本书只要求读者具备 C 语言基础和一定的程序设计实践，不需要更多的知识。有其它语言的使用经验，通过努力也可以掌握本书的内容。

本书还用了三章的篇幅来介绍 Windows 的程序设计风格、Windows 程序的基本概念和 Windows 应用程序设计的基本方法。由于 Windows 本身是用 C 语言编写成的，因此，在这部分内容的若干地方介绍了 Windows 的 C 程序特点、函数、方法等。无疑这十分有利于对 Borland C++的理解。当然，我们在若干地方强调了 Borland C++的新特点，甚至 Borland C++高版本的新特点。所给出的示例程序也是采用 Borland C++基于目标 Windows 库(OWL)编写的。本书要求读者对 Windows 的基本操作有一定的经验。所涉及的 Windows 术语、概念等，本书全部假设读者已掌握，不再另作解释。

本书用了大部分篇幅分章节介绍了 Windows 各组成部分的功能和如何用 Borland C++对这些功能进行开发。在本书的最后一章，还讲述了对所开发的应用程序进行调试分析的方法。

有关用 Borland C++开发 Windows 应用程序的著作有多部，内容、风格各有千秋，本书并没有沿袭和效仿它们的编写指导思想和方法，而是具有自己的特色：

1. 语言通俗易懂。

2. Windows 系统是以中、西文 3.1 版本为基础的，没有追逐版本的升级。实际上，应用程序的开发受版本升级的影响并不大。问题的关键是所开发的应用程序功能是否齐全强大、界面是否友好以及是否美观等。

3. 本书采用了目前流行的 Borland C++ 3.1 和 4.0 版本进行 Windows 应用程序开发，并且也适用于 Borland C++ 4.5 版本。所开发的应用程序在 386 系列以上微机上调试通过。读者可以直接使用书中给出的源程序进行学习，也可以选用我们为书中所有应用程序源程序制做的软盘。

4. 本书各章节中的每个应用程序都相对独立。每个应用程序都可单独使用，而不是贯穿全书提供一个大的、完善的应用程序。这样做是为了便于读者“化整为零”来学习与掌握。许多应用程序之间没有任何联系。

5. 各章节中所给出的应用程序都不是完善的。书中给出的是一些应用程序框架，只开发了其中一部分功能。并不是不可以开发出功能齐全的应用程序，而是从读者的学习效果角度出发，留下许多功能让读者学完有关章节后自己去实践，去开发。这是我们在多年教学中总结出的引导法在本书中的具体体现。这样会调动读者的学习兴趣，以便于读者在原有程序的基础上设计并开发出自己感兴趣的更完善的应用程序。

本书适合于 Borland C++系列软件的用户、Windows 应用程序的开发者以及 Windows 系统的用户。特别是对大学生计算机应用程序设计水平的提高，具有较强的指导作用。本书亦可作为 Windows 应用程序设计与开发的培训班教材。

本书由李舜酩、石晓虹、王好臣编著。石东洋、梅立泉、惠延波同志参加了部分章节的编写工作。许庆余教授在各方面给予了大力支持并以其严谨的治学风范给作者以深刻的影响。李香莲、李晖、李立和董小鹏等同志在书稿的整理、组织方面做了许多工作，在此一并表示感谢。在此，还要感谢刘路放同志的大力支持与帮助。

本书由李舜酩统稿。

由于时间和水平所限，书中疏漏与错误在所难免，诚请读者提出宝贵的意见。

作　者

1996 年 1 月

目　录

第1章

Borland C++简介

1.1 C++的起源与发展

C语言有诸多优点：它在绝大多数系统程序设计中能够替代汇编语言；它具有高效、灵活、可移植性好等特点；模块化、容易独立编写与调试；等等。但是，C语言也有其不足：当程序达到一定规模时（一般认为超过25 000行），它就变得相当复杂，要全面掌握就很困难了。这样不仅不利于程序开发，而且维护起来也更加费力。

1980年美国贝尔实验室的Bjarne Stroustrup将一种所谓Simula67面向对象的语言中的“类”这个概念引入到C语言中，对C语言进行了扩展，并称之为“带类的C语言”。后来在他人的建议下于1989年将该语言正式定名为“C++”。在使用C++语言的过程中，Stroustrup与他的同事们对其进行了两次修订，先后引入了运算符重载、引用和虚拟函数等特性，使它更加精炼。到1989年底推出的是AT&T C++ 2.0版本。

C++不但保持了C语言的诸多优点，而且增加了对“面向对象程序设计”的支持，即“使程序结构清晰、易于扩展、易于维护而不失其效率”。由于C++的这些优势，使它不仅不局限于其原先设计这种语言时的想法——帮助其它系统管理大型程序，而且可用于实际的程序设计工作（这已为目前的许多设计工作所证明）。C++常用于设计编辑器、数据库、个人文件系统及通讯程序等。而且，由于C++共享C的效率，所以用C++可以构成很多高性能的系统软件。因此，C++语言随着其版本的不断更新（目前已有7.0版本），其使用面也越来越广泛。

1.2 C++与C的关系

除了上节所述C语言的优点外，它还有许多其它方面的优点，从而导致了其广泛的使用。这些优点表现在：

1）C语言是一种高级语言，很适合于表达各类算法；

2）用C语言编写的程序可以十分精巧、易读；

3）C语言拥有优秀的编译器；

4）C语言在许多领域都拥有优秀的软件工具；

5）C编译器可以生成短小、高效的可执行代码；

6）C语言允许用户存取和控制硬件特性；

7）使用C语言编程可以考虑到低级的细节内容；等等。

但从某种角度上讲，原始的C语言并不是一种真正的高级语言。因为C语言编程常常要

考虑到低级的地址方面的内容，这就会导致编写出来的C程序与算法本身在形式上有着较大的差异。而且当C程序使用不当时，所写出的程序也难以让人理解。而C++则完全支持高级语言特性。下面来看一下C++与C之间的异同。

1.2.1 C++与C的相似点

可以说C++的绝大部分是ANSI标准C的高级集合，所以，C程序都是C++程序。它们之间存在若干相似点：

- 所有编程控制逻辑相同。例如：WHILE、FOR等循环，IF条件以及SWITCH语句并未改变。
- 运算符，包括位操作运算符均未改变。
- 函数的概念及它的调用过程基本未变。稍有不同的是C++中的大的函数通常定义为某对象的一部分而不是任何人都可使用。
- 现有C运行时间库不作修改还可继续工作，使一些熟悉的函数如printf()、scanf()还存在。
- 所有C的类型修饰符，如struct（结构）和union（联合）还保留着而且含义也未改变。

1.2.2 C++与C之间的差别

C++对C在系统上进行了以下较大的扩展：

- 安全类型连接域分辩运算符：域分辩运算符::，可以用来引用当前程序模块的作用域以外的变量。
- new和delete运算符：new和delete运算符可以为用户定义的类型分配内存，而不要求程序员跟踪对象的大小。
- I/O流库：I/O库支持面向对象的输入和输出。
- 函数原型：函数原型允许编译器和连接器在进行更好的编译时的类型检查。
- 类型转换函数编译器：用来对所使用的对象进行类型转换。
- Const和Volatile存储类：数据可以声明为Volatile或Const。

同时为了更方便地编程，C++还对C进行了以下内容的扩展：

- 函数重载：函数重载可让不同函数用同样的函数名字对不同的数据类型执行相同的逻辑的操作。
- 缺省参数：可以为函数参数指定缺省值。
- 块内变量说明：可以在使用变量的位置定义它们，而不是在函数的开头。说明的变量的作用域便是它们所在的块。
- 引用变量：通过引用使传递参数更方便，访问引用参数也很方便。
- 重载运算符：标准的运算符（+，－，<<等等）可以和用户定义类型一起工作。
- 内联函数：函数可以内嵌地展开，从而节省调用开销。和宏不一样，C++要对内联函数执行类型检查，而且参数也象普遍函数调用一样的传送。
- 单行注释：一个//指明该行剩余部分是注释。

前面已讲过C++是面向对象的程序设计语言，这些对象一般包括：类、访问控制、友元、虚函数、构造函数、析构函数和继承等。这些将在下面几节中给出具体的解释。

需要说明的是，本书假定读者已经学习并掌握了C语言的基本编程方法，所以在讲解许多概念、用法上直接提出而不加解释。如果没有C语言基础，建议先学习一些C语言的基本知识并调试几个程序后再回头来学习本书。

1.3 C++的封装

在C++中，封装是通过类、访问控制和友元来实现的。所谓C++的封装(Encapsulation)，可描述为数据结构和操纵这些数据的函数（动作或方法）的结合。

1.3.1 类

C++是面向对象的程序设计语言，对象包含属性和行为。计算机环境就是一个对象集合体，对象之间彼此通过消息相互作用。面向对象的程序设计提高了程序的模块化和可维护性。面向对象的程序设计的关键特征是将数据成员和函数成员封装到一个对象中。在C++中，这种实体叫做“类”(class)。对使用C++的类的用户来讲，类与语言本身内部的数据类型是一样的。

1. 类的实现

定义sAstring和sAnotherString为Str类，记为：

```
Str sAstring, sAnotherString;
```

则此Str类可实现如下功能：

```
Str sAstring [100], sAstringMatrix [10] [10]; //创建对象数组。
if (sAstring<=sAnotherString) //比较。
{sAstring=sAnotherString+" demo";} //连接。
Str * spAPointer; //定义指向对象的指针。
spAPointer=new Str; //从内存堆中给对象分配空间。
delete spAPointer; //将内存空间归还给内存堆。
```

用户可以给类增加功能（对标准数据类型不适用）。这些功能通过成员函数来实现，成员函数是对象的一部分。例如：

```
sAString · Find (" Test"); //找内嵌串位置。
sAnotherString=sAString · Slice (10, 15); //访问一个子串。
```

注意，成员函数Find()和Slice()并不将串当作一个参数。函数调用sAString · Find()自动对sAString操作。

2. 类的说明

C++中的类说明的语法和C中的结构定义几乎是相同的，唯有一点不同是类说明中包含了成员函数。下面是一个类的例子：

```
class Circle {
    int x;
    int y;
    int radius;

    int DrawCircle (int a, int b, int rad);
```

```
    int DeleteCircle (int a, int b, int rad);
};
```

这个类的数据成员是 x、y 和 radius；类的成员函数是 DrawCircle 和 DeleteCircle，成员函数的实现使用如下形式：

```
int Circle :: DrawCircle (int a, int b, int rad)
{
……函数体
}
```

成员函数的语法和一般的函数语法基本相同。只是要在成员函数前面加上类名来说明成员函数属于哪个类。

1.3.2 访问控制

封装的一个重要功能是能控制对象内部的访问。C++提供了三种类型的数据隐蔽以控制访问：

• public（公有）数据，可由任何其他代码读写。公有成员函数可以被任何成员函数调用。

• protected（保护）数据，可由类的成员函数读写，也可以被类似派生类的成员函数读写，对其它成员函数来说则不可见。保护成员函数可以被类的成员函数、派生类的成员函数调用。其它成员函数不能调用保护成员函数。

• private（私有）数据，只能被类的成员函数读写。其它成员函数，包括该类的派生类的成员函数都不能访问私有数据。私有成员函数只能被该类的成员函数调用。

将成员数据或成员函数定义为公有、保护、私有，可按如下格式：

```
class SampleClass
{
private:
  int nprivateData;
  void Privatefunction (void);
protected:
  int nProtectedData;
  void ProtectedFunction (void);
public:
  int nPublicData;
  void PublicFunction (void);
}
```

1.3.3 友元

有些情况下要“越过”访问控制而让某些类或函数来访问类的数据成员和函数成员。友元访问权限就是为此而设置的，用关键字 friend 来定义友元，它可以应用于一个函数名，一个类名或者一个运算符。

下面这段程序把成员函数 PointArray 定义为类 Point 的友元：

```
class PointArray;        //Forward declaration
class Point
{
    private:
      int x, y;

    public:
      friend PointArray;
}
```

友元函数和友元类相似，友元函数可以访问类的私有数据和成员函数。友元函数的一个最经常的用法是重载运算符。重载的运算符可以作为友元函数或是作为类成员函数。假设要为类 Str 重载运算符＋，可写出如下程序段：

```
class Str
{
    Public:
    friend Str operator + (Str &. sString);
}
```

1.4 C++中的抽象

C++包含了许多内部的运算符，可进行基本的算术操作甚至进行位操作。这种操作不是直观地而是抽象地进行。这种抽象要求程序设计者定义一个新的类，并将逻辑上合理的运算符定义为类的行为。这个定义过程是通过对类进行运算符重载来实现的（使某个运算符对某一类具有特定含义称作运算符重载，重载后原运算符的含义不变）。

1.4.1 运算符重载

C++只允许对所定义的一个新的类进行运算符重载，已定义的类不能进行运算符重载。运算符应是类成员，或者至少有一个参数为类。重载运算符使用的语法形式为：

```
<return type>operator<op> (Arg....)     //重载非成员函数
<return type><class>::operator<op> (Arg....)     //重载成员函数
```

在这两种形式中，<return type>是运算符函数的返回类型，<class>是类名，<op>是被重载的运算符，(Arg....) 是运算符的参数。

例如，要定义一个叫 Matrix 的类重载＋＋运算符（单目运算符），若不使用成员函数，可写成：

```
Matrix operator ++ (Matrix &. mFirstMatrix)
```

若重载运算符要定义为类成员函数，则语法上稍有区别。由于所有成员函数都有一个隐含的参数，叫做 this 指针，于是上例中的参数可省掉，格式如下：

```
Matrix Matrix::operator ++ ()
```

此式中，由于第二个 Matrix 是类名，所以双目运算符＋＋就被定义为 Matrix 类的成员函数。