计算机技术入门提高精通系列丛书

GOTOP

Visual C++ 4.x

实用教程

杨宗璟 林猷琮 编著
王 晟 王 晖 改编
方 裕 审校

人民邮电出版社

计算机技术入门提高精通系列丛书

Visual C++ 4.x实用教程

杨宗璟 林猷琮 编著
王 晟 王 晖 改编
方 裕 审校

人民邮电出版社

计算机技术入门提高精通系列丛书

Visual C++ 4.X 实用教程

◆ 编　　著　杨宗璟　林猷琮
改　　编　王　晟　王　晖
审　　校　方　裕
责任编辑　李　际

◆ 人民邮电出版社出版发行　北京崇文区夕照寺街14号
北京鸿佳印刷厂印刷
新华书店总店北京发行所经销

◆ 开本:787×1092　1/16
印张:13.25
字数:323千字　　1997年9月第1版
印数:6 001—12 000册　　1998年3月北京第2次印刷
著作权合同登记　图字:01—97—0581号
ISBN7-115-06669-8/TP·511

定价:18.00元

内容提要

Visual C++是Microsoft公司推出的新一代面向对象的程序开发工具。Visual C++以其功能齐全的集成开发环境，在各种版本的C++语言中占有十分重要的地位，是目前国内外最为流行的C++语言之一。Visual C++ 4.x（包括Visual C++ 4.0 ~ Visual C++ 4.2）是目前国内外广泛使用的C++系统开发软件包。

本书共分十章，从入门着手，详细介绍了Visual C++ 4.x的使用方法与技巧，并在附录中提供了在Internet上获取C++资源的途径及其它必要信息。

本书论述简明扼要、图文并茂、通俗易懂，适合于计算机软件开发人员及大专院校计算机专业师生阅读参考，也可作为Visual C++ 4.x的培训教材或参考资料。

版权声明

出版说明

在计算机技术飞速发展的今天，为了进一步向全社会普及计算机知识，提高计算机应用人员的技术水平，使计算机在各个领域发挥更大作用，也为了促进海峡两岸计算机技术图书的交流，台湾碁峰资讯股份有限公司授权我社陆续组织出版该公司的部分计算机技术书籍。这些书基本覆盖了当前最常用的各类计算机软、硬件技术，并紧随世界上计算机技术的飞速发展，不断有所更新。在写作特点上，这些书内容深入浅出、实用性强，在台湾地区很受读者欢迎。

在组织出版过程中，我们请有关专家在尊重原著的前提下进行了改编，并对有关图文进行了核对和精心制作。

由于海峡两岸计算机技术名词和术语差异较大，改编者依照有关规定和我们的习惯用法进行了统一整理。

对原书文字叙述中由于海峡两岸不同的语言习惯而造成的差异，我们的处理原则是只要不会造成读者理解上的歧义，一般没做改动，以尊重原著写作风格。另外改编时对原书的一些差错及疏漏之处做了订正。

由于本书改编和出版时间紧张，如有差错和疏漏，敬请读者指正。

人民邮电出版社

1997年6月

目　录

第一章 C++语言概述

如果读者对C++语言已有一定的基础，可以略过本章，直接学习后面的内容；如果对C++语言只是略懂皮毛，而不了解其精髓，请花费少许时间将本章浏览一番。本章讨论的内容以C++的“面向对象”概念为重点，要了解其它一般性的概念，建议读者参考有关C++的书籍。

1.1 对象与类(Objects and Classes)

对象与类是C++语言中很重要的概念，读者必须深入地了解它们。简单地说，一个对象类除必须拥有足以代表其特征的特性外，还必须提供操作此对象的程序。

C++的类定义语法如下：

```
class InfoID
{
 private:
     int infoid_x,infoid_y;
public:
     void setinfo(int I);
     int getinfo_x( );
     int getinfo_y( );
}
```

在C++的类中提供了private、protected和public三个关键字来设定

各成员的保护等级（可使用的范围），如表1-1所示。

表1-1 类关键字的比较

使用范围	定义
只能在类的内部使用	private和protected所定义的数据
可以提供给外界使用	public所定义的数据

注：private和protected在类的继承时会有所不同。

1.1.1 构造函数(Constructor)

在建立对象时，经常需要能自动设定某些初始值，类内的成员函数可以使用构造函数(Constructor)的特殊成员函数。构造函数会在对象建立时自动执行。

```
class counter
{
    private:
        int id_count;
    public:
        Construct( )      //构造函数
        {
            id_count=0;
        }
        void incre_construct( )
        {
            id_count++;
        }
        int get_construct( )
        {
            return id_count;
        }
}
```

1.1.2 析构函数(Destructor)

在类中不但有构造函数，而且还有析构函数。析构函数是另一种特殊的C++成员函数，它只是在类名称前面加上一个“~”符号。每一个类都有一个析构函数，没有任何参数，也不返回任何值。

```
class destruct
{
    private:
```

```
    int des_count;
public:
    const( )        //构造函数
    {
        data=0;
    }
    ~const( )       //析构函数
    {
    }
}
```

1.2 继承(Inheritance)

继承是面向对象语言中最有用的功能。它是从已有的“基类”产生出“派生类”。派生类继承了基类所有的功能，并加上了一些自定义的功能，如图1-1所示。

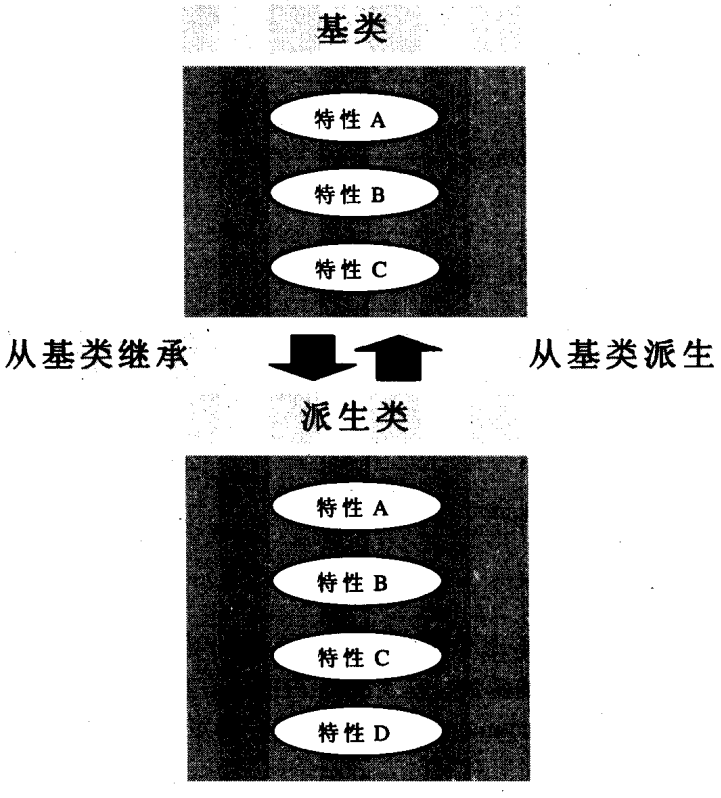


图1-1 继承关系

继承允许原始程序代码的“重用”(Reusability)。一个基类被编写并且调试后，不需要经过任何修改就可以被继承使用。重新使用已有程序码不但省时省力，而且提高了程序的编写效率。

```
class demo          //基类
{
 protected:         //不可定义成private，因其要继承，故是protected
  int id count;
 public:
  demo( )           //构造函数，无参数
  {
   id count=0
  }
  demo(int num)             //构造函数，有参数
  {
   id count=num;
  }
  int get_counter( )
  {
   return id count;
  }
  demo operator++( )        //operator是一个重载运算符
  {
   id count++;
   return demo(id count);
  }
}
class demoDn : public demo  //派生类
{
 public:
  demo operator-- ( )           //operator是一个重载运算符
  {
   id count-- ;
   return demo(id_count);
  }
}
```

上面例子中的demoDn类就是派生类，它是一个新的类，其内部加入了一个新的函数operator--( )，这是一个递减计算的函数。基类demo内的所有功能都派生到demoDn的类中，因此demoDn类中的构造函数、get_counter( )及operator++( )的定义都不需要再重新进行，完全继承demo基类中的。

1.3 运算符重载(Overload)

运算符重载是面向对象程序设计中一项很令人振奋的功能，它可以使复杂的程序变得很清晰。刚开始编写类程序的设计者，可能不习惯使用很多的重载运算符，不过当他逐步了解重载的设计后，就会知道它的好处。

```
0_load 0_load :: operator + (0_load d1)
{
    return 0_load(x+d1.num_dx,y+d1.num_dy);
}
```

以下面的实例来说明重载：

```
0_load 0_load1(10,20)
0_load 0_load2=0_load1*2.5;          //值是(25,50)
0_load2 +=10;                        //值是(35,60)
```

1.4 虚函数(Virtual Functions)

虚函数是一个不完全存在但对程序的某些部分起作用的函数。当想要在一些属于不同类的对象中，用同一个函数对某些对象进行同一个特定的操作时，使用者便可以用虚函数来解决此类问题。

```
virtual void show( )          //虚函数的定义
{
 cout<<"virtual=0";
}
```

1.5 友元函数(Friend Functions)

当两个不同类的对象关系十分密切时，可以利用C++的friend保留字来使其拥有同等的地位。

```
class num1
{
```

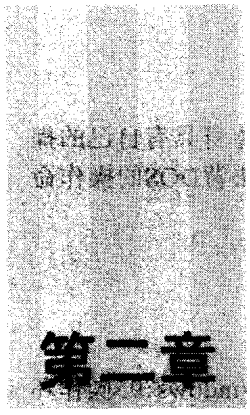
```
  int data1;
public :
  fun1( ){data1=10;}
  friend int fun_friend(fun1,fun2);
}
    class num2
{
  int data2;
public :
  fun2( ){data2=20;}
  friend int fun_friend(fun1,fun2);
}
int fun_friend(num1,num2)
{
 return(num1.data1+num2.data2);
}
```

1.6 this指针

任何一个对象的成员函数都可以存取一个特殊的指针this，它指向对象自己。因此任何成员函数都可以找出自己的地址。

```
this ->addr=11;
cout << this ->addr;          //输出值为11
```

本章所讨论的只是C++语言中重要的面向对象概念，如读者尚不甚了解C++语言的概念，请准备一本有关C++语言的书籍以作为参考。



第二章 Visual C++ 4.X概述

2.1 Visual C++ 4.X的环境

近年来，软件的发展已经从Win16操作环境转移到Win32操作环境。Win32代表新的32位Windows API，由Windows 95和Windows NT所支持，而以往的Win16代表16位Windows API，以Windows 3.1和Windows for Workgroup为环境。

Visual C++ 4.X是一个在Win32环境下的程序开发工具，所以其执行环境为Windows 95或Windows NT 3.51。本书是以Windows95为操作环境的，书中插图皆为在Windows 95中的操作界面。

2.2 Windows 95的特色

读者对Windows 95恐怕不会很陌生，但对其为什么会如此受到青睐，必定感到好奇，它到底有何特性呢？

2.2.1 崭新的图形操作系统

Windows 95操作系统不同于以往的DOS加上Windows 3.1操作系统，其本身具有自己的操作系统，只要一开机即可进入Windows 95的操作环境。用户完全不需要再去背DOS的操作命令，也看不到DOS下的提示符“C:\”。

2.2.2 统一的操作界面

对任一个使用者而言，操作上的统一可以减少学习过程中的障碍。在Windows 95操作环境中，用户只需了解众多操作功能之一，再加上对鼠标的熟练运用，即可轻而易举地操作Windows 95。由于Windows 95的功能众多，故操作方式及界面的一致，可减少学习上的困难。

2.2.3 32位的操作模式

32位处理模式比16位处理模式的执行速度快得多。Windows 95已具备32位的能力，如果再使用32位的软件，则会有更好的执行效率。Visual C++ 4.X即是32位的套装软件。

2.2.4 即插即用功能

Windows 95提供“即插即用”的功能，这项功能可以自动进行检测并完成设定，免除以往人工设定系统文件或调整硬件上IRQ冲突的问题。

它主要的特性是：协助安装和检测硬件上的设定与操作系统是否吻合，而且可以由操作系统对硬件设备进行管理与资源分配，减少安装上的麻烦。

2.3 Windows应用程序的编写

以往用C或C++语言编写Windows应用程序时，可以说是一件非常艰巨的事，小者需要几百行，大者需要几千行描述，甚至要加上数十个、数百个新库函数及DOS、BIOS的调用才能完成一幅简单的Windows画面。因此设计Windows环境下的应用程序，一直是程序设计者最头痛的事。

在此先回忆以往C / C++语言程序的编译(Compiler)及链接(Linking)流程，如图2-1所示。