普通高等教育“十一五”国家级规划教材

Visual C++ 面向对象与可视化程序设计（第4版）

编著　黄维通　解　辉

高等教育出版社

十一五 普通高等 规划教材

Visual C++面向对象与可视化程序设计

Visual C++ Mianxiang Duixiang yu Keshihua Chengxu Sheji

（第4版）

编著 黄维通 解 辉

高等教育出版社·北京

内容提要

本书是从面向对象的基本概念出发，讲述可视化程序设计的思想与方法。对每一部分的知识点、概念、难点，都力求以较精炼的语言进行讲解，同时，对每一个知识点都配以必要的实例，实例中配以较为详细的步骤说明、代码说明及语法说明，力求通过实例让读者较好地掌握“面向对象与可视化程序设计”的思路、开发技巧与体系。

为了更好地让读者掌握本知识点及其应用实例，本教材配备了二维码，读者只要扫描二维码就可以免费观看相关内容的 MOOC 视频进行学习，还可以登录清华大学的“学堂在线”免费注册该课程进行全程学习，并能得到在线答疑和同学们的在线研讨。

本教材分为如下四个部分：

- 第一部分讲述 VC++的基础知识，包括 C++的基础知识，主要是考虑到读者在有了 C 语言的基础后，能直接学习本教材。
- 第二部分介绍 Windows 编程构架及部分专题应用，包括 Windows 绘图、文本输入/输出、键盘与鼠标的应用以及资源的应用等基础知识。
- 第三部分介绍 MFC 构架（包括类库的基本知识）、各种常用类在编程中的应用、常用控件的应用、利用 Visual C++的资源编辑器编写资源文件及其应用、文档操作等知识点。
- 第四部分介绍了高级编程应用，如多媒体、数据库的基本概念与方法及相关应用案例。

图书在版编目（CIP）数据

Visual C++面向对象与可视化程序设计/黄维通，解辉编著. --4 版. --北京：高等教育出版社，2016.6

ISBN 978-7-04-045554-0

Ⅰ. ①V… Ⅱ. ①黄… ②解… Ⅲ. ①C 语言-程序设计-高等学校-教材 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2016）第 105259 号

策划编辑 刘 茜	责任编辑 刘 茜	封面设计 张 志	版式设计 张 杰
插图绘制 杜晓丹	责任校对 杨凤玲	责任印制 赵义民	

出版发行	高等教育出版社	网 址	http://www.hep.edu.cn
社 址	北京市西城区德外大街 4 号		http://www.hep.com.cn
邮政编码	100120	网上订购	http://www.hepmall.com.cn
印 刷	北京市鑫霸印务有限公司		http://www.hepmall.com
开 本	850mm×1168mm 1/16		http://www.hepmall.cn
印 张	25.5	版 次	2001 年 6 月第 1 版
字 数	580 千字		2016 年 6 月第 4 版
购书热线	010-58581118	印 次	2016 年 6 月第 1 次印刷
咨询电话	400-810-0598	定 价	45.00 元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换

版权所有 侵权必究

物 料 号 45554-00

前　言

随着程序设计与开发技术的不断发展，在大学的计算机基础课程中，讲授面向对象技术及开发过程的可视化，已经成为计算机基础课程教学改革的方向之一。掌握面向对象的程序设计，已经成为大学生信息素养和能力结构的重要组成部分，也是社会对人才的计算机应用与开发水平的要求。

学习本教材，要求先修C语言（或其他任何一种编程语言），虽然C语言已经成为高校理工科学生的必修或选修课程，是很重要的程序设计的入门基础课程，但它是面向过程的编程语言。应用面向对象的编程技术已经成为当今软件开发的重要手段之一，因此，掌握“面向对象与可视化程序设计”的技术与方法，已经成为大学生掌握信息技术并用来解决本学科计算问题或相关领域问题的基本方法之一。

本教材的第一版自2001年出版以来，国内很多学校开设的相关课程采用了此教材，第1版被评为教育部“全国普通高等学校优秀教材”二等奖，第2版被评为“北京高等教育精品教材”，第3版是普通高等教育“十一五”国家级规划教材。本教材是第4版，作为第3版的修订版。

本教材充分考虑VC++面向对象程序设计的技术发展，在原有第3版的基础上，结合前面几版的教学体会，从面向应用的教学改革定位出发，对部分例题进行修订，提高了例题的实用性和趣味性。同时，所有例题全部在Visual Studio 2012环境下调试通过。

本教材主要分为四个部分，第一部分讲述VC++的基础知识，包括C++的基础知识，主要是考虑到读者在有了C语言的基础后，能直接学习本教材；第二部分介绍Windows编程构架及部分专题应用，包括Windows绘图、文本输入/输出、键盘与鼠标的应用以及资源的应用等基础知识；第三部分介绍MFC构架（包括类库的基本知识）、各种常用类在编程中的应用、常用控件的应用、利用Visual C++的资源编辑器编写资源文件及其应用、文档操作等知识点；第四部分介绍了高级编程应用，如多媒体、数据库的基本概念与方法及相关应用案例。本教材内容丰富翔实，建议授课学时为48学时，当然，不同群体根据培养定位的不同，可以做相应的取舍并调整学时。

本教材特点是从面向对象的基本概念出发，讲述可视化程序设计的思想与方法。对每一部分的知识点、概念、难点，都力求以较精炼的语言进行讲解，同时，对每一个知识点都配以必要的实例，实例中配以较为详细的步骤说明、代码说明及语法说明，力求通过实例让读者较好地掌握“面向对象与可视化程序设计”的思路、开发技巧与体系。

为了更好地让读者掌握本教材涉及的知识点及其应用实例，书中配备了二维码，读者只要扫描二维码就可以免费观看相关内容的MOOC视频进行学习，还可以登录清华大学的“学堂在线”免费注册该课程进行全程的学习，并能得到在线答疑，与同学们在线研讨。

本教材中部分专题内容如第8章中介绍的“对话框通用控件”中的应用程序、第9章的文档应用、第10章的资源应用程序、第12章的数据库应用程序等，都是分别以一个综合应用程序的方式，把相关知识点内容分解到各节的内容中去，通过各节内容的介绍，不断增强本章实例中的功能，使得读者在循序渐进的学习过程中掌握一个完整应用程序的开发方法及相关知识点。第11章的多媒体编程介绍了常用的音频、视频的应用，以及简单的图形处理软件功能的应用。这些都是非常实用的知识。

本书面向高等院校本科生、研究生及从事计算机软件开发的专业人员，既适用于作为高等学历教育的教材，也适合非学历教育的各类培训作为培训教材，同时还适合计算机爱好者自学。

本书由黄维通、解辉编写，其中解辉编写了第1章和第7章的内容。在本书的编写过程中，作者还查阅了部分文献，在本书的“参考文献”部分列出了这些文献的作者，在此对上述作者表示感谢。

由于作者水平有限，缺点和错误在所难免，恳请读者批评指正。

谢谢喜欢阅读本书的读者！

作者联系信箱：huangwt@ tsinghua. edu. cn。

黄维通

2016年元旦于清华园

试读结束：需要全本请在线购买： www.ertongbook.com

郑重声明

高等教育出版社依法对本书享有专有出版权。任何未经许可的复制、销售行为均违反《中华人民共和国著作权法》，其行为人将承担相应的民事责任和行政责任；构成犯罪的，将被依法追究刑事责任。为了维护市场秩序，保护读者的合法权益，避免读者误用盗版书造成不良后果，我社将配合行政执法部门和司法机关对违法犯罪的单位和个人进行严厉打击。社会各界人士如发现上述侵权行为，希望及时举报，本社将奖励举报有功人员。

反盗版举报电话　(010)58581897　58582371　58581879
反盗版举报传真　(010)82086060
反盗版举报邮箱　dd@ hep.com.cn
通信地址　北京市西城区德外大街 4 号　高等教育出版社法务部
邮政编码　100120

目　录

第 1 章　C++基础知识

第 2 章　Windows 应用程序基础

第 3 章　Windows 的图形设备接口及 Windows 绘图

第 4 章　字体及其应用

第 5 章 Windows 应用程序对键盘与鼠标的响应

第 6 章 资源在 Windows 编程中的应用

第 7 章 MFC 基础知识

第 8 章 控件在可视化编程中的应用

第 9 章 单文档与多文档的应用

第 10 章 在 MFC 中创建应用程序的资源

第 11 章 多媒体应用程序的设计

第 12 章 数据库应用程序的开发

第 1 章

C++基础知识

本章将介绍 C++编程的基础知识。相信大家都学过 C 语言，因此，有了 C 语言的基础，再介绍 C++的基础，大家就很容易熟悉并掌握面向对象的程序设计方法。

在简要介绍 C++基本概念的时候，那些在 C 语言中应该已经学过的内容和基本概念，在这里就不再赘述了。如果已经能够熟练地使用 C++语言进行编程，请直接进入第 2 章的学习。

1.1　C++的发展历程

C++是既适合于作为系统描述语言，也适合于编写应用软件的，既面向对象又面向过程的混合型程序设计语言，它是在 C 语言的基础之上发展而来的。

在 C 语言推出之前，操作系统等系统软件主要是用汇编语言编写的（如著名的 UNIX 操作系统）。由于汇编语言依赖计算机硬件，因此程序的可移植性和可读性比较差。为了提高程序的可读性和可移植性，最好采用高级语言来编写这些系统软件。然而，一般的高级语言难以实现汇编语言的某些功能（如汇编语言可以直接对硬件进行操作，对内存地址进行操作和位操作等）。人们设想有一种能集一般高级语言和低级语言特性于一身的语言。于是，C 语言便应运而生了。

最初的 C 语言只是为描述和实现 UNIX 操作系统而提供的一种程序设计语言。1973 年，贝尔实验室的 K.Thompson 和 D.M.Ritchie 两人合作把 UNIX 90%以上的代码用 C 语言改写（即 UNIX 第五版）。后来 C 语言又进行了多次改进，1978 年以后，C 语言已先后移植到大、中、小及微型机上，使得 C 语言成为风靡全球的计算机程序设计语言。

到了 20 世纪 80 年代，美国 AT&T 贝尔实验室的 Bjarne Stroustrup 在 C 语言的基础上推出了 C++程序设计语言。由于 C++提出了把数据和在数据之上的操作封装在一起的类、对象和方法的机制，并通过派生、继承、重载和多态性等特征实现了软件复用。这使得软件，尤其是大型复杂软件的构造和维护变得更加有效和容易，并使软件开发能更自然地反映事物的本质，从而大大提高了软件的开发效率和质量。

C++越来越受到重视并得到广泛的应用，许多软件公司都为 C++设计编译系统。如 AT&T，Apple，Sun，Borland 和 Microsoft 等，其中较为流行的是 Microsoft 的 Visual C++。与此同时许多大学和公司也在为 C++编写各种不同的类库，Microsoft 的 MFC（Microsoft Foundation Class Library）就是比较优秀的代表之一，在国内外得到较为广泛的应用。

1.2　一个简单的 C++程序

下面通过一个非常简单的用 C++编写的例子，让读者了解一下 C++的基本输出操作。其功能是在屏幕上显示“Welcome !”，其程序代码如下：

试读结束：需要全本请在线购买：　www.ertongbook.com

```
#include "stdafx.h"
#include "iostream"
using namespace std;
void main()                              //main 函数，程序入口
{                                        //程序体开始
    char str_greet[]="Welcome!";         //定义一个数组并初始化
    cout<<str_greet<<endl;               //在屏幕上输出字符串内容
}                                        //程序体结束
```

熟悉 C 语言的读者不难看出，用 C++编写的程序和用 C 编写的程序在程序结构上是基本相同的，都是以 main 函数作为程序的入口；两者都是以一对 {} 把函数中的语句括起来；两者都是以分号作为语句的结束标志。但是，两者也有一些不同之处，C++中是以 iostream 文件作为标准输入/输出头文件，C 是以 stdio.h 作为标准输入/输出头文件；C++中采用符号“<<”作为标准输出，而不是通过 printf 函数来实现。

由于 C++标准函数库的所有元素都被声明在一个“命名空间”中，上述代码“using namespace std;”之中的 std 就是“命名空间”。因此为了能够访问它的功能，用这条语句来表达我们将使用标准命名空间中定义的元素。

cout 是 C++中的标准输出流（通常为控制台，即屏幕），这句话把一串字符串（本例中为“Welcome!”）插入输出流（控制台输出）中。cout 声明在头文件 iostream 中，所以要想使用 cout 必须将该头文件包括在程序开始处。

通过上面的例子可以看出，C++语言和 C 语言之间既有紧密的联系，又有各自的特点。下面的内容将介绍 C++程序设计中的一些基础知识，这部分内容，C++和 C 有很多是一致的。由于本书是面向已经熟悉 C 语言并初步掌握 C++语言的读者，因此，对 C++的内容只是作一个简单的总结性概述，如果读者对 C 及 C++语言很熟悉，可以跳过这部分内容的学习。

1.3 C++中的变量和数据类型

在 C++中，局部变量的作用范围被定义在声明它的程序块内（一个程序块是被一对花括号括起来的一组语句）。如果在函数中声明变量，那么它是这个函数范围内的变量，如果在一个循环内声明变量，那么它的作用范围只是在这个循环之中，以此类推。

1.3.1 变量的初始化

（1）数值变量的初始化

在 C++中，数值变量的初始化可以在括号内进行，其格式如下：

```
数据类型 变量名（初值）;
```

比如下面的例子中，语句“int b(10);”就是这种定义模式。

```
#include "stdafx.h"
#include <iostream>
using namespace std;
void main()
{
  int a=3;              //初始值为 3
  int b(10);            //初始值为 10
  int result;           //不确定初始值
  result = a - b;
  cout << result;
}
```

(2) 非数值变量的初始化

字符串是用来存储一个以上字符的典型的非数值的变量。C++提供一个 string 类来支持字符串的操作，string 类并不是一个基本的数据类型，但在一般的使用中与基本数据类型很相似。

用户如果需要声明和使用字符串类型的变量，需要引用头文件<string>，如下面例子所示：

```
#include "stdafx.h"
#include <iostream>
#include <string>
using namespace std;
int main()
{
  string test_string;
  test_string = "Test String *** China *** ";
  cout << test_string << endl;
  test_string = "Test String *** Beijing *** ";
  cout <<test_string << endl;
  return 0;
}
```

(3) 常量的初始化

C++中常量的定义，是通过使用 const 关键字来完成的，用户可以定义指定类型的常量，就像定义一个变量一样：

```
const int x = 5;
const char y = 'n';
```

1.3.2 C++的输入与输出操作

(1) 输入操作

C++中的标准输入是通过在 cin 数据流上重载运算符“>>”来实现的，重载的问

题，在后面的内容中进行介绍，对于初学者，可以认为输入采用“cin >>”这个格式来完成。它后面必须跟一个变量以便存储读入的数据。例如：

```
int x;
cin >> x;
```

声明一个整型变量 x，然后通过 cin 并将输入值存储在这个变量 x 中。对于其他类型变量的输入，方法类似。

也可以利用 cin 输入多个数据，如：

```
cin >> x >> y;
```

等同于：

```
cin >> x;
cin >> y;
```

在以上两种情况下用户都必须输入两个数据，分别给变量 x 和 y。输入时两个变量之间可以使用空格符、Tab 符或回车。

值得注意的是，当使用 cin 和>>操作符来读取字符串时，例如：

```
cin >> test_ string;
```

由于“cin >>”只能读取一个单词，一旦碰到任何空格，读取操作就会停止，这样就无法输入一个英文句子。要解决此问题，可以使用 C++的 getline 函数，对于字符串的读入，更建议使用 getline 来进行。下面是一个读取字符串的例子：

```
#include "stdafx.h"
#include <iostream>
#include <string>
using namespace std;
int main()
{
  string test_string;
  cout << "你在哪个学校上学？";
  getline(cin,test_string);
  cout << "Hello "<< test_string << "\n";
  cout << "你喜欢什么专业？";
  getline(cin,test_string);
  cout << "我喜欢 "<< test_string << "专业\n";
  return 0;
}
```

（2）输出操作

运算符“<<”又叫插入运算符，因为它将后面所跟的数据插入到它前面的数据流中。插入运算符可以在同一语句中被多次使用，例如：

```
cout << "Hello! "<< "Visual C++ "<< "Object Oriented Programming";
```

初学者要注意输出字符串和输出变量的区别。请观察如下的代码：

```
cout << "abc";        //打印字符串 abc 到屏幕上
cout <<abc;          //把变量 abc 存储的内容打印到屏幕上
```

上面这一行语句将会打印 Hello! Visual C++ Object Oriented Programming 到屏幕上。插入运算符（<<）的重复使用在我们想要打印变量和内容的组合内容或多个变量时有所体现：

值得注意的是，cout 操作不会产生换行，因此，当需要换行的时候，可以输出一个换行符“\n”，如：

```
cout << "Hello! \n";
```

此外还可以用操作符 endl 来换行，例如：

```
cout << "Hello!" << endl;
cout << "Visual C++" << endl;
```

将会输出：

```
Hello!
Visual C++
```

(3) 字符串流

标准头文件 <sstream> 定义了一个叫做 stringstream 的类，使用这个类可以对基于字符串的对象进行操作，这对将字符串与数值互相转换非常有用。例如，如果想将一个字符串转换为一个整数，可以这样写：

```
string my_string("12345");
int my_int;
stringstream(my_string)>> my_int;
```

上述代码定义了一个字符串类型的对象 my_string，并赋初值为“12345”，然后接着定义了整型变量 my_int，最后调用 stringstream 类的构造函数，并以字符串变量 my_string 为参数。这段代码执行之后变量 my_int 存储的是数值 12345。下面是一个完整的实例体验此应用。

```
#include "stdafx.h"
#include <iostream>
#include <string>
#include <sstream>
using namespace std;
int main()
{ string str;
  float a=0.0;
  int b=0;
  cout << "a=?";
  getline(cin,str);
```

```
    stringstream(str)>> a;
    cout << "b=?";
    getline(cin,str);
    stringstream(str)>> b;
    cout << "a * b= "<< a * b << endl;
    return 0;
}
```

在上述代码的执行过程中，输入了变量 a 和 b 的值。但不同于从标准输入中直接读取数值，这里使用函数 getline 从标准输入流 cin 中读取字符串对象（my_str），然后再从这个字符串对象中提取数值 a 和 b。

当 cin 被用来输入字符串时，它通常与函数 getline 一起使用，方法如下：

```
cin.getline(char buffer[],int length,char delimiter = '\n');
```

其中：

- buffer：用来存储输入的地址，如一个数组名。
- length：一个缓存 buffer 的最大容量。
- delimiter：用来判断用户输入结束的字符，它的默认值是换行符（'\n'）。

下面的例子显示了如何使用 cin. getline 来输入字符串：

```
#include "stdafx.h"
#include <iostream>
using namespace std;
void main()
{
  char str[50];
  cout << "你在哪一所学校？";
  cin.getline(str,50);
  cout << "Hello "<< str << ".\n";
  cout << "你学什么专业？";
  cin.getline(str,50);
  cout << "我喜欢 "<< str << "专业\n";
}
```

（4）字符串和其他数据类型的转换

将字符串内容转换成数值型变量的功能在数据处理中经常会用到。例如一个字符串的内容可能是“1234”，如何转换为一个整数？因此，函数库 cstdlib 提供了 3 个有用的函数：

- atoi：将字符串 string 转换为整型 int。
- atol：将字符串 string 转换为长整型 long。
- atof：将字符串 string 转换为浮点型 double。

所有这些函数接收一个参数，返回一个指定类型的数据（int、long 或 float）。这 3 个函数与 cin. getline 一起使用来获得用户输入的数值，比传统的 cin>>方法更可靠。下

面的例子体现其应用：

```
#include "stdafx.h"
#include <iostream>
#include <cstdlib>
using namespace std;
void main()
{
  char str[50];
  double a;
  int b;
  cout << "a=?";
  cin.getline(str,50);
  a = atof(str);
  cout << "b=? ";
  cin.getline(str,50);
  b = atoi(str);
  cout << "a * b= "<< a * b;
}
```

上述代码的执行过程中，如果对第一个 str 输入字符串 11，对第二个 str 输入字符串 22，那么转换成整型后分别赋给变量 a 和 b，相乘后输出结果为 242。

1.4 动态内存分配

目前我们所接触到的程序，所声明的变量、数组和其他对象所必需的内存空间都是确定的。但如果需要内存大小为一个变量，其数值只有在程序运行时才能确定，那该如何处理呢？可以采用 C++提供的动态内存分配方法来处理，为此 C++集成了操作符 new 和 delete。

(1) 操作符 new 和 new[ ]

用操作符 new 开辟动态内存空间，关键字 new 后面跟一个数据类型，并跟一对可选的方括号“ [ ] ”，里面为要求的元素数。其结果返回一个指向内存块起始位置的指针。其形式为：

```
p = new type
```

或者

```
p = new type[elements]
```

第一个表达式用来给一个元素的数据类型分配内存。第二个表达式用来给一个数组分配内存。

例如：

试读结束：需要全本请在线购买： www.ertongbook.com