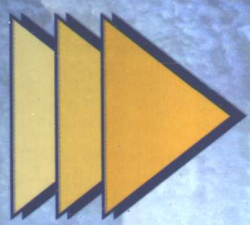
高等学校教材

大学Visual C++程序设计案例教程

罗建军 崔舒宁 杨 琦 编著
冯博琴 审



高等教育出版社
HIGHER EDUCATION PRESS

内 容 提 要

本书按照教育部高等学校计算机基础课程新的教学基本要求组织编写。全书以 Visual C++案例开发为特色，深入分析了大量开放性项目案例的开发过程，使学习者不仅能领悟程序设计所必须掌握的知识重点，也能熟悉开发软件项目的各个环节，从而真正学到应用软件的开发技术。

本书内容分为两个主要部分：第一部分为关键技术，简明地介绍了 Windows 编程的五大知识点，读者可以根据自身情况参考学习；第二部分为实用案例，提供了大量经过多年实践检验的、具有一定复杂度和代表性的案例，读者可以通过阅读案例分析说明文档，调试修改本书中的代码，进而从这些精心准备的案例中获得扎实的程序开发技能。本书还提供了一个有关调试技术的附录，供读者上机实验时参考使用。

本书适用对象为具有一定 C 或 C++基础，希望学习基于 Windows 可视化编程的读者，可作为高等院校计算机及相关专业的教材或参考书，也可供应用开发人员学习参考。

本书支持网站为西安交通大学计算机教学实验中心的网站（详见 http://ctec.xjtu.edu.cn 相关版块），提供了一个全交互性、立体化的网络教学环境（包括课件、代码、讨论答疑区和最新学习指导信息等），所有内容都在不断更新，供教师教学和学生学习使用。

图书在版编目（CIP）数据

大学Visual C++程序设计案例教程/罗建军，崔舒宁，杨琦编著.—北京：高等教育出版社，2004.8
ISBN 7-04-015504-4

Ⅰ.大… Ⅱ.①罗…②崔…③杨… Ⅲ.C语言-程序设计-案例-高等学校-教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2004）第 077605 号

出版发行	高等教育出版社	购书热线	010-64054588
社　　址	北京市西城区德外大街 4 号	免费咨询	800-810-0598
邮政编码	100011	网　　址	http://www.hep.edu.cn
总　　机	010-82028899		http://www.hep.com.cn
经　　销	新华书店北京发行所		
印　　刷	济南新华印刷厂		
开　　本	787×1092　1/16	版　　次	2004 年 8 月第 1 版
印　　张	20.25	印　　次	2004 年 8 月第 1 次印刷
字　　数	470 000	定　　价	24.30 元

前　言

目前支持 Windows 环境的通用程序设计语言工具很多，如 Microsoft Visual C++、Visual Basic、DELPHI 和 Java 等。这些语言各有特点，但从性能指标和应用范围来看，Visual C++ 无疑是其中的佼佼者。它支持面向对象的程序设计方法，支持 MFC 类库编程，有完善的集成开发环境，可用来开发各种类型、不同规模和复杂程度的应用程序，开发效率很高，生成的应用软件代码品质优良，是许多专业程序开发人员的首选编程语言。

但另一方面，Visual C++ 是一个非常复杂的软件系统，所以也有难于学习的“恶名”，被初学者视为畏途。

通过四五年的 Visual C++ 教学实践，我们发现仅仅讲授、强调基本概念和原理方法对提高学习者的 Visual C++ 编程能力是不够的，要想真正获得较高的程序开发技能，必须以实际软件开发为主线，通过学习开放性的项目案例的研究开发过程，提高全面的应用开发能力（包括功能设计、编写代码、程序调试等各个方面），这样才能使学习者彻底吃透必须掌握的知识重点，熟悉开发软件项目的各个环节，从而真正掌握应用软件的开发技术。

正是基于这种考虑，我们将本书分为两部分：第一部分为关键技术，包含五章，分别对应 Windows 编程的五大知识点；第二部分为实用案例，提供了大量的具有一定复杂度和代表性的案例。本书还提供了一个有关“调试技术”的附录，主要是讲解如何利用 Visual C++ 的 Developer Studio 集成开发环境来调试、连接和运行 C++ 程序。

为了便于读者学习，第一部分的每章均按以下主题进行组织：

知识点　言简意赅地讲解了 Windows 编程的某一方面内容的主要知识点。

基本范例　提供了针对主要知识点的基本程序设计例子。通过这些简单的例子，读者可以对主要知识点的实际应用有一个直观的认识。适用于对 Visual C++ 编程不太熟悉的读者参考使用，有一定基础的读者可以跳过该部分。

典型案例　提供了 33 个具有一定复杂度和代表性的案例的描述，在本书的第二部分，作者对这些案例进行了详尽而全面的讲解，读者可以从中领会如何将一个具有一定复杂性的问题分解实现乃至完成。每个案例也都具有一定的开放性，读者可以在提供案例工作的基础上进一步去完善，从而更进一步地理解升华所学到的技能，培养开发小型软件的能力。

本书的构思和编写得到了冯博琴教授的多方指导，并由他审读了书稿，同时刘路放教授对本书所有案例的早期组织建设工作起到了关键作用，在此谨向冯老师和刘老师表示深深的谢意。参加编写工作的队伍由西安交通大学计算机教学实验中心长期从事程序设计教学和科研的一线教师组成，主要人员有：罗建军（本书第一部分，第二部分 7 个案例，附录），崔舒宁（第二部分 7 个案例，所有案例组织整理），杨琦（第二部分 5 个案例，文字校对），朱丹军、仇国巍、薛涛、卫颜俊、张伟、吕军等也参加了本书部分案例的整理工作。

由于作者学识有限，编写时间仓促，书中错误和片面之处在所难免，恳请读者不吝批评指正，编者的电子邮件是 jjluo@ ctec. xjtu. edu. cn。

编　者

2004 年 7 月

目　录

第一部分　关键技术

第二部分　实用案例

第一部分

关键技术

◇ Windows 与 Visual C++ 基础

◇ Windows 消息机制

◇ 图形设备接口和资源编程

◇ 文档/视图结构

◇ 对话框

第一章　Windows 与 Visual C++基础

知识点

目前，Microsoft Windows 已成为微机上的主流操作系统，几乎一统天下，在 Windows 平台上进行软件开发也已成为程序设计的主流。

为了适应 Windows 编程，各软件厂商纷纷推出了新型 C++编译器，Microsoft 公司的 Visual C++就是其中比较优秀的一种。它并不是一个单纯的编译器，而是一整套用于软件开发的集成环境，其中包括了文本编辑、编译调试、可视化界面设计和在线帮助等。

1.1　Windows 的用户界面对象

Windows 操作系统具有图形用户界面和多任务、多窗口等特点。它支持丰富的用户界面对象，包括窗口、图标、菜单、对话框等。程序员只需编写简单的几十行代码，就可以设计出一个非常漂亮的图形用户界面。下面就介绍几个常用的用户界面对象的术语和相关概念。

1.1.1　窗口

窗口是用户界面中最重要的部分，是屏幕上与一个应用程序相对应的矩形区域，是用户与产生该窗口的应用程序之间的可视界面。每当用户开始运行一个应用程序时，应用程序就创建并显示一个窗口；当用户操作窗口中的对象时，程序会作出相应反应。用户通过关闭一个窗口来终止一个程序的运行；通过选择相应的应用程序窗口来选择相应的应用程序。

1.1.2　系统菜单

系统菜单图标位于窗口左上角，用鼠标单击一下该图标（或按 Alt + 空格键）就可弹出系统菜单。系统菜单提供标准的应用程序选项，包括还原、移动、大小、最小化、最大化和关闭。

1.1.3　标题栏

标题栏位于窗口的顶部，其中显示的文本信息用于标注应用程序的名字。鼠标双击标题栏可使窗口在正常大小和最大化状态之间切换。在标题栏上按下鼠标左键可以拖动并移动该窗口，单击右键将弹出系统菜单。

1.1.4　菜单栏

菜单栏位于标题栏下方，横跨窗口，上面列出了应用程序所支持的命令。选中菜单栏某

一项通常会显示一个弹出菜单,其中的项是对应于指定命令或类中的某个任务。通过选择菜单中的菜单项,用户可以向程序发出命令,以执行某一功能。

1.1.5　工具条

工具条位于菜单栏下方,上面有一组位图按钮,代表一些最常用的命令。工具条可以显示或隐藏。用户还可以用鼠标拖动工具条,将其放在窗口的任何一侧。

1.1.6　客户区

客户区是窗口中最大的一块空白矩形区域,用于显示应用程序的输出。例如,字处理程序在客户区中显示文档的当前页面。应用程序负责客户区的绘制工作,且只有和该窗口相对应的应用程序才能向该客户区输出。

1.1.7　垂直滚动条和水平滚动条

垂直滚动条和水平滚动条分别位于客户区的右侧和底部。可以用鼠标选中滚动条的箭头上下或水平卷滚客户区的内容。滚动块的位置表示客户区中显示的内容相对于要显示的全部内容的位置,滚动块的长度表示客户区中显示的内容相对于全部内容的比例。

1.1.8　状态栏

状态栏一般位于窗口底部,用于输出菜单的说明和其他一些提示信息(如鼠标位置、当前时间、某种状态等)。

1.1.9　图标

图标是一个用于提醒用户的符号,它是一个小小的图像,用于代表一个应用程序。当一个应用程序的主窗口缩至最小时,就呈现为一个图标。

1.1.10　光标

光标是显示屏上的一个位图,用于响应鼠标或其他定位设备的移动。程序可以通过改变光标的形状来反映系统中的变化。例如,绘图程序经常通过改变光标来反映被绘制对象的类型是直线、圆或其他。

1.1.11　插入符

插入符是一个很小并闪烁的位图,作为键盘位置指示。Windows 在同一时间只允许有一个插入符存在。

1.1.12　对话框

对话框是一种特殊的窗口,它提供了一种接收用户输入、处理数据的标准方法。在一般

情况下,在选择菜单名字后面跟着省略号(...)的菜单项通常会弹出一个对话框。

1.1.13 控件

在对话框中还包含了许多称为控件的小窗口。它是应用程序用来获得用户特定信息的窗口。应用程序也会通过控件获取所需的信息,以便控制程序的某种属性。

1.2 Windows 编程的主要概念

无论使用哪种开发工具,对 Windows 编程都要面对一些特色问题,它不同于基于传统的单任务 DOS 操作系统的字符界面编程技术。

1.2.1 事件驱动

程序的控制流程是由实际运行时各种事件的实际发生来触发,没有预定的顺序,允许用户用各种合理的顺序来安排程序的流程。事件驱动是靠消息循环机制实现的,所以也称为消息驱动。

Windows 操作系统不仅执行程序,而且还会与应用程序进行对话。它对消息有一套完善、严格的定义,并在其产生时将其发送至所有相关的应用程序,这些消息用于驱动应用程序运行以实现一定的功能。

例如,当用户单击鼠标左键或按下键盘上的某一个键时,都将发送消息,然后这些消息将由操作系统负责发送至所有相关的应用程序。

除 Windows 提供的大量标准消息外,程序员也可以根据需要自己定义一些消息。

1.2.2 设备无关性

设备无关性也称设备独立性。在 Windows 程序中,应用程序的输出并不直接面向物理设备(如显示屏或打印机),而是针对一个称之为设备环境(Device Context)的逻辑设备进行操作。设备环境与实际设备之间的信息传送由 Windows 直接管理,所以在具体编程时程序员无需了解他所要处理的不同设备的内部指令。Windows 操作系统所提供的各种抽象接口,使得在程序中可以通过调用标准函数与硬件交互。这些函数通过设备环境由操作系统将其映射到相应的物理设备,而程序员就不必关心诸如打印机、显示器、声卡、显卡等具体设备类型及其驱动程序。

1.2.3 资源管理

资源是一种数据,它包含了决定程序的用户界面的文本和图形。程序员设计任何 Windows应用程序均将涉及到诸如菜单、加速键、位图、图标、对话框、控件、字符串以及工具栏等资源数据。Windows 将这些数据保存在资源文件中,程序员可以通过编程工具编辑、修改这些文件,并将其放入设计的程序之中。

1.3　Windows 应用程序类型

典型的 Windows 应用程序有以下四种类型：

（1）控制台应用程序：控制台应用程序结构简单，可以不使用 MFC 类库。

（2）基于框架窗口的应用程序：主应用程序窗口为框架窗口，CFrameWnd 派生类对象附属于应用程序的 CWinApp 派生类对象的 m_pMainWnd 成员。适用于某些仅需最小的用户界面和简单的窗口结构的应用程序。

（3）基于对话框的应用程序：与基于框架窗口的应用程序差别不大，只是用 CDialog 派生类对象代替了 CFrameWnd 派生类对象作为应用程序的主窗口。

（4）基于文档/视图结构的应用程序：具有较复杂的结构和较强的功能，基于文档/视图结构的应用程序又可分为单文档界面（Single Document Interface，SDI）和多文档界面（Multiple Document Interface，MDI）。

1.4　MFC 应用程序框架

Microsoft 提供了一个基础类库（Microsoft Foundation Class，MFC），其中包含用来开发 C++应用程序和 Windows 应用程序的一组类。

MFC 中的类可分为两种：CObject 类的派生类及非 CObject 派生类。非 CObject 派生类数量不多，但大多很常用，如 CTime、CTimeSpan、CString 和 CFile 等。CObject 是大多数 MFC 类的根类或基类，它具有很多有用的特性，如支持序列化、运行时类信息访问、对象诊断输出和与集合类兼容等。

MFC 将 Windows 应用程序从开始运行、消息传递到结束运行所需的各步骤均封装在 CWinApp 类中，CWinApp 类表示 MFC 应用程序的应用对象。CWinApp 类由 CObject 类的子类 CWinThread 类（定义 MFC 内的线程行为）派生。一个 MFC 应用程序必须有且只能有一个从 WinApp 类派生的全局应用程序对象，此对象在运行时刻控制应用程序中所有其他对象的活动。

使用 MFC 可以大大简化 Windows 编程工作。它将各种类结合起来构成了一个应用程序框架，其目的就是让程序员在此基础上建立 Windows 应用程序。因为，总体上 MFC 框架定义了应用程序的轮廓，并提供了用户接口的标准实现方法，程序员所要做的仅仅是通过预定义的接口把具体应用程序所特有的东西填入这个框架内。而且 Visual C++也提供了相应的工具来完成这个工作：AppWizard 用来生成初步的框架文件；资源编辑器用于帮助直观地设计用户接口；ClassWizard 用来协助添加代码；类库实现了应用程序特定的逻辑等。

1.5　MFC 编程

下面以一个最简单的 SDI 程序为例来学习如何使用 MFC 应用程序框架进行编程（具体

上机过程请参考附录 Visual C++ 调试技术)。

运行 AppWizard 来产生一个新项目。输入工程名,并确保工程类型(Projects)选择 MFC AppWizard(exe),在"下一步"(MFC AppWizard Step 1)中选择 Single document 选项,其余步骤均采用默认参数。

项目生成后,可直接编译、运行。该程序显示了一个完整的 Windows 窗口,还有与窗口关联的菜单。这就是应用程序框架的工作方式:提供一个可运行的空程序,仅实现一些通用的基本功能,其他功能由程序员去实现。理解应用程序框架的程序结构,尤其是找到加入代码的地方并加入适当的代码,就是使用应用程序框架编程的第一步。

通过 Workspace 的 ClassView(类视图)可以看到,AppWizard 生成了五个类,对于名为 My 的项目,它们是:

CAboutDlg	about 对话框类
CMainFrame	框架类,由 CFrameWnd 派生
CMyApp	应用程序类,由 CWinApp 派生
CMyDoc	文档类,由 CDocument 派生
CMyView	视图类, 由 CView 派生

这些类之间有相当复杂的关系,在第四章将对其做详细介绍。

除此之外,在程序中还声明了一个 CMyApp 类的全局对象 theApp。在程序运行时,MFC 应用程序首先调用由框架提供的标准的 WinMain()函数。在 WinMain()函数中,先初始化由 CMyApp 定义的惟一全局对象 theApp(通过重载的虚函数 InitInstance(),它构造并显示应用程序的主窗口),然后调用其由 CWinApp 类继承的 Run()成员函数,进入消息循环。程序结束时调用 CWinApp 的 ExitInstance()函数退出。

因此,应用程序框架不仅提供了构建应用程序所需要的类,还规定了程序的基本执行结构。所有的应用程序都在这个基本结构的基础上完成不同的功能。

1.6 在窗口的客户区输出文字和图形

OnDraw 是 CView 类中的一个虚成员函数,用于绘制窗口客户区内容。每次当窗口需要被重新绘制时,应用程序框架都要调用 OnDraw 函数。

在 MFC 下,使用 CDC 的成员函数来完成所有的窗口绘制工作,主要的函数有:

1. 文字信息显示

```
BOOL TextOut(int x, int y, LPCTSTR lpszString);
```

在指定坐标(x, y)处显示字符串 lpszString 的内容,显示成功返回非零值,否则返回零。坐标原点(0,0)在客户区左上角,Y 轴向下。下面各成员函数的坐标参数均同此。

2. 画线

画线工作需经两步完成:首先确定线的起始端位置,这可通过调用成员函数 MoveTo 完成,其原型为:

```
CPoint MoveTo (int x, int y);
```

```
CPoint MoveTo (POINT point);
```

MoveTo 将绘图位置移至指定坐标处，并返回移动前的绘图位置。确定了线的起点后，即可使用成员函数 LineTo 画线：

```
BOOL LineTo (int x, int y);
BOOL LineTo (POINT point);
```

其参数为线终点的坐标。

3. 绘制矩形

```
BOOL Rectangle (int x1, int y1, int x2, int y2);
BOOL Rectangle (LPCRECT lpRect);
```

其参数为需要绘制的矩形的左上角坐标(x1, y1)和右下角坐标(x2, y2)。

4. 绘制椭圆

```
BOOL Ellipse(int x1, int y1, int x2, int y2);
BOOL Ellipse(LPCRECT lpRect);
```

其参数的含义为所绘椭圆的包含矩形的左上角和右下角坐标。

5. 画多边形

```
BOOL Polygon (LPPOINT lpPoints, int nCount);
```

其中参数 lpPoints 为一 LPPOINT 类型的指针，可用 CPoint 数组(存放多边形的各顶点坐标)作为实参。参数 nCount 为顶点个数。例如

```
CPoint pointPoly[3];
pointPoly[0] = CPoint(100, 100);
pointPoly[1] = CPoint(200, 100);
pointPoly[2] = CPoint(200, 200);
pDC -> Polygon(pointPoly, 3);
```

在窗口客户区相应位置画出一个三角形。

6. 其他绘图函数

还有画点 SetPixel()、画弧 Arc()、画弓形 Chord()、画扇形 Pie()和画圆角矩形 InvertRect()等，具体使用方法可参看 MSDN 联机帮助。

1.7　使客户区重绘

每次当窗口需要被重新绘制时，应用程序框架都要调用 OnDraw 函数。当窗口生成、用户改变了窗口尺寸，或者当窗口恢复了先前被遮盖的部分时，OnDraw 都会被自动调用。但当程序中某个函数(如 OnLButtonDown)修改了数据而需要重绘窗口时，OnDraw 并不会被自动调用。这时，需要调用 CWnd 类的 Invalidate 或 InvalidateRect 成员函数来触发 Windows 的 WM_PAINT 消息，从而引起对 OnDraw 的调用。

这两个函数的原型是：

```
void Invalidate(BOOL bErase = TRUE);
```

```
void InvalidateRect(LPCRECT lpRect, BOOL bErase = TRUE);
```

其中,InvalidateRect 多了一个参数 lpRect,一般用 CRect 类型,用来指明需要重新绘制的区域(无效区域),而 Invalidate 则要求重绘全部窗口(当快速更新数据并重绘时,会引起屏幕闪烁)。参数 bErase 指明重绘时对背景的处理方式,它为 TRUE 时,要求擦去背景重绘,为 FALSE 时,不要求重绘背景。缺省的参数值为 TRUE。

1.8 Windows 数据类型

Windows 应用编程接口(Application Programming Interface,API)自行定义了一些关键字,用来定义 Windows 函数中的有关参数和返回值的大小和意义,通常将它们看做Windows 的数据类型。其中较常用的有:

关键字	类型	说明(等价的类型)
BOOL	逻辑类型	int
CHAR	字符	char
DOUBLE	双精度	double
FLOAT	浮点数	float
HANDLE	句柄	void
INT	整数	int
LONG	长整数	long
UCHAR	无符号字符	unsigned char
UINT	无符号整数	unsigned int
VOID	空的、无定义	void
WCHAR	双字节码	unsigned short
WPARAM	消息参数	UINT
LPARAM	消息参数	LONG
LRESULT	消息返回值	LONG
HINSTANCE	实例句柄	unsigned long
HWND	窗口句柄	unsigned long
HDC	设备环境句柄	unsigned long
TCHAR	字符	char
LPSTR	字符指针	char *
LPCSTR	常量字符指针	const char *
LPTSTR	字符指针	TCHAR *
LPCTSTR	常量字符指针	const TCHAR *
LPVOID	无类型指针	void *
LPCVOID	无类型常量指针	const void *

其中句柄(handle)是一个 4 字节长的惟一的数,用以标识许多不同的对象类型,如窗口、菜单、内存、画笔、画刷等。

除了这些基本数据类型外,使用 MFC 编程时还会遇到如下的数据类型: