北京科海培训中心

以完成作品为教学导向，真正整合理论与实务

Visual C++ 案例教程

邓华 毛岩 吉正 编著

- 从界面设计、系统编程、数据库开发、组件对象模型和网络程序设计5个方面介绍了使用Visual C++编写Windows程序的方法和技巧，使读者能够在最短的时间内学会编写各种应用程序
- 通过讲解在MFC类库的基础上构造大量实用的类来完成案例程序的各种功能，使读者能深入了解使用MFC编写应用程序的实质，并举一反三，构造出更多更好的类，开发出功能更强大，使用更简单的应用程序
- 光盘包括每个案例的完整源代码和执行程序，另外还有200多个实用控件

- 案例源文件导读
- 实用控件大派送

中科多媒体电子出版社

北京科海培训中心

Visual C++案例教程

邓华 吉正 毛岩 编著

中科多媒体电子出版社

2001.8

内 容 提 要

Visual C++是 Microsoft 开发的一个功能强大的 C++语言开发工具，它为使用 C++语言进行程序设计提供了可视化的集成开发环境。本书通过若干个简明实用的编程实例，全面介绍了使用 Visual C++编写 Windows 程序的方法和技巧，从界面设计、系统编程、数据库开发、组件对象模型和网络程序设计五个方面进行了介绍，使读者能够在最短的时间内快速掌握各种应用程序的编写方法，成为一名真正的 Visual C++程序开发人员。

本书结构清晰，内容全面，注重实用。以丰富的案例来介绍抽象概念和具体技术的实质，这也是本书的最大特色。

本书面向中高级程序员、软件开发人员以及广大计算机编程的业余爱好者，要求读者具备一定的 C 语言基础和 Visual C++使用经验。

本书中所有案例程序都在中文 Windows 系统、Visual C++ 6.0 中文版环境下编译通过。由于篇幅有限，书中未完整地列出每个案例的代码，只讲解了其中的重点部分，详细的代码请查阅光盘。另外，光盘上还放置有一些实用的控件，相信能提高你的编程效率。

品　　名：Visual C++案例教程
作　　者：邓华　吉正　毛岩
责任编辑：马首鳌
出　　品：中科多媒体电子出版社
印 刷 者：北京门头沟胶印厂
发　　行：新华书店总店北京科技发行所
开　　本：787×1092　1/16　　印张：28　　字数：658 千字
版　　次：2001 年 8 月第 1 版　　2001 年12月第 2 次印刷
印　　数：5001~7000
盘　　号：ISBN 7-900084-03-7
定　　价：39.00 元（1CD）

前　言

何为Visual C++?

作为Windows环境下的程序开发工具，Visual C++倍受用户青睐，它提供的可视化开发环境可供C语言程序人员快速简便地开发出功能强大的Windows程序，实现开发人员的各种梦想。

Visual C++为我们提供了功能丰富的编程工具，包括编辑器、测试容器和类库等等。可视化开发工具提供了一条简便的途径，使初学者能够快速掌握Visual C++程序设计的基本方法。

如何学好Visual C++?

在程序设计领域，有很多种语言可供你选择，但是，如果你想成为一名真正的软件设计人员，C++是你最理想的选择。“聪明的程序员用Delphi，真正的程序员用Visual C++”，这是Delphi程序员经常引用的一句话。其原意是想说明Delphi比Visual C++更简单易用。的确，在界面设计、数据库等许多方面Delphi具有更加直观的集成环境支持，由此而成为倍受初学者欢迎的一种编程工具。但是正如他们所说的，真正的程序员用Visual C++。这是因为Visual C++能够为程序员提供更多的控制，允许程序员更加自由地发挥自己的设计才能。刚使用Visual C++的时候会觉得很不习惯，尤其对于那些曾经使用过Delphi或Visual Basic的程序员来说，这经常会让他们望而却步。但是，Visual C++更像是一个无底洞，越往里钻，会有越来越多意想不到的收获。时间越长，就会觉得它越来越好用。所以，如果你想成为一名Visual C++高手，首先就要认识到学习使用Visual C++编程是永无止境的一种追求，只有在不断的实践中，你才能够不断地取得进步。通过阅读别人的程序，从中吸取编程思想的精华，使之成为自己的经验，这就是学习Visual C++的最好方法。

本书的目的

前面已经提到，学习Visual C++最理想的途径是阅读程序。然而，阅读别人的程序是一件相当痛苦的事情，因为每个人都会有自己独特的想法，体现在程序上就是一些难以理解的代码，这就是学习过程中最大的障碍。本书的目标是为读者朋友们提供若干个实用的编程案例，通过简明扼要的讲解，使读者能够在最短的时间内领会到程序设计中的精华，达到学习的目的。同时，这些案例程序也能够为读者朋友们提供一种参考，成为解决难题的帮手，为自己省去大量的摸索时间。

本书的特色

学习Visual C++是一种永无止境的追求，因此，不可能在一本书中包罗所有的内容。我们更看重编程的实际能力，而不仅仅是一些小技巧。因此本书把大量的注意力放在编程

思想的体现上，力争每一个案例都能够使读者朋友们得到不同程度的收获。本书不是一本大而全的编程指导书，然而里面出现的案例都是作者根据自己的编程经验总结出来的，具有很强的实用价值。

本书的结构

全书分为5章，分别从界面设计、系统编程、数据库开发、组件对象模型和网络程序设计5个角度介绍了设计Visual C++程序的方法和技巧。

第1章介绍了在Windows界面程序设计中经常遇到的问题，如图形绘制、用户交互、对话框、文档和视图等等。

第2章介绍了如何通过Visual C++设计系统控制程序，包括文件操作、线程控制、后台处理、音频视频等内容。

第3章以完整的案例介绍了在Visual C++环境下进行数据库开发的各种基本技巧。

第4章介绍了如何开发当前最流行的ActiveX服务器和组件。组件对象模型COM历来是程序开发中的难点，本书力图以简洁的案例来说明开发的基本过程。

第5章介绍了如何在Windows环境下利用Socket编写网络通信程序。这里举的例子都是经常使用的网络程序，如Ping、FTP、电子邮件发送和聊天室，相信读者朋友们一定会喜欢。

读者对象

本书以案例为编排格式，对基本的编程语言知识和操作过程不作详细的介绍，因此要求读者具备一定的编程基础以及Visual C++使用经验。如果你已经具备了前述条件，相信在读完本书之后，你的编程技能将更上一层楼！

光盘使用说明

运行环境：

- Pentium 166以上的处理器
- VGA显卡
- 光盘驱动器
- Windows 95/98/NT/2000操作系统
- Internet Explorer 4.0及以上版本浏览器
- 800×600分辨率
- 16位真彩色以上显示模式

操作方法：

- 一般情况下，将本光盘放到光驱中后，光盘就会自动运行。
- 如果本光盘没有自动运行，请双击光盘根目录下的Start.exe或Index.htm文件即可开始运行。
- 如果计算机上没有安装Winzip解压软件，请先运行光盘根目录下的Winzip80.exe程序进行安装。

光盘内容：

- 案例源文件。
- 实用控件（注意有些控件使用前需要注册）。

目 录

第1章 界面设计

设计界面是开发程序中一项基本但却是十分重要的工作，想要很好的掌握它并不是件容易的事。本章精选了20个与界面设计有关的案例，内容涉及到控件、对话框、工具栏、状态条、标题栏等的使用以及动画的制作。为了能让大家学到更多的编程技巧，本章没有去介绍基本的Visual C++控件的使用，而是通过使用一些新的且十分实用的类来完成案例，相信会对大家很有帮助。同时，希望读者能够在此基础上举一反三，构造出更多更好的类，开发出丰富多彩的程序界面。

案例1.1 平铺风格的编辑控件

〖案例说明〗

当你要获得来自用户的文本信息，或者要让用户输入或编辑文本的时候，应该选择编辑控件。编辑控件是一个看起来非常简单的矩形窗口。本例将介绍一种具有平铺风格的编辑控件，我们可以使用它来改善我们的程序界面。本例程序的运行效果如图1.1及图1.2所示。

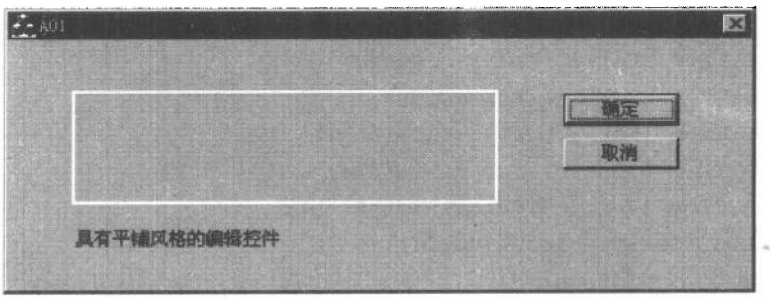

图 1.1 开始时的编辑控件

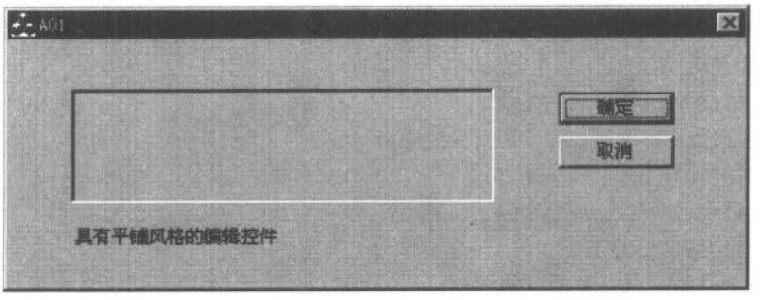

图 1.2 鼠标移动上去时的编辑控件

〖设计思想〗

MFC通过CEdit类提供了标准的Windows编辑控件。这里，我们为了要实现具有平铺风格的编辑控件，需要使用一个基于CEdit类的新类CEditFlat。然后在程序中的所有声明CEdit的地方，将CEdit换成CEditFlat就行了。下面我们来看一下类CEditFlat的结构：

```
class CEditFlat : public CEdit
    {
    public:
       // Default constructor
       CEditFlat();
       // Destructor
       virtual ~CEditFlat();
       // 重载函数声明
       //{{AFX_VIRTUAL(CEditFlat)
    protected:
       virtual void PreSubclassWindow();
       //}}AFX_VIRTUAL
    protected:
       BOOL DrawEdit (VOID);
       // 主程序句柄
       HINSTANCE m_hInstance;
       // 变量声明
       BYTE m_bHorizontalFrameWidth;
       BYTE m_bVerticalFrameWidth;
       BYTE m_iHorizontalScrollWidth;
       BYTE m_iVerticalScrollWidthLeft;
       BYTE m_iVerticalScrollWidthRight;
       BOOL GetSysColors (VOID);
       BOOL m_bIsFocused;
       // 系统颜色
       COLORREF m_clrButton;
       COLORREF m_clrDarkShadow;
       COLORREF m_clrShadow;
       COLORREF m_clrHiLite;
       // 鼠标移动到控件上时对应的变量
       BOOL m_bMouseOnEdit;
       BOOL m_bHScrollPressed;
       BOOL m_bVScrollPressed;
       CString m_StrWaveName;
       //{{AFX_MSG(CEditFlat)
       afx_msg void OnMouseMove(UINT nFlags, CPoint point);
       afx_msg void OnPaint();
```

```
        afx_msg void OnSetFocus(CWnd* pOldWnd);
        afx_msg void OnKillFocus(CWnd* pNewWnd);
        afx_msg void OnSysColorChange();
        afx_msg void OnStyleChanging( int nStyleType,
            LPSTYLESTRUCT lpStyleStruct);
        afx_msg void OnVScroll(UINT nSBCode, UINT nPos,CScrollBar*
            pScrollBar);
        afx_msg void OnHScroll(UINT nSBCode, UINT nPos,CScrollBar*
            pScrollBar);
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
    };
```

通过运用此类，我们就可以得到一个具有平铺风格的编辑控件了。

〖设计步骤〗

1. 用MFC AppWizard（exe）创建一个新工程，将其命名为A01。

2. 在MFC AppWizard第1步中设置应用程序的类型为对话框模式，然后直接单击“完成”按钮，应用程序A01就创建完毕。

3. 打开对话框编辑器编辑对话框IDD_A01_DIALOG，向其中加入1个编辑控件，设置其属性如表1.1所示。

表 1.1 对话框 IDD_A01_DIALOG 的设置

资源标识	标题	相关属性/事件
IDC_STATIC	具有平铺风格控件	
IDC_EDIT1		CEditFlat m_edit1
IDOK	退出	

4. 编写EditFlat.h以及EditFlat.cpp文件，构建新类CEditFlat。

5. 修改代码。在CA01Dlg.h文件中加入如下代码：

```
#include "EditFlat.h"
……
public:
……
//{{AFX_DATA(CA01Dlg)
……

//声明CEditFlat类的变量
CEditFlat   m_edit1;
//}}AFX_DATA
```

在CA01Dlg::DoDataExchange(CDataExchange* pDX)中加入如下代码：

```
//将声明的m_edit1变量和对话框控件联系起来
DDX_Control(pDX, IDC_EDIT1, m_edit1);
```

〖代码分析〗

```
/////////////////////////////////////////////////////////////////////////////
//      EditFlat.cpp: implementation for the CEditFlat class.          //
//                        CeditFlat类的实现模块                          //
/////////////////////////////////////////////////////////////////////////////
......
//消息映射
BEGIN_MESSAGE_MAP(CEditFlat, CEdit)
    //{{AFX_MSG_MAP(CEditFlat)
    ON_WM_MOUSEMOVE()
    ON_WM_PAINT()
    ON_WM_SETFOCUS()
    ON_WM_KILLFOCUS()
    ON_WM_SYSCOLORCHANGE()
    ON_WM_VSCROLL()
    ON_WM_HSCROLL()
    ON_WM_STYLECHANGING()
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

// CEditFlat构造函数
CEditFlat::CEditFlat()
    {
     // 初始化所有变量
     CWinApp *pMainApplication;
     pMainApplication = AfxGetApp();
     m_hInstance = pMainApplication->m_hInstance;
     m_bVScrollPressed = TRUE;
     m_bHScrollPressed = TRUE;
     m_bMouseOnEdit = FALSE;
     m_bIsFocused = FALSE;
     m_iHorizontalScrollWidth = NULL;
     m_iVerticalScrollWidthLeft = NULL;
     m_iVerticalScrollWidthRight = NULL;
     m_bHorizontalFrameWidth = NULL;
     m_bVerticalFrameWidth = NULL;
     GetSysColors();
     }
```

```
// CEditFlat析构函数
CEditFlat::~CEditFlat()
    {
    }

// 鼠标移动响应函数
void CEditFlat::OnMouseMove(UINT nFlags, CPoint point)
    {
     CEdit::OnMouseMove(nFlags, point);
     if (m_bIsFocused) return;
     if (GetCapture() != this)
        {
         m_bMouseOnEdit = TRUE;
         SetCapture();
         DrawEdit();
        }
     else
        {
         CRect RectEditControl;
         GetClientRect(&RectEditControl);
         if (!RectEditControl.PtInRect(point))
            {
             m_bMouseOnEdit = FALSE;
             DrawEdit();
             ReleaseCapture();
            }
        }
    }

// 绘制函数
void CEditFlat::OnPaint()
    {
     // 默认过程
     Default ();
     // 调用绘制控件函数
     DrawEdit();
    }

// 绘制平铺风格的编辑控件
BOOL CEditFlat::DrawEdit(VOID)
    {
     CRect RectEditControl;
     GetClientRect (&RectEditControl);
     CDC *pEditControlDC;
```

```
    pEditControlDC = GetDC ();
    // 设置矩形框的属性
    RectEditControl.top -= m_bHorizontalFrameWidth;
    RectEditControl.bottom += m_iHorizontalScrollWidth;
    RectEditControl.bottom += m_bHorizontalFrameWidth;
    RectEditControl.right  += m_iVerticalScrollWidthRight;
    RectEditControl.right  += m_bVerticalFrameWidth;
    RectEditControl.left -= m_iVerticalScrollWidthLeft;
    RectEditControl.left -= m_bVerticalFrameWidth;
    if (m_bMouseOnEdit)
       {
        // 鼠标在编辑控件上
        pEditControlDC->Draw3dRect(&RectEditControl,m_clrShadow,
           m_clrHiLite);
        RectEditControl.DeflateRect (1,1);
        pEditControlDC->Draw3dRect(&RectEditControl,m_clrDarkShadow,
           m_clrButton);
       }
    else
       {
        // 鼠标不在编辑控件上
        pEditControlDC->Draw3dRect(&RectEditControl,m_clrHiLite,
           m_clrHiLite);
        RectEditControl.DeflateRect (1,1);
        pEditControlDC->Draw3dRect(&RectEditControl, m_clrHiLite,
           m_clrHiLite);
       }
    // 释放设备环境
    ReleaseDC (pEditControlDC);
    return (TRUE);
   }

// 获取焦点响应函数
void CEditFlat::OnSetFocus(CWnd* pOldWnd)
   {
    CEdit::OnSetFocus(pOldWnd);
    m_bIsFocused = TRUE;
    m_bMouseOnEdit = TRUE;
    Invalidate ();
   }

// 失去焦点响应函数
void CEditFlat::OnKillFocus(CWnd* pNewWnd)
   {
```

```
    CEdit::OnKillFocus(pNewWnd);
    m_bIsFocused = FALSE;
    m_bMouseOnEdit = FALSE;
    Invalidate();
   }

// 获取系统颜色函数
BOOL CEditFlat::GetSysColors(VOID)
   {
    // 设置系统颜色
    m_clrHiLite = GetSysColor(COLOR_BTNHIGHLIGHT);
    m_clrShadow = GetSysColor(COLOR_BTNSHADOW);
    m_clrDarkShadow = GetSysColor(COLOR_3DDKSHADOW);
    m_clrButton = GetSysColor(COLOR_BTNFACE);
    return (TRUE);
   }

// 系统颜色改变消息响应函数
void CEditFlat::OnSysColorChange()
   {
    // 系统颜色改变，需要改变Edit颜色
    CEdit::OnSysColorChange();
    GetSysColors();
   }

void CEditFlat::PreSubclassWindow()
   {
    DWORD dwEditStyle;
    dwEditStyle = GetStyle();
    DWORD dwEditStyleEx;
    dwEditStyleEx = GetExStyle();
    if (dwEditStyle & WS_VSCROLL)
       {
        // 有垂直滚动条
        m_iVerticalScrollWidthRight=(BYTE)::GetSystemMetrics
           (SM_CXVSCROLL);
        if (dwEditStyleEx & WS_EX_LEFTSCROLLBAR)
           {
            m_iVerticalScrollWidthLeft = m_iVerticalScrollWidthRight;
            m_iVerticalScrollWidthRight = NULL;
           }
       }
    if (dwEditStyle & WS_HSCROLL)
       {
```

```
            m_iHorizontalScrollWidth = (BYTE) ::GetSystemMetrics
                (SM_CYHSCROLL);
            }
        m_bHorizontalFrameWidth = (BYTE)::GetSystemMetrics (SM_CYEDGE);
        m_bVerticalFrameWidth = (BYTE)::GetSystemMetrics (SM_CXEDGE);
        CEdit::PreSubclassWindow();
        }

// 形状改变响应函数
void CEditFlat::OnStyleChanging( int nStyleType,LPSTYLESTRUCT
    lpStyleStruct)
    {
     if (nStyleType == GWL_EXSTYLE)
        {
         if (lpStyleStruct->styleNew & WS_EX_LEFTSCROLLBAR)
            {
             m_iVerticalScrollWidthLeft= m_iVerticalScrollWidthRight;
             m_iVerticalScrollWidthRight = NULL;
            }
         else if (lpStyleStruct->styleOld & WS_EX_LEFTSCROLLBAR)
            {
             m_iVerticalScrollWidthRight= m_iVerticalScrollWidthLeft;
             m_iVerticalScrollWidthLeft = NULL;
            }
        }
     // 调用父类的响应处理函数
     CEdit::OnStyleChanged (nStyleType, lpStyleStruct);
    }

// 垂直滚动函数
void CEditFlat::OnVScroll(UINT nSBCode,UINT nPos, CScrollBar* pScrollBar)
    {
     if (m_bVScrollPressed)
        {
         m_bVScrollPressed = FALSE;
        }
     else
        {
         m_bVScrollPressed = TRUE;
        }
     CEdit::OnVScroll(nSBCode, nPos, pScrollBar);
    }

// 水平滚动函数
```

```
void CEditFlat::OnHScroll(UINT nSBCode,UINT nPos, CScrollBar* pScrollBar)
    {
     if (m_bHScrollPressed)
        {
         m_bHScrollPressed = FALSE;
        }
     else
        {
         m_bHScrollPressed = TRUE;
        }
     CEdit::OnHScroll(nSBCode, nPos, pScrollBar);
    }
```

〖总结〗

本例给读者提供了一个具有平铺风格的编辑控件，它能很好的改善应用程序的界面。它的实现主要是通过引入了一个新类CEditFlat，该类继承了CEdit的功能，并在此基础上进行了一番扩展。利用CEditFlat::DrawEdit函数，达到我们想要的效果。希望读者能够以此作为基点，再进行扩展，得到一个更加适于自己应用程序界面的编辑控件。

案例1.2 椭圆型按钮

〖案例说明〗

按钮是一种非常通用的控件，它具有很多用途，但主要还是接收来自用户的命令或响应。Visual C++给我们提供了多种按钮控件，利用它们可以完成许多功能。但是Visual C++提供的均是方型的按钮，有没有办法让按钮换一个面孔呢？有。本节将通过一个小程序来实现椭圆型的按钮，使界面看起来更吸引人。图1.3所示是本例程序运行的结果，看起来是不是很酷。

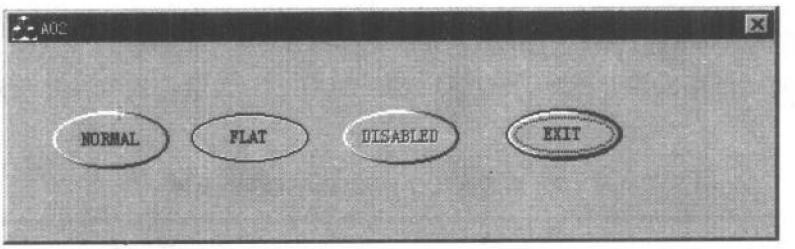

图 1.3 椭圆型按钮

〖设计思想〗

MFC给我们提供了CButton类，该类提供了标准Windows按钮的所有功能。在本例中，