万水课程设计丛书

Visual C++ 课程设计案例精编

严华峰　等编著

中国水利水电出版社
www.waterpub.com.cn

万水课程设计丛书

Visual C++课程设计案例精编

严华峰　等编著

中国水利水电出版社

内 容 提 要

Visual C++是 Microsoft 公司出品的可视化开发工具。本书以翔实的内容、精选的案例全面介绍了如何利用 Visual C++进行课程设计和软件制作。

本书介绍了 11 个课程设计案例，各个案例基本独立，覆盖了 Windows 下编程的大部分内容，包括图形和图像、多媒体、Internet 网络、数据库、程序通信、ActiveX 技术等各个方面，深入浅出地说明了 Visual C++最典型的和最有用途的程序设计方法，其中很多内容是一般介绍 Visual C++基础编程的书籍没有涉及到的。本书应用性极强，案例全部都可以运行，读者可以根据这些案例进行研究、修改和扩展，使其符合自己的要求。总之，这是本非常有用的实用案例丛书。

本书是 VC 爱好者学习 VC 编程课程设计的好帮手和课程资料。同时还是广大教师、计算机专业编程人员的学习参考书。

图书在版编目（CIP）数据

Visual C++课程设计案例精编/严华峰等编著. —北京：中国水利水电出版社，2002

（万水课程设计丛书）

ISBN 7-5084-1004-1

Ⅰ. V… Ⅱ. 严… Ⅲ. C 语言－程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2002）第 019420 号

书　　名	Visual C++课程设计案例精编
作　　者	严华峰　等编著
出版、发行	中国水利水电出版社（北京市三里河路 6 号　100044） 网址：www.waterpub.com.cn E-mail：mchannel@public3.bta.net.cn（万水） sale@waterpub.com.cn 电话：（010）68359286（万水）、63202266（总机）、68331835（发行部）
经　　售	全国各地新华书店
排　　版	北京万水电子信息有限公司
印　　刷	北京市天竺颖华印刷厂
规　　格	787×1092 毫米　16 开本　20.25 印张　441 千字
版　　次	2002 年 4 月第一版　2002 年 4 月北京第一次印刷
印　　数	0001—5000 册
定　　价	35.00 元（含 1CD）

凡购买我社图书，如有缺页、倒页、脱页的，本社发行部负责调换

前　言

Microsoft 出品的 Visual C++是一个功能非常强大的可视化编程工具，可以说是目前功能最为强大的程序开发平台之一。人们常说“真正的程序员用 C++”，但是掌握一门计算机语言并不是一件容易的事情，需要一个实践的过程，而从案例开始学习不失为一个好办法。

本书的目的是为了帮助广大 Visual C++的读者快速熟悉和掌握 Microsoft Visual C++软件，使具有不同编程背景的人更加精通 C++的 Windows 编程技术。

本书的主要特色是采用了编程案例的形式来编写，利用大量生动有趣的编程案例向读者介绍可视化编程的技术和软件开发的思维方式，并使读者能够从中领悟到一些编程技巧，而且读者还可以根据这些案例进行研究、修改和扩展，使其符合自己的要求。书中提供的所有案例都经过作者编译通过，完整无误，通过每一个案例的学习，读者可以轻松掌握有关 VC 编程案例的设计和完成。

本书的各个案例基本上互相独立，全书覆盖了包括图形和图像、多媒体、Internet 网络、数据库、程序通信、ActiveX 技术等，深入浅出地说明了 VC 中最具典型性和最有用的程序设计方法。读者可以根据自己实际情况选择不同章节进行阅读，相信丰富的案例讲解可以解除读者阅读枯燥之苦。

作者系 VC 编程爱好者，也是微软 Visual Studio 系列的忠实用户，愿与广大读者交流其应用的技术、技巧，联系方式：Howard@china.com

参加本书编写的人员还有童剑、王琪、李炳照、白雪、李萍、魏云芳、张丽华、赵鸿斌、朱海峰、张磊、李欣欣、王丽霞、张强、要黎宝等，在此一并表示感谢。

作者

2002 年 2 月

目　　录

案例一

指针式时钟

1.1　案例功能说明

在应用程序中，经常有一些任务需要在后台处理，实现方式有两种：计时器和 OnIdle 循环处理。本案例将使用计时器创建一个时钟。

• 计时器是程序中最常用的后台任务机制之一，其时间间隔最低约 55 毫秒，被广泛用于时钟、磁盘备份程序或需要在某一时刻运行的程序等。

• 多媒体计时器能编程设定 1 毫秒或者更小，它是诸如 MIDI 序列发生器之类的专用型应用程序的理想选择，在 Windows API 中有很多查询时钟的函数，利用它们就可以编写出高精度的计时器。

如图 1.1 所示为本例程序的运行界面。

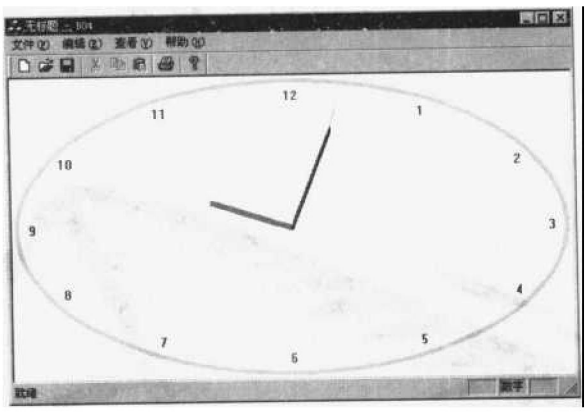

图 1.1　程序运行结果

1.2　程序设计思想

使用计时器只需了解两个函数：CWnd::SetTimer()函数用来设置一个计时器以指定的时间间隔触发，CWnd::KillTimer()函数用来使一个正在运行的计时器停止。

计时器以两种方式通知应用程序计时器间隔时间已到：发送 WM_TIMER 消息和调用应用程序定义的回调函数。其中前者相对比较简单，但对于多个计时器则应使用回调函数。计时器消息发送给应用程序时都是低优先级，因此只有当消息对列中没有其他消息时才会处理它们。

计时器消息不能在消息队列中积存，这避免了出现永远无法清空消息队列的状态。尽管如此，Windows 应用程序决不应该花费过量的时间来处理消息，除非该处理过程已经委派给辅助线程。如果主线程运行时间过长而没有检查消息队列，则程序的响应能力会受到影响。

1.3　程序设计框架和步骤

（1）用 MFC AppWizard（exe）创建一个新工程，将其命名为 B04。

（2）在 MFC AppWizard 第一步中设置应用程序的类型为单个文档模式，然后单击“确定”按钮，于是应用程序 B04 就创建完毕。

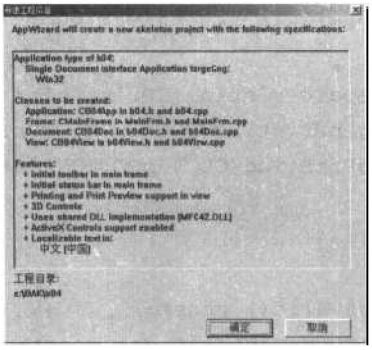

图 1.2　程序工程配置

（3）打开 MFC ClassWizard ，为 B04View 类添加 OnCreate 和 Ontimer 函数。

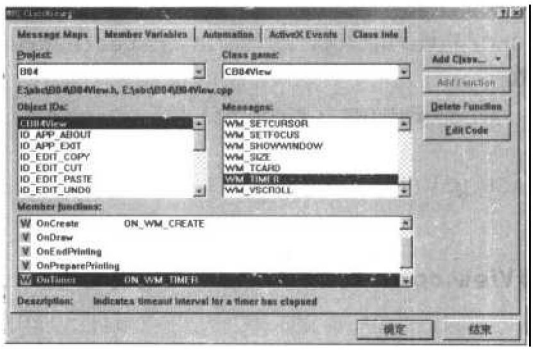

图 1.3　用 ClassWizard 添加消息响应函数

（4）编辑 OnDraw 方法的代码，实现绘制时钟；编辑 OnTimer 函数的代码，用来更新窗口；编辑 OnCreate 方法，用来设置时钟。

1.4　程序代码分析

1.4.1　头文件 B04View.h：定义类 CB04View 的接口，完成变量初始化

```
class CB04View : public CView
{
```

```
protected:
    CB04View();
    DECLARE_DYNCREATE(CB04View)
public:
    CB04Doc* GetDocument();
public:
    public:
    virtual void OnDraw(CDC* pDC);
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    protected:
    virtual BOOL OnPreparePrinting(CPrintInfo* pInfo);
    virtual void OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo);
    virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo);
public:
    virtual ~CB04View();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
protected:
protected:
//创建视图消息处理函数
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
//计时器消息处理函数
    afx_msg void OnTimer(UINT nIDEvent);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
```

1.4.2 源文件 B04View.cpp，类 CB04View 的实现文件

1. 消息映射定义

```
BEGIN_MESSAGE_MAP(CB04View, CView)
    //{{AFX_MSG_MAP(CB04View)
    ON_WM_CREATE()
    ON_WM_TIMER()
    //}}AFX_MSG_MAP
    //标准输出命令
    ON_COMMAND(ID_FILE_PRINT, CView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_DIRECT, CView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_PREVIEW, CView::OnFilePrintPreview)
END_MESSAGE_MAP()
```

2. CB04View 构造函数和析构函数

```
CB04View::CB04View()
{
}
CB04View::~CB04View()
{
}
```

3. 客户区重绘函数，用来绘制时钟

```
void CB04View::OnDraw(CDC* pDC)
{
    CB04Doc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    //获取客户区
    RECT Rect;
    GetClientRect(&Rect);
    //计算椭圆中心位置
    int CenterX = Rect.right/2;
    int CenterY = Rect.bottom/2;
    //取当前时间
    CTime Time = CTime::GetCurrentTime();
    CString str;
    int i,x,y;
    CSize size;
    //创建一支黄色的笔，用来画椭圆
    CPen Pen(PS_SOLID,5,RGB(255,255,0));
    //设置当前画笔，并记下以前的画笔
    CPen *OldPen = pDC->SelectObject(&Pen);
    //绘制钟面椭圆
    pDC->Ellipse(5,5,Rect.right-5,Rect.bottom-5);
    double Radians;
    //设置字体颜色为红色
    pDC->SetTextColor(RGB(255,0,0));
    for(i = 1;i <= 12;i++){
        //格式化钟点值
        str.Format("%d",i);
        size = pDC->GetTextExtent(str,str.GetLength());
        Radians = (double)i*6.28/12.0;
        //计算钟点放置的位置
        x = CenterX - (size.cx/2) + (int)((double)(CenterX - 20)*
            sin(Radians));
        y = CenterY - (size.cy/2) - (int)((double)(CenterY - 20)*
            cos(Radians));
        //绘制钟点
```

```
            pDC->TextOut(x,y,str);
        }
        //计算时钟指针的夹角
        Radians = (double)Time.GetHour() + (double)Time.GetMinute()/60.0 +
            (double)Time.GetSecond()/3600.0;
        Radians *= 6.28/12.0;
        //创建时钟指针画笔
        CPen HourPen(PS_SOLID,5,RGB(0,255,0));
        pDC->SelectObject(&HourPen);
        //绘制时钟指针
        pDC->MoveTo(CenterX,CenterY);
        pDC->LineTo(CenterX + (int)((double)(CenterX/3)*sin(Radians)),
            CenterY - (int)((double)(CenterY/3)*cos(Radians)));
        Radians = (double)Time.GetMinute()+(double)Time.GetSecond()/60.0;
        Radians *= 6.28/60.0;
        //创建分钟指针画笔
        CPen MinutePen(PS_SOLID,3,RGB(0,0,255));
        pDC->SelectObject(&MinutePen);
        //绘制分钟指针
        pDC->MoveTo(CenterX,CenterY);
        pDC->LineTo(CenterX + (int)((double)(CenterX*2)/3)*sin(Radians),
            CenterY - (int)((double)(CenterY*2/3)*cos(Radians)));
        Radians = (double)Time.GetSecond();
        Radians *= 6.28/60.0;
        //创建秒钟指针画笔
        CPen SecondPen(PS_SOLID,1,RGB(0,255,255));
        pDC->SelectObject(&SecondPen);
        //绘制秒钟指针
        pDC->MoveTo(CenterX,CenterY);
        pDC->LineTo(CenterX + (int)((double)(CenterX*4)/5)*sin(Radians),
            CenterY - (int)((double)(CenterY*4)/5*cos(Radians)));
        pDC->SelectObject(OldPen);
}
```

4. 视图创建消息处理函数，初始化计时器

```
int CB04View::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CView::OnCreate(lpCreateStruct) == -1)
        return -1;
    //设置计时器，1 秒发送一次消息
    SetTimer(1,1000,NULL);
    return 0;
}
```

5. 计时器消息处理函数

```
void CB04View::OnTimer(UINT nIDEvent)
{
    InvalidateRect(NULL,true);
    UpdateWindow();
    CView::OnTimer(nIDEvent);
}
```

1.5 总结

需要注意的是，计时器消息不是和其他消息异步处理的，即计时器不会中断另一个非计时器消息。另外，计时器编程所设定的时间间隔并不是精确值，因为计时器所基于的硬件计时器每 54.9 秒“滴答”一次，Windows 将把计时值舍入到下一个 55 毫秒的倍数。例如，若设置的计时器间隔为 60 毫秒，则计时器消息将是每 110 毫秒到达一次。

案例二

屏幕保护程序