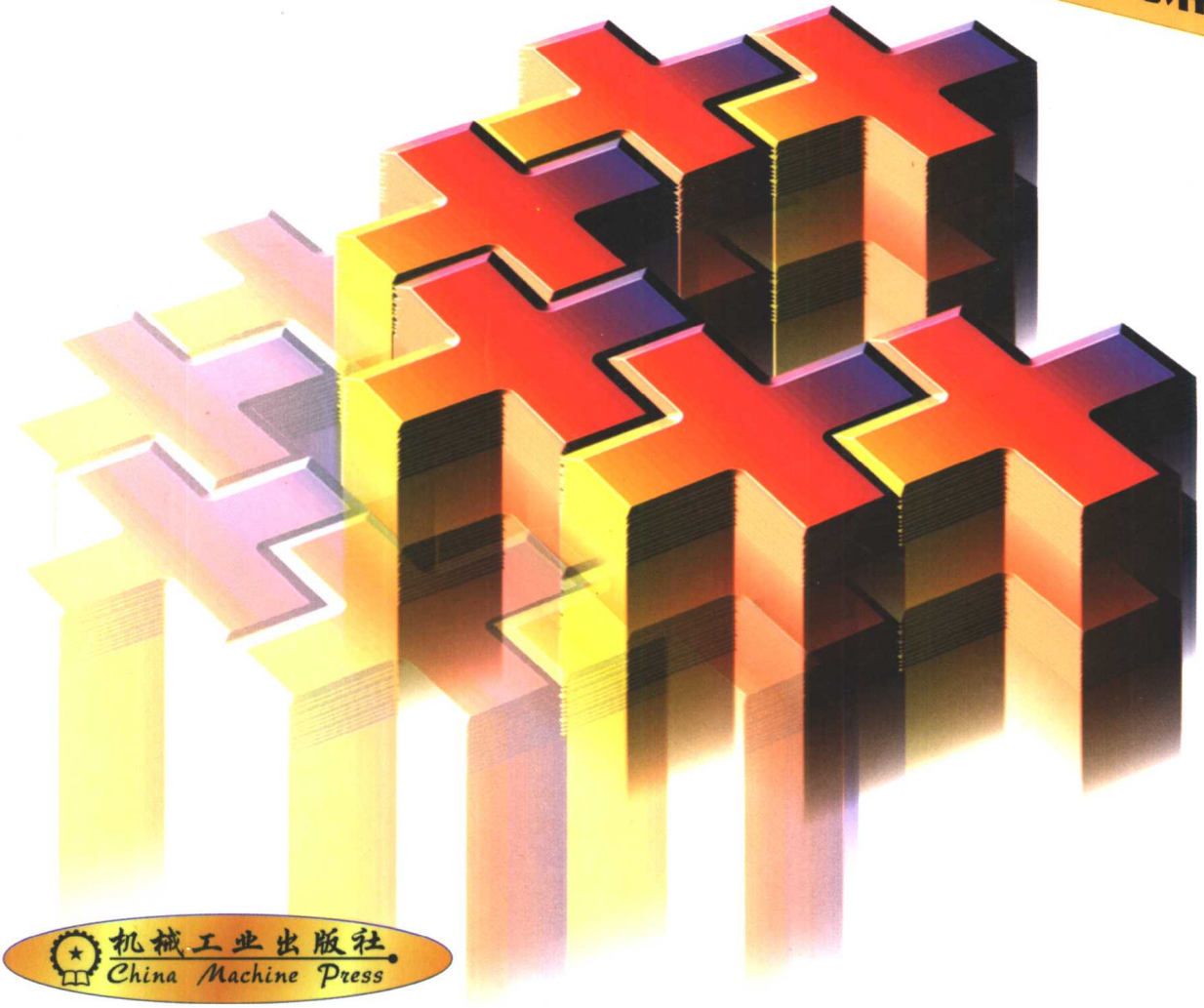
计算机编程技巧

ABC 系列丛书

Visual C++ 编程技巧

多媒体与系统篇



清宏计算机工作室 编著

机械工业出版社
China Machine Press

计算机编程技巧 ABC 系列丛书

Visual C++ 编程技巧

（多媒体与系统篇）

清宏计算机工作室　编著

机 械 工 业 出 版 社

Visual C++是基于Windows操作平台的功能强大的编程开发环境。本书通过上百个实例全面讲述和分析了应用 MFC 进行 Visual C++ 编程的思想，对 Visual C++在编程实际应用方面的基本使用方法和一些高级编程技术要点作了详尽的讲解。

本书包含的大量的实例，便于读者掌握，适用于有一定的 Visual C++ 编程基础的计算机爱好者阅读。

机械工业出版社（北京市百万庄大街 22 号　邮政编码 100037）
责任编辑：边　萌　封面设计：姚　毅
责任印制：郭景龙
北京京丰印刷厂印刷·新华书店北京发行所发行
2001 年 5 月第 1 版第 2 次印刷
787mm×1092mm 1/16·23.25 印张·571 千字
5 001—7 000 册
定价：41.00 元（1CD，含配套书）
ISBN 7-900043-99-3/TP·95
凡购本书，如有缺页、倒页、脱页，由本社发行部调换
本社购书热线电话（010）68993821、68326677-2527

前　言

随着计算机技术的飞速发展，计算机的开发迅速得到普及推广。当前有一大批计算机开发人员，虽然已经掌握了某种开发工具的基础知识、甚至达到了相当高的水平，但仍有很多盲区，或者急需更多的编程经验和技巧。为此，我们决定组织编写这套“计算机编程技巧ABC系列丛书”，以满足广大读者的需求，帮助读者快速成为计算机编程得行家里手。

这套丛书针对当前最流行的几种开发语言，介绍开发过程中的编程经验和技巧，并解决开发过程中的疑点和难点，主要突出“技巧”二字。针对某些开发语言，根据其内容相关性和独立性，我们编写了两个系列的书，即“网络与数据库篇”和“多媒体与系统篇”。这样可以使读者有更大的选择余地，并且在内容编排、读者阅读上都更为合理。

每本书按章节编写，各章由既相对独立，又相互关联的技巧主题组成，并且在顺序编排上考虑了内容的前后连续性。

针对每一个技巧主题，我们将其分为“ABC”三个部分进行讲解。

A——关键所在（或原理方法）　以简单明了的语言指出实现技巧的关键所在或原理方法。

B——实现与应用　以一个简单、完整实例向读者介绍技巧的实现方法或步骤。

C——专家点评　对技巧加以适当说明，包括补充解释，强调注意事项等。比如简要介绍实现该技巧的其他方法，或者介绍关键技术的扩展应用等。

针对每一个技巧主题，力求在内容讲解清楚的前提下，篇幅短小，使读者在有限篇幅内学到更多的技巧，整套书充分体现强调短小精悍、高效率的特点。

读者在阅读本套丛书之前，应该对相应开发语言有了一定了解，但并不要求读者开发水平一定很高，在章节编排和内容讲解过程中，我们都尽量照顾了水平还比较低的读者，因此初学者还是可以阅读这套丛书的。

希望这套丛书能使你很快成为软件开发的高手。

恳请广大读者对书中不足之处提出宝贵建议。

清宏计算机工作室

编 者 的 话

本书是为了让读者能够掌握 Visual C++ 常用技巧而编写的。

本书是一本有关在 Visual C++ 6.0 环境下开发应用程序的图书，书中通过一系列的实例向读者介绍如何使用 Visual C++ 6.0 的强大功能来开发各种应用程序，并向读者展示了应用 Visual C++ 6.0 进行程序开发的基本方法和相关技巧。实际上，当学习一个新的编程课题时，研究一个相关的例子可以在较短的时间内获得较好的效果。

由于这是一本介绍 Visual C++ 6.0 应用开发的书，所以书中没有从最基本的一般知识讲起，而是假定读者已经学习过 C 语言和 C++语言，有过一些 Windows 编程经验，并对 MFC 有一定的认识的计算机爱好者。

关于光盘：

本书的配套光盘中含有此书中的全部源程序，这些程序都是经过严格的调试和测试的。读者可以拷贝到硬盘上打开工程，运行程序。

如果读者运行程序时发生问题，可以给作者发信，作者会尽快帮您解决问题。

作者联系方式：

jxllc@263.net

由于编者的时间关系和水平有限，书中难免出错，希望读者谅解并批评指正。

此为试读，需要完整PDF请访问：www.ertongbook.com

目 录

第 1 章　扩展标准控件功能的技巧

1.1　实现 Cool 型浮动按钮

关键所在

从 CButton 类派生出新的按钮类，重载 DrawItem 函数，并处理适当的 Windows 消息，即可实现 Cool 型按钮。当自绘按钮的可视属性发生变化时，框架将调用 DrawItem 函数，其原型如下所示：

virtual void DrawItem(LPDRAWITEMSTRUCT lpDIS)

DrawItem 函数用到了结构 DRAWITEMSTRUCT，它提供了绘制控件所需要的信息，其定义如下：

```
typedef struct tagDRAWITEMSTRUCT {
    UINT     CtlType;
    UINT     CtlID;
    UINT     itemID;
    UINT     itemAction;
    UINT     itemState;
    HWND     hwndItem;
    HDC      hDC;
    RECT     rcItem;
    DWORD    itemData;
} DRAWITEMSTRUCT, *LPDRAWITEMSTRUCT
```

对于按钮来说，最重要的成员变量有两个：一个是 hDC，另一个是 itemState。hDC 提供了绘制控件时所要用到的设备文本。itemState 则标识控件将要表现的可视状态，其取值如下：

- ODS_CHECKED　仅对菜单使用，当要标记菜单项时设置该位。
- ODS_DISABLED　绘制项处于禁止状态时设置该位。
- ODS_FOCUS　绘制项拥有输入焦点时设置该位。
- ODS_GRAYED　仅对菜单使用，当菜单项变灰时设置该位。
- ODS_SELECTED　绘制项被选中时设置该位。
- ODS_COMBOBOXEDIT　绘图发生在自画组合框控件中的编辑控件。
- ODS_DEFAULT　该项为默认项。

实现与应用

下面的例子将实现一个从 CButton 派生出来的 Cool 型按钮类 CCoolButton，并提供如下功能：

- 标准 CButton 属性。
- 可以显示文本、图像的组合或两者之一。
- 图像可在左或在右。
- 图像来源于图标资源、位图资源或位图文件。
- 根据图像的大小和文本的长度更改按钮客户区的尺寸。
- 标准、半浮动或浮动风格。
- 具有提示并可更改。

实现步骤如下：

（1）新建一个基于对话框的项目。利用 ClassWizard 或 ClassView 向项目中添加一个派生于 CButton 的新类 CCoolButton。重载 CCoolButton 类的 DrawItem 函数。首先获取按钮的客户区尺寸，用背景色填充客户区；然后根据按钮的边框类型（浮动，半浮动，3D）和当前状态（鼠标是否在按钮上面，是否被按下，是否获得输入焦点）来绘制按钮的边框，体现其立体效果；最后，调用 OnDraw 函数来实现文本和图像的绘制。

```
void CCoolButton::DrawItem(LPDRAWITEMSTRUCT lpDIS)
{
    ASSERT (lpDIS != NULL);
    ASSERT (lpDIS->CtlType == ODT_BUTTON);
    CDC* pDC = CDC::FromHandle (lpDIS->hDC);
    ASSERT_VALID (pDC);
    CRect rectClient = lpDIS->rcItem;
    // 填充按钮客户区
    COLORREF clrBtnFace = ::GetSysColor(COLOR_BTNFACE);
    CBrush brBtnFace;
    brBtnFace.CreateSolidBrush (clrBtnFace);
    pDC->FillRect (rectClient, &brBtnFace);

    if (lpDIS->itemState & ODS_FOCUS)
        OnDrawFocusRect (pDC, rectClient);
    // 绘制边框
    COLORREF clrBtnShadow,clrBtnHilite,clrBtnDkShadow,clrBtnLight;
    clrBtnShadow = ::GetSysColor(COLOR_BTNSHADOW);
    clrBtnDkShadow = ::GetSysColor(COLOR_3DDKSHADOW);
    clrBtnLight = ::GetSysColor(COLOR_3DLIGHT);
    clrBtnHilite = ::GetSysColor(COLOR_BTNHIGHLIGHT);
```

```
    if (m_nFlatStyle != BUTTONSTYLE_NOBORDERS)
    {
        if(m_bPushed&& m_bHighlighted || (lpDIS->itemState & ODS_SELECTED))
        {
            pDC->Draw3dRect (rectClient,clrBtnDkShadow,clrBtnHilite);
            rectClient.DeflateRect (1, 1);
            if (m_nFlatStyle != BUTTONSTYLE_FLAT)
                pDC->Draw3dRect (rectClient,clrBtnShadow,clrBtnLight);
            rectClient.DeflateRect (1, 1);
            rectClient.left += m_sizePushOffset.cx;
            rectClient.top += m_sizePushOffset.cy;
        }
        else if (m_nFlatStyle != BUTTONSTYLE_FLAT || m_bHighlighted)
        {
            pDC->Draw3dRect (rectClient,clrBtnHilite,clrBtnDkShadow);
            rectClient.DeflateRect (1, 1);
            if (m_nFlatStyle == BUTTONSTYLE_3D ||
                (m_nFlatStyle==BUTTONSTYLE_SEMIFLAT&&m_bHighlighted))
                pDC->Draw3dRect (rectClient,clrBtnLight,clrBtnShadow);
            rectClient.DeflateRect (1, 1);
        }
        else
            rectClient.DeflateRect (2, 2);
    }
    else
        rectClient.DeflateRect (2, 2);
    // 绘制按钮内容
    OnDraw (pDC, rectClient, lpDIS->itemState);
}
```

（2）利用 ClassView 为 CCoolButton 类添加 OnDraw 函数，实现文本和图像的绘制。根据按钮的外观类型、图像位置和使能状态来计算文本和图像的位置，并在相应的位置绘制按钮的图像和文字标题。

```
void CCoolButton::OnDraw (CDC* pDC, const CRect& rect, UINT uiState)
{
    COLORREF clrBtnText = ::GetSysColor(COLOR_BTNTEXT);
    COLORREF clrBtnHilite = ::GetSysColor(COLOR_BTNHIGHLIGHT);
    COLORREF clrGrayedText = ::GetSysColor (COLOR_GRAYTEXT);
    // 计算文本和图像的位置
    CRect rectText = rect;
```

此为试读，需要完整PDF请访问：www.ertongbook.com

```
CRect rectImage = rect;
CString strText;
GetWindowText (strText);
if (m_sizeImage.cx != 0 && m_nDisplayStyle != BUTTONSTYLE_TEXT)
{
    if (!strText.IsEmpty ()&& m_nDisplayStyle != BUTTONSTYLE_IMAGE)
    {
        if (m_bRighImage)
        {
            rectText.right -= m_sizeImage.cx + nImageHorzMargin / 2;
            rectImage.left = rectText.right;
            rectImage.right -= nImageHorzMargin / 2;
        }
        else
        {
            rectText.left +=   m_sizeImage.cx + nImageHorzMargin / 2;
            rectImage.left += nImageHorzMargin / 2;
            rectImage.right = rectText.left;
        }
    }
    // 使图像居中
    rectImage.DeflateRect ((rectImage.Width () - m_sizeImage.cx) / 2,
        max (0, (rect.Height () - m_sizeImage.cy) / 2));
}
else
    rectImage.SetRectEmpty ();
// 绘制文本
pDC->SetBkMode (TRANSPARENT);
if(m_nDisplayStyle != BUTTONSTYLE_IMAGE)
{
    COLORREF clrText=m_clrRegular==clrDefault?clrBtnText:m_clrRegular;
    if (m_bHighlighted && m_clrHover != clrDefault) clrText = m_clrHover;
    UINT uiDTFlags = DT_VCENTER | DT_SINGLELINE | DT_CENTER;
    if (uiState & ODS_DISABLED)
    {
        pDC->SetTextColor (clrBtnHilite);
        CRect rectShft = rectText;
        rectShft.OffsetRect (1, 1);
        pDC->DrawText (strText, rectShft, uiDTFlags);
```

```
            clrText = clrGrayedText;
        }
        pDC->SetTextColor (clrText);
        pDC->DrawText (strText, rectText, uiDTFlags);
    }
    // 绘制图像
    if (!rectImage.IsRectEmpty () && m_nDisplayStyle != BUTTONSTYLE_TEXT)
    {
        UINT uiFlags = (uiState & ODS_DISABLED) == 0 ?
DSS_NORMAL : DSS_DISABLED;
        if (m_hIcon != NULL)
        {
            ASSERT (m_hBitmap == NULL);
            HBRUSH hbr = NULL;
            pDC->DrawState (rectImage.TopLeft (), m_sizeImage,
                m_bHighlighted&&m_hIconHot != NULL?m_hIconHot : m_hIcon,
                uiFlags, hbr);
        }
        else
        {
            HBITMAP hbmp = m_hBitmap;
            if (uiState & ODS_DISABLED)
                hbmp = m_hBitmapDisabled;
            else if (m_bHighlighted && m_hBitmapHot != NULL)
                hbmp = m_hBitmapHot;
            ASSERT (hbmp != NULL);
            pDC->DrawState (rectImage.TopLeft (), m_sizeImage, hbmp, uiFlags);
        }
    }
}
```

（3）利用 ClassView 为 CCoolButton 类添加 SizeToContent 函数，根据按钮的外观类型和文本、图像的大小得到按钮的新客户区尺寸，当参数 bCalcOnly 为 TRUE 时，重置按钮客户区为新的尺寸，从而使按钮的大小与其所显示的内容相符合。

```
CSize CCoolButton::SizeToContent (BOOL bCalcOnly)
{
    ASSERT (GetSafeHwnd () != NULL);
    CClientDC dc (this);
    CString strText;
    GetWindowText (strText);
```

```
    CSize sizeText = dc.GetTextExtent (strText);
    int cx,cy;
    switch (m_nDisplayStyle)
    {
    case BUTTONSTYLE_IMAGE:
        cx = m_sizeImage.cx + nImageHorzMargin;
        cy = m_sizeImage.cy + nVertMargin * 2;
        break;
    case BUTTONSTYLE_TEXT:
        cx = sizeText.cx + nImageHorzMargin;
        cy = sizeText.cy + nVertMargin * 2;
        break;
    case BUTTONSTYLE_BOTH:
        cx = sizeText.cx + m_sizeImage.cx + nImageHorzMargin;
        if (sizeText.cx > 0)
            cx += nImageHorzMargin;
        cy = max (sizeText.cy, m_sizeImage.cy) + nVertMargin * 2;
        break;
    }
    if (!bCalcOnly)
        SetWindowPos (NULL, -1, -1, cx, cy,
            SWP_NOMOVE | SWP_NOACTIVATE | SWP_NOZORDER);
    return CSize (cx, cy);
}
```

（4）在 CCoolButton 类中添加 WM_MOUSEMOVE、WM_LBUTTONDOWN、WM_LBUTTONUP 三个消息处理函数，设置必要的标识位，具体实现方法见所附程序源码。其他辅助函数的实现也请参考所附程序源码。

（5）对 CCoolButton 进行测试，程序运行结果如图 1-1、1-2 所示。

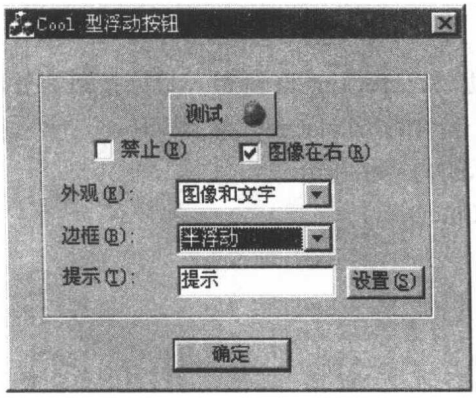


图 1-1 半浮动样式图像在右的 Cool 型浮动按钮

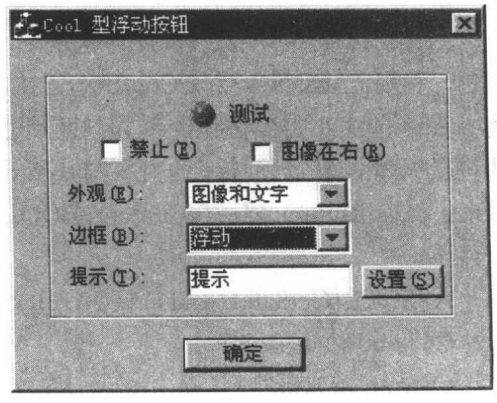


图 1-2 浮动样式的 Cool 型浮动按钮

专家点评

实现 CCoolButton 的关键之处在于 DrawItem 函数的实现和鼠标消息的处理。通过采用不同的绘制处理函数，可以设计出各种各样形状的按钮来。在本例中，按钮客户区尺寸的动态修改、禁止文本和图像的绘制以及工具提示的设置，则是对其功能的进一步完善和美化。

在 DrawItem 和 OnDraw 函数中，利用 GetSysColor 函数来获取系统的颜色设置，确保了按钮能与其使用环境相一致。

在 SizeToContent 函数中，利用 SetWindowPos 函数重置了按钮客户区的大小，其用法和各参数的含义请参见联机在线帮助或参见 2.8 节。

1.2　显示 3D 文本的按钮

关键所在

在对话框中使用显示 3D 文本的按钮可以突出显示该文本。与上例一样，实现 3D 文本按钮的步骤也是从 CButton 类派生出新的按钮类，重载其 DrawItem 函数，其关键所在是根据按钮客户区的大小来定制字体，并采用近似的颜色在不同的位置多次绘制文本，从而形成 3D 效果。

实现与应用

（1）新建一个基于对话框的程序，编辑主对话框资源使其如图 1-3 所示。

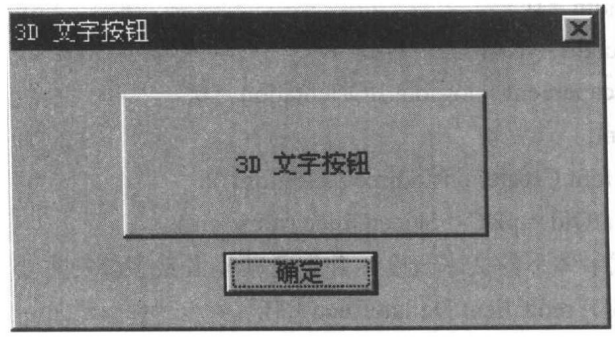


图 1-3　程序设计期间的主对话框界面

（2）利用 ClassWizard 或 ClassView 向工程中添加一个派生于 CButton 的新类 C3DtextBtn，重载其 DrawItem 函数如下所示：

```
void C3DTextBtn::DrawItem(LPDRAWITEMSTRUCT lpDrawItemStruct)
{
```

```
    CDC* pDC = CDC::FromHandle(lpDrawItemStruct->hDC);
    ASSERT_VALID(pDC);
    CRect rectClient = lpDrawItemStruct->rcItem;
    Draw(pDC,rectClient, lpDrawItemStruct->itemState);
}
```

（3）为类 C3DTextBtn 添加 OnDraw 函数。通过将 LOGFONT 的成员变量 lfWidth 设置为 0，使在调用 CFont 类的 CreateFontIndirect 函数直接从 LOGFONT 创建字体时，字体的宽度能被自动设置。先取得测试字体下的文本的大小，这是最终生成与按钮客户区相适应的字体的关键。最后，采用不同的颜色在不同的位置多次绘制文本，构成最终的 3D 效果的文本外观。

```
void C3DTextBtn::Draw(CDC* pDC, const CRect& rect, UINT state)
{
    CString text; GetWindowText(text);
    int l = text.GetLength();
    CRect rectClient = rect;

    // 获取控件字体
    CFont* pFont = GetFont();

    // 确保该字体具有正确的大小，并将其选入设备环境
    LOGFONT logfont;
    pFont->GetObject(sizeof(LOGFONT),&logfont);
    if (logfont.lfHeight == 0) logfont.lfHeight = 20;
    // 将 lfWidth 设置为 0，从而使其能够被自动计算
    logfont.lfWidth = 0;
    logfont.lfWeight = 1000;
    logfont.lfEscapement = logfont.lfOrientation = 0;
    CFont tryfont;
    VERIFY(tryfont.CreateFontIndirect(&logfont));
    CFont* pFontOld = pDC->SelectObject(&tryfont);
    // 取得测试字体下的文本长度，并据此精确调整字体的大小
    if (m_bUse3D) rectClient.DeflateRect(3,3);
    CSize textSizeClient = pDC->GetTextExtent(text,l);
    if (rectClient.Width()*textSizeClient.cy > rectClient.Height()*textSizeClient.cx)
        logfont.lfHeight = ::MulDiv(logfont.lfHeight,rectClient.Height(),textSizeClient.cy);
    else
        logfont.lfHeight = ::MulDiv(logfont.lfHeight,rectClient.Width(),textSizeClient.cx);

    logfont.lfHeight--;
```

```
if (m_bUse3D) rectClient.InflateRect(3,3);

CFont font;
font.CreateFontIndirect(&logfont);
pDC->SelectObject(&font);
textSizeClient = pDC->GetTextExtent(text,l);
int minx = rectClient.left+(rectClient.Width()-textSizeClient.cx)/2;
int miny = rectClient.top+(rectClient.Height()-textSizeClient.cy)/2;

int oldBkMode = pDC->SetBkMode(TRANSPARENT);
COLORREF textcol = ::GetSysColor((state & ODS_FOCUS) ? COLOR_GRAYTEXT :
                                 COLOR_BTNTEXT);
COLORREF oldTextColor = pDC->SetTextColor(textcol);
int cx = minx;
int cy = miny;
if (m_bUse3D)
{
        int s = (state & ODS_SELECTED) ? -1 : +1;
        cx += 3; cy += 3;

        // 绘制 3D 凸出文本
        pDC->SetTextColor(::GetSysColor(COLOR_3DDKSHADOW));
        pDC->TextOut(cx-s*2,cy+s*2,text);
        pDC->TextOut(cx+s*2,cy-s*2,text);
        pDC->TextOut(cx+s*2,cy+s*2,text);
        pDC->SetTextColor(::GetSysColor(COLOR_3DHILIGHT));
        pDC->TextOut(cx+s*1,cy-s*2,text);
        pDC->TextOut(cx-s*2,cy+s*1,text);
        pDC->TextOut(cx-s*2,cy-s*2,text);
        pDC->SetTextColor(::GetSysColor(COLOR_3DSHADOW));
        pDC->TextOut(cx-s*1,cy+s*1,text);
        pDC->TextOut(cx+s*1,cy-s*1,text);
        pDC->TextOut(cx+s*1,cy+s*1,text);
        pDC->SetTextColor(::GetSysColor(COLOR_3DLIGHT));
        pDC->TextOut(cx,cy-s*1,text);
        pDC->TextOut(cx-s*1,cy,text);
        pDC->TextOut(cx-s*1,cy-s*1,text);
        pDC->SetTextColor(textcol);
}
```

此为试读，需要完整PDF请访问：www.ertongbook.com