



Visual C++

高级编程技巧

计算机实用教程



◆ 高强 朱安国 主编



人民邮电出版社

编 者 的 话

Visual C++无疑是基于 Windows 环境应用系统的功能最强大的开发工具。介绍 Visual C++的书已有很多，而大多是 Visual C++使用说明的基本读物，有关的高级话题也只是泛泛而谈，并不实用。Windows 程序员不能体会到 Visual C++强大的开发功能，不能对开发过程中遇到的常见问题得到解决，不能利用 Visual C++开发出体现计算机发展新技术、新潮流的程序。

本书阐述了 Windows 程序员在开发应用系统过程中常碰到的高级或较新的技术问题：网络技术、多媒体编程、与数据库的连接、多种语言混合编程；同时也介绍了基本的、通用的技术问题：视窗技术、控件技术、消息机制等。本书覆盖了软件开发所涉及的很多领域，是集体智慧的结晶，是多年使用 Visual C++的软件开发人员实际工作经验的总结。希望本书能解决您实际工作中的问题，使您了解到计算机发展的潮流。本书的读者是程序员、大学生和研究生。由于不是 Visual C++和 Windows 编程的教科书，使用本书需要有 Windows 编程的基础知识和 Visual C++的基本使用常识。

本书共分五章，参加编写的人员有徐保民、孟昭睿、张全法、连卫民、金红、朱安国、朱德洪、王国余、于磊等。其中朱安国编写了第 2 章，孟昭睿编写了第 1 章，张全法编写了第 5 章，其他同志共同编写了本书的其它章节。

本书所附例程都在 Visual C++ 6.0 环境下调试过，读者可以直接使用或者稍作修改即可使用。对本书例程进行编译时最好在如下环境下进行：

CPU：奔腾 166 或更好

内存：32MB 或更大

硬盘：2.1GB 或更大

操作系统：Windows95/WindowsNT

SVGA 或更好的显示卡

符合 MPC 标准的声卡和音箱

本书覆盖面广、知识点新。由于作者水平有限，错误、浅陋和陈旧之处在所难免，恳请广大读者批评指正。

编者

一九九九年十一月

目 录

第 1 章 菜单、工具条和状态条设计技巧.....	1
1.1 菜单.....	1
1.1.1 如何激活变灰的弹出菜单	1
1.1.2 如何对两个菜单进行合并	2
1.1.3 如何确定顶层菜单所占据的菜单行数	6
1.1.4 如何给系统菜单添加一个菜单项	6
1.1.5 为什么即使调用 EnableMenuItem 菜单项后, 菜单项还处于禁止状态	8
1.1.6 如何在已存在的菜单中插入一子菜单项	8
1.1.7 如何实现一个使用图标的自画菜单	10
1.2 工具条.....	24
1.2.1 如何为工具条上的按钮增加一个下拉箭头	24
1.2.2 如何在工具栏上显示文本	26
1.2.3 如何实现平面工具条	29
1.2.4 如何使能和禁止工具条的工具提示	29
1.2.5 如何设置工具条标题	30
1.2.6 如何在代码中获取工具条的指针	31
1.2.7 如何编写类似于 Word97 的工具栏	31
1.3 状态条.....	39
1.3.1 如何在状态条中显示工具条或菜单的帮助信息	39
1.3.2 如何在状态条中增加一新的状态格	39
1.3.3 如何在状态格中显示文本信息	41
1.3.4 如何在状态条上加入一个时钟	42
1.3.5 如何在代码中获取状态条的指针	45
1.3.6 如何在状态条中加入 Bitmap 图	45
第 2 章 窗口、对话框与控件编程技巧.....	49
2.1 窗口.....	49
2.1.1 如何去掉窗口的框架和标题栏并使窗口充满全屏	49
2.1.2 如何防止用户改变窗口大小	50
2.1.3 如何改变窗口标题	52
2.1.4 如何改变窗口的图标	52
2.1.5 如何改变窗口的缺省风格	53

2.1.6	如何将窗口居中显示	53
2.1.7	如何让窗口一启动就最大化或最小化	54
2.1.8	如何让 MDI 窗口一启动就最大化或最小化	54
2.1.9	如何使窗口始终在最前方	55
2.1.10	如何防止主框窗口在其说明中显示活动的文档名	56
2.1.11	如何获取有关窗口正在处理的当前消息的信息	57
2.1.12	如何构造一多边形窗口	57
2.1.13	如何在应用程序窗口中加入动态分割线	58
2.1.14	如何检测一个窗口是否是分割窗口	59
2.1.15	如何在应用程序中使用树型控件和列表控件	60
2.2	对话框编程技巧	91
2.2.1	如何使对话框上的关闭按钮无效?	91
2.2.2	如何实现对话框的拖放	92
2.2.3	如何改变对话框的背景色	93
2.2.4	如何为对话框中的控件提供提示信息	94
2.2.5	如何改变对话框内控件的字体	95
2.2.6	如何在对话框内使用 CCheckBox 类	98
2.2.7	如何获取一个对话框控件的指针	99
2.2.8	如何在对话框内使用动画控件	100
2.3	控件	104
2.3.1	如何动态创建控件	104
2.3.2	如何改变控件的颜色	104
2.3.3	如何向编辑控件中添加文本	106
2.3.4	如何产生具有 3D 效果的文字	107
2.3.5	如何创建一个三态下压按钮	110
2.3.6	如何用位图显示下压按钮	111
2.3.7	如何生成圆形下压按钮	112
2.3.8	如何实现平面格式自画按钮	116
第 3 章	图形、图像与多媒体	127
3.1	图像部分	127
3.1.1	如何在窗口客户区显示一 BMP 格式的图像	127
3.1.2	如何利用位图来捕捉一个图像	133
3.1.2	如何进行图像的缩放	138
3.1.3	如何将图像按 BMP 格式保存在一个文件中	140
3.1.4	如何在程序运行时在图像上加入文字说明	143
3.1.5	如何将一个 BMP 图像拷贝到剪贴板中	145
3.2	图形部分	147
3.2.1	如何进行直线、任意线、矩形、圆等的绘制	147

3.2.2	如何实现图形元素的旋转	155
3.2.3	在图形绘制过程中如何判断直线图形元素是否选中	156
3.2.4	如何判定一任意线是否选中	157
3.2.5	如何判断图形元素——圆是否选中	160
3.2.6	如何判断矩形图元是否被选中	161
3.2.7	如何实现一个橡皮筋矩形	162
3.3	多媒体程序设计技巧	164
3.3.1	如何控制多媒体周边设备来编制自己的多媒体应用程序	164
3.3.2	如何编制录制波形音频的应用程序	185
3.3.3	如何播放波形音频	204
3.3.4	如何利用 Windows 的视频特性开发应用程序	215
第 4 章	数据库与网络编程技巧	231
4.1	数据库编程技巧	231
4.1.1	何谓 ODBC	231
4.1.2	ODBC 的体系结构	231
4.1.3	在 Visual C++ 中如何利用 ODBC 进行数据库编程	232
4.1.4	如何直接使用 ODBC API 进行数据库操作	233
4.1.5	如何通过 MFC 提供的 ODBC 数据库类进行数据库操作	241
4.1.6	如何在应用程序中实现表的动态连接	252
4.1.7	如何动态连接数据库	253
4.2	通信程序设计技巧	254
4.2.1	如何利用 Socket 进行通信程序设计	254
4.2.2	如何利用 CSocketFile 类和 Archive 类进行数据通信	257
4.2.3	如何利用 CSocket 的成员函数实现数据通信	258
第 5 章	其它 Windows 高级编程技巧	261
5.1	应用程序	261
5.1.1	如何获取应用程序的实例句柄	261
5.1.2	如何保证某一时刻只能运行应用程序的一个实例	261
5.1.3	如何保存和恢复应用程序的大小和位置	262
5.1.4	如何获取应用程序主窗口的句柄	263
5.1.5	如何获取其它应用程序的图标	263
5.1.6	如何结束应用程序的运行	264
5.1.7	怎样加载其它应用程序	264
5.1.8	如何获取应用程序的路径	266
5.1.9	如何使用自定义消息	266
5.2	系统	267
5.2.1	如何获取当前驱动器内磁盘的可用空间和内存的可用空间	267

5.2.2 如何检测 WINDOWS 版本.....	268
5.2.3 如何阻止窗口关闭	269
5.2.4 如何获得 Windows 和 Windows 系统目录.....	269
5.2.5 如何同时使用 VC5 和 VC6	270
5.2.6 如何获得 Windows 临时文件目录并创建临时文件	270
5.2.7 如何访问桌面窗口	271
5.2.8 如何获取系统显示元素的颜色	272
5.2.9 如何查询和设置系统参数	273
5.3 注册表编程.....	274
5.3.1 如何在应用程序中显示注册表的主次关键字及其值	274
5.3.2 在应用程序中如何获取 CPU 的有关信息	277
5.3.3 如何对系统注册表进行读写操作	278
5.4 混合编程.....	302
5.4.1 在 C 程序中如何混合使用汇编语言.....	302
5.4.2 如何实现 Java 与 C 的混合编程.....	306
5.5 其它.....	309
5.5.1 如何隐藏类视内的一个函数或一个变量	309
5.5.2 如何设置一全局变量, 以使文档中的所有类都能访问	310
5.5.3 如何才能建立一个等待光标	310
5.5.4 如何制作应用程序真彩色启动封面	310
5.5.5 如何快速装入一 BMP 文件到 Cbitmap 对象中	312
5.5.6 在程序运行时如何设置新的提示信息取代状态条上 “Ready” 提示信息	314
5.5.7 如何设置基于对话框的应用的初始位置	316
5.5.8 如何改变 MFC 的文档/视中打开/保存对话框中所显示的默认文件	317
5.5.9 如何使用钩子函数	322
5.5.10 如何实现不同进程间通信	339
5.6 调试技巧.....	346
5.6.1 如何减少 VC++ 编译时的链接时间	346
5.6.2 如何调试一个程序的发行版本	347
5.6.3 如何将调试信息输出到控制台窗口	347
附录 A Windows 编程的基本概念.....	349
A.1 与窗口有关的基本概念.....	349
A.1.1 窗口	349
A.1.2 桌面窗口	349
A.1.3 父窗口	349
A.1.4 子窗口	350
A.1.5 窗口名	350
A.1.6 窗口句柄.....	350

A.1.7 框架、客户以及子窗口	350
A.1.8 实例句柄	350
A.1.9 子窗口与父窗口的关系	351
A.1.10 禁止窗口	351
A.1.11 活动窗口	351
A.1.12 窗口可见性	351
A.1.13 窗口过程	351
A.1.14 窗口属性	351
A.2 有关消息的概念	352
A.2.1 消息	352
A.2.2 消息路由	352
A.2.3 投递和发送消息	353
A.2.4 消息种类	353
A.2.5 消息过滤	353
附录 B ODBC API 主要函数一览	355

第1章 菜单、工具条和状态条设计技巧

1.1 菜单

1.1.1 如何激活变灰的弹出菜单

【问题】

在应用程序中，为了限制用户的操作，经常需要将一个弹出的菜单项变灰，即使其处于非活动状态。但当条件得到满足后，则要将那些变灰的菜单项激活，在程序中如何实现呢？

【解决方法】

将变灰的菜单项激活，其实现方法是通过调用 `CMenu::EnableMenuItem` 函数来实现，该函数用法如下：

```
UINT EnableMenuItem ( UINT nIDEnableItem, UINT nEnable );
```

其中参数“`nIDEnableItem`”为菜单项的 ID 号，用于指定被操作的菜单项；参数“`nEnable`”为动作参数，用于说明要对被指定的菜单进行何种操作，其取值为 `MF_DISABLED`、`MF_ENABLED`、或 `MF_GRAYED` 与 `MF_BYCOMMAND` 或 `MF_BYPOSITION` 的“或运算”，其各取值常量的含义为：

- `MF_BYCOMMAND`：表示用命令 ID 拾取菜单项，为缺省值。
- `MF_BYPOSITION`：在当前菜单中用基于零的偏移量来拾取菜单项。
- `MF_DISABLED`：用户对菜单不可用。
- `MF_ENABLED`：用户对菜单可用。
- `MF_GRAYED`：用户对菜单不可用，并且菜单以灰色显示。

从上面的介绍可以知道，只要在调用 `EnableMenuItem` 函数时，将其参数“`nEnable`”的值设置为 `MF_BYCOMMAND|MF_ENABLED`，即可将所指定的菜单项激活。

【实现程序】

```
void CMyView::OnRButtonDown(UINT nFlags, CPoint point)
{
    CScrollView::OnRButtonDown(nFlags, point);

    CMenu *menu, *popup;
    menu = new CMenu();
```

```

// 装入菜单资源
menu->LoadMenu( IDR_POPUPMENU );
popup = menu->GetSubMenu(0);

UINT nEnable;
nEnable = MF_BYCOMMAND|MF_GRAYED;

if( your test ) {
    nEnable = MF_BYCOMMAND|MF_ENABLED;
}

popup->EnableMenuItem(ID_YOUR_ID, nEnable );

//显示菜单
ClientToScreen(&point);
popup->TrackPopupMenu(TPM_LEFTALIGN | TPM_RIGHTBUTTON,
                    point.x, point.y, this );

delete menu;
}

```

1.1.2 如何对两个菜单进行合并

【问 题】

由于应用程序运行环境的变化，有时需要将两个或多个菜单合并成一个菜单，以使应用程序能更好地适应当前的运行环境。那么在程序中如何实行这些工作呢？

【解决方法】

在 MFC 的 Cmenu 类中，给用户提供这样一个函数 Cmenu::AppendMenu，通过这个函数可以在当前菜单中加入新的菜单项。同时在该类中也提供了得到菜单项信息的诸函数，例如得到菜单项字符串信息的 Cmenu::GetMenuString 函数、得到菜单项状态的 Cmenu::GetMenuState 函数，以及得到菜单项子菜单的 Cmenu::GetSubMenu 函数和取消菜单的 Cmenu::Detach 函数等。通过综合利用这些函数，用户便可在应用程序中实现两个菜单项的合并工作。由于涉及的函数很多，这里只将 Cmenu::AppendMenu 的用法作个介绍，其用法如下：

```

BOOL AppendMenu( UINT nFlags,
                UINT nIDNewItem = 0,
                LPCTSTR lpszNewItem = NULL
            );

```

或

```

BOOL AppendMenu( UINT nFlags,

```

```
UINT nIDNewItem,
const Cbitmap* pBmp
```

```
);
```

其中参数“nFlags”用于说明新菜单的状态信息，其取值为下列值的组合：

- MF_CHECKED：显示复选标志。
- MF_UNCHECKED：不显示复选标志。
- MF_DISABLED：菜单对用户来说不可用。
- MF_ENABLED：菜单对用户来说可用。
- MF_GRAYED：菜单对用户来说不可用，且它以灰色显示。
- MF_MENUBARBREAK：除了在弹出式菜单中画一条线来分割外，与 MF_MENUBREAK 完全一致。
- MF_MENUBREAK：在带该标识的菜单项中，使菜单改变方向。
- MF_OWNERDRAW：依靠应用程序来绘制菜单内容。
- MF_POPUP：弹出下拉式菜单。
- MF_SEPARATOR：用于在视角上分开各组菜单项的占半个高度的直线。
- MF_STRING：带字符串的菜单项或弹出式菜单。

注意：

下面是四组不能同时用在一起的取值组合：

- MF_DISABLED、MF_ENABLED 和 MF_GRAYED；
- MF_STRING、MF_OWNERDRAW、MF_SEPARATOR 和图像菜单；
- MF_MENUBARBREAK 和 MF_MENUBREAK；
- MF_CHECKED 和 MF_UNCHECKED；

参数“nIDNewItem”用于指定新菜单命令 ID，如果参数“nFlags”取值为 MF_POPUP 时则用于指定弹出菜单的句柄。当参数“nFlags”取值为 MF_SEPARATOR，则参数“nIDNewItem”可以忽略。

参数“lpszNewItem”用于指定新菜单的内容。

参数“pBmp”为用于菜单项的 Cbitmap 对象的指针。

【实现程序】

```
void MergeMenu(Cmenu* pMenuDestination, Cmenu* pMenuAdd, bool bTopLevel
/*=false*/)
{
    int iMenuAddItemCount = pMenuAdd->GetMenuItemCount();
    int iMenuDestItemCount = pMenuDestination->GetMenuItemCount();

    // 若无菜单项则直接返回
    if( iMenuAddItemCount == 0 )
        return;
```

```
// 如果该菜单不是顶层菜单，而且目标菜单不空的情况下，
//加入一个分割菜单项
if( !bTopLevel && iMenuDestItemCount > 0 )
    pMenuDestination->AppendMenu( MF_SEPARATOR );

for( int iLoop = 0; iLoop < iMenuAddItemCount; iLoop++ )
{
    // 获取要加入的菜单的菜单字符串
    CString sMenuAddString;
    pMenuAdd->GetMenuString( iLoop, sMenuAddString, MF_BYPOSITION );

    //获取当前菜单项的子菜单
    Cmenu* pSubMenu = pMenuAdd->GetSubMenu( iLoop );

    // 检查是否有一个子菜单
    if( !pSubMenu )
    {
        // 追加菜单项到目标菜单
        VERIFY( pMenuDestination->AppendMenu(
            pMenuAdd->GetMenuState( iLoop,
                                    MF_BYPOSITION ),
            pMenuAdd->GetMenuItemID( iLoop ),
            sMenuAddString ));
    }
    else
    {
        //创建或插入一新的弹出菜单项
        int iInsertPosDefault = -1;
        if( bTopLevel )
            //寻找已存在的弹出菜单
            for( int iLoop = 0; iLoop < iMenuDestItemCount; iLoop++ )
            {
                //从目标菜单中获取菜单字符串
                CString sDest;
                pMenuDestination->GetMenuString( iLoop, sDest,
                                                    MF_BYPOSITION );

                if( sMenuAddString == sDest )
```

```

    {
        //获取目标菜单项的子菜单
        Cmenu* pSubMenuDest =
            MenuDestination->GetSubMenu( iLoop );

        if( pSubMenuDest )
        {
            //菜单合并
            MergeMenu( pSubMenuDest, pSubMenu );
            continue;
        }
    }

    if( iInsertPosDefault == -1 && (sDest == "&Window" ||
        sDest == "&Help"))
        iInsertPosDefault = iLoop;
}

if( iInsertPosDefault == -1 )
    iInsertPosDefault = pMenuDestination->GetMenuItemCount();

Cmenu NewPopupMenu;
VERIFY( NewPopupMenu.CreatePopupMenu() );

// 合并新的弹出菜单
MergeMenu( &NewPopupMenu, pSubMenu );

// 插入一新的弹出菜单
VERIFY( pMenuDestination-
        iInsertPosDefault,
        MF_BYPOSITION | MF_POPUP | MF_ENABLED,
        (UINT)NewPopupMenu.GetSafeHmenu(),
        sMenuAddString ));

NewPopupMenu.Detach();
}
}

```

}

1.1.3 如何确定顶层菜单所占据的菜单行数

【问题】

在应用程序的开发过程中,有时需要知道主菜单条(顶层菜单)所占据的高度,以便调整整个窗口的布局,那么如何确定顶层菜单所占据的菜单行数呢?

【解决方法】

首先,需要计算主框窗口的高度和客户区;其次,从主框窗口的高度中减去客户区、框边界以及标题的高度;最后,除以菜单栏的高度即可。下边的例程就可以完成计算主菜单所占据的行数。

【实现程序】

```
int CMainFrame:: GetMenuRows ()
{
    CRect rcFrame,rcClient;

    //获取主框架的尺寸
    GetWindowRect (rcFrame);

    //获取客户区的尺寸
    GetClientRect (rcClient);

    return (rcFrame.Height () -rcClient.Height ()- :: GetSystemMetrics (SM_CYCAPTION)
    - (:: GetSystemMetrics (SM_CYFRAME) *2)) / :: GetSystemMetrics (SM_CYMENU);
}
```

1.1.4 如何给系统菜单添加一个菜单项

【问题】

系统菜单是所有程序所公用的菜单资源,如何在应用程序中给系统菜单添加一个菜单项,以满足某些特殊要求呢?

【解决方法】

其实现方法如下:

(1) 使用“Resource Symbols”(资源符号)对话框(即在VC的编程环境中的“View”菜单中选择“Resource Symbols. . .”可以显示该对话框)定义菜单项ID,该ID应大于0x0F而小于0Xf000。

(2) 调用CWnd::GetSystemMenu 获取系统菜单的指针并调用CWnd::Appendmenu 将菜单项添加到系统菜单中。

(3) 使用 ClassWizard 处理 WM_SYSCOMMAND 消息并检测用户菜单的 Nid 参数, 加入相应的响应代码。

【实现程序】

```
int CMainFrame:: OnCreate (LPCREATESTRUCT lpCreateStruct)
{

    ASSERT (IDM_MYSYSITEM &0Xfff0)==IDM_MYSYSITEM);

    ASSERT (IDM-MYSYSITEM<0Xf000);

    //获取系统菜单
    Cmenu* pSysmenu=GetSystemmenu (FALSE);

    ASSERT_VALID (pSysMenu);

    //在系统菜单中增加分隔符和新菜单项
    CString StrMenuItem (_T ("New menu item"));

    pSysMenu->Appendmenu (MF_SEPARATOR);

    pSysMenu->AppendMenu (MF_STRING, IDM_MYSYSITEM, strMenuItem);

    ...

}
```

//使用 ClassWizard 处理 WM_SYSCOMMAND 消息并检测用户菜单的 Nid 参数, 加入相应处理代码

```
void CMainFrame:: OnSysCommand (UINT Nid,LPARAM lParam)
{
    //检测是否系统菜单项被选取
    if ( (Nid & 0Xfff0)==IDM_MYSYSITEM)
    {
        //加入自己的处理代码
        .....
    }
    else
        CMDIFrameWnd:: OnSysCommand (Nid, lParam);
}
```

```
}
```

1.1.5 为什么即使调用 EnableMenuItem 菜单项后，菜单项还处于禁止状态

【问 题】

前面介绍过通过调用函数 EnableMenuItem 可以改变菜单项的属性，如是否处于禁止状态等。但有时候即使调用 EnableMenuItem 函数激活菜单项，菜单项依然处于禁止状态，如何处理这种情况？

【解决方法】

需要将 CFrameWnd:: m_bAutomenuEnable 设置为 FALSE，如果该数据成员为 TRUE（缺省值），主框架将自动地禁止没有 ON_UPDATE_COMMAND_UI 或者 ON_COMMAND 的菜单项。

【实现程序】

```
m_bAutoMenuEnable=FALSE;

//使菜单项处于使能状态
CMenu* pMenu=GetMenu ();
ASSERT_VALID (pMenu);
pMenu->EnableMenuItem (ID_MENU_ITEM,MF_BYCOMMAND | MF_ENABLED);
```

1.1.6 如何在已存在的菜单中插入一子菜单项

【问 题】

前面介绍过如何将两个菜单进行合并，但在有些应用中，随着运行环境的变化或应用程序本身运行的需要，应用程序必须在运行过程中动态地插入一些菜单项，这些工作将如何进行呢？

【解决方法】

在 Cmenu 类中提供了插入菜单函数 InsertMenu，通过它便可以实现上述工作，与之相反，在 Cmenu 类中也提供了移去一个菜单项的函数 RemoveMenu。下面先介绍一下 InsertMenu 函数的使用方法：

```
BOOL InsertMenu( UINT nPosition,
                 UINT nFlags,
                 UINT nIDNewItem = 0,
                 LPCTSTR lpszNewItem = NULL
                );
```

或

```
BOOL InsertMenu( UINT nPosition,
                 UINT nFlags,
                 UINT nIDNewItem,
```

```
const Cbitmap* pBmp
```

```
);
```

其中函数参数与 1.2 节中的 AppendMenu 函数类似，这里不再介绍。

移去一个菜单项的函数 RemoveMenu 的用法为：

```
BOOL RemoveMenu( UINT nPosition, UINT nFlags );
```

其中参数“nPosition”用于指定被移去的菜单项，参数“nFlags”用于解释参数“nPosition”的指定方式，其取值为：

- MF_BYCOMMAND: 以命令 ID 的方式来指定菜单项，为缺省值。
- MF_BYPOSITION: 以已存在菜单项的位置来指定菜单项，第一个菜单项位置为 0。

有了上述函数，用户便可以实现子菜单项的插入，见下面程序段。

【实现程序】

//若 pViewClass 为 NULL，则不加入菜单

```
void CmainFrame::AddViewMenu(CruntimeClass* pViewClass)
```

```
{
```

```
    const int INSERT_POSITION = 3;
```

```
    const int DEFAULT_MENU_COUNT = 6;
```

```
    enum { PROGRAM_MENU, LAYER_MENU, KEYMAP_MENU,
           SAMPLE_MENU };
```

```
    //获取菜单句柄
```

```
    Cmenu* pMenu = GetMenu();
```

```
    ASSERT(pMenu->GetMenuItemCount() == DEFAULT_MENU_COUNT
```

```
    || pMenu->GetMenuItemCount() == DEFAULT_MENU_COUNT + 1);
```

```
    //若目前已存在一扩展的菜单，则移去它
```

```
    if (pMenu->GetMenuItemCount() == DEFAULT_MENU_COUNT + 1)
```

```
        pMenu->RemoveMenu(INSERT_POSITION, MF_BYPOSITION);
```

```
    //增加适当的菜单
```

```
    int iMenuToAdd = -1;
```

```
    if (pViewClass == RUNTIME_CLASS(CprogramView))
```

```
        iMenuToAdd = PROGRAM_MENU;
```

```
    else if (pViewClass == RUNTIME_CLASS(ClayerView))
```

```
        iMenuToAdd = LAYER_MENU;
```

```
    else if (pViewClass != NULL)
```

```

        ASSERT(FALSE);

        if (iMenuToAdd != -1)
        {
            // 获取要追加的菜单
            Cmenu objectMenu;
            objectMenu.LoadMenu(IDR_OBJECT_MENUS);
            Cmenu* pPopupMenu = objectMenu.GetSubMenu(iMenuToAdd);
            ASSERT(pPopupMenu != NULL);

            CString strMenuItem;
            objectMenu.GetMenuString(iMenuToAdd, strMenuItem, MF_BYPOSITION);

            //插入菜单项
            VERIFY(pMenu->InsertMenu(INSERT_POSITION, MF_BYPOSITION |
                                    MF_POPUP,
                                    (UINT)pPopupMenu->GetSafeHmenu(), strMenuItem));

            //从其它菜单中移去
            objectMenu.RemoveMenu(iMenuToAdd, MF_BYPOSITION);
        }
        DrawMenuBar();
    }
}

```

1.1.7 如何实现一个使用图标的自画菜单

【问 题】

有时，为增强应用程序的视角效果，经常会使用一些带图标的自画菜单，在程序中如何实现这种菜单呢？

【解决方法】

实现这种菜单的代码使用了不少图形函数，这里不详细讨论。其实现细节可从程序中领会，为了使用方便，我们以类（TCMenu）的形式将能实现上述功能的代码给出。这样只用将该类加入到你的工程文件中，调用该类的方法即可。

【实现程序】

//头文件(TCMenu.h)的代码部分

```

#ifndef TCMENU_H
#define TCMENU_H
class TCMenuData
{

```