**第6课 菜单**

**1 菜单添加响应**

资源视图中打开菜单，在需要添加响应的菜单项上右击属性将Popup改为false，右键添加响应函数，选择响应函数所在的类即可，编辑响应函数。

**菜单消息响应顺序：**

4个类的消息响应优先次序分别是：1.View;2.CDOC;3.CMainFrame.4.CWinAPP.

**注意：**

View类和CMainFrame继承自CWIND所以可以直接调用MessageBox(“…”)函数，CDOC和CWinAPP只能用一般的AfxMessageBox(\_T("doc clicked"))函数。

**消息分类**

标准消息（以WM\_开头的消息，但不包括ON\_COMMAND），如WM\_Paint；b;命令消息 ON\_COMMAND(IDM\_PHONE1, OnPhone1)，菜单和工具栏的消息。c.通告消息：按钮，对话框，列表框发出的消息。

CCmdTarget只能接受命令消息,如文档类和APP类只能接受命令消息，不能接受标准消息。而从CCmdTarget派生的CWnd可以接收命令消息，也可以接受标准消息。通告消息都可以接收。

**2创建标记菜单**

在CMainFrame::OnCreate函数下：

GetMenu(); CWnd的成员函数，返回整个菜单栏的指针。

GetMenu()->GetSubMenu(0); 获取子菜单，参数为子菜单索引，从0开始

GetMenu()->GetSubMenu(0)->CheckMenuItem(0,MF\_BYPOSITION|MF\_CHECKED); /通过位置索引设置标记

GetMenu()->GetSubMenu(0)->CheckMenuItem(ID\_FILE\_NEW,MF\_BYCOMMAND|MF\_CHECKED); //通过菜单项的ID进行索引设置标记

**注意**

在VS2010中，由于CMainFrame继承至CFrameWndEx会导致无法像VC6.0一样使用GetMenu()等函数，所以这时在建立工程的时候就应该选择为：  
第一步：项目标准： MFC标准  
视觉样式和颜色：windows 本机/默认  
第五步：命令栏：使用经典菜单  
这样就和VC6.0中的菜单一样了

**3创建缺省菜单项**

CMenu::SetDefaultItem(UINT ,BOOL); //bool为false则UINT为ID表示，true为地址索引表示。

GetMenu()->GetSubMenu(0)->SetDefaultItem(1,true);将文件-打开设置为缺省项。

**注意**

确定菜单的索引号，注意从0开始，分隔符也算数。

一个子菜单只能有一个缺省菜单。

**4创建图形标记菜单**

CMenu::SetMenuItemBitmaps(UINT nposition, UINT nFlags, const CBitmap\* pBmpUnchecked, const CBitmap\* pBmpChecked);

nFlags: MF\_BYPOSITION 和MF\_BYCOMMAND位置索引和ID索引。

pBmpUnchecked： 没有选中时的位图

pBmpChecked： 选中时的位图

具体操作：

先通过GetSystemMetrics()函数获取菜单图标大小，按这个大小制作位图：

CString str;

str.Format(\_T("x=%d,y=%d"),GetSystemMetrics(SM\_CXMENUCHECK),GetSystemMetrics(SM\_CYMENUCHECK));

MessageBox(str); //显示出来宽高，13\*13的，然后制作13\*13的位图

然后CMainFrame中添加成员变量：m\_bitmap

m\_bitmap.LoadBitmapW (IDB\_BITMAP1);

GetMenu()->GetSubMenu(0)->SetMenuItemBitmaps(0,MF\_BYPOSITION,&m\_bitmap,&m\_bitmap);

**5屏蔽菜单项**

CMenu::EnableMenuItem(UINT nIDEnableItem, UINT nEnable);

具体操作：

在CMainFrame构造函数中设置m\_bAutoMenuEnable=false;从而EnableMenuItem可以起作用，然后在OnCreate函数中：

GetMenu()->GetSubMenu(0)->EnableMenuItem(1,MF\_BYPOSITION|MF\_DISABLED|MF\_GRAYED)； //屏蔽打开菜单项并使其变灰

**6取消菜单**

CMenu::SetMenu(CMenu\* pMenu); //pMenu为NULL时为当前菜单被移走

**取消并重新加载菜单，可用于更换菜单**

SetMenu(NULL); //取消当前菜单

CMenu menu; //定义成成员变量，或紧接着Detach函数

menu.LoadMenuW(IDR\_MAINFRAME); //重新加载菜单

SetMenu(&menu);

menu.Detach();

**7命令更新**

在CMainFrame构造函数中取消设置m\_bAutoMenuEnable=false;选择采用命令更新机制。可以看到编辑子菜单下的撤销、剪切、复制、粘贴等菜单项都不能使用了。

添加时间处理函数，其中消息类型选择UPDATE\_COMMAND\_UI，创建相应函数：

void CMainFrame::OnUpdateEditCut(CCmdUI \*pCmdUI)

该函数传递了一个CCmdUI类的指针。在该相应函数中：

pCmdUI->Enable(); //这样剪切键就可以使用了, Enable的默认参数为true

pCmdUI->Enable(FALSE); //使菜单项无法使用

在执行Enable函数之前可以先进行判断，判断pCmdUI是否指向当前菜单项：

if(2==pCmdUI->m\_nIndex) //通过地址判断，这是菜单项的剪切可用，工具栏中剪切依然不可用

pCmdUI->Enable();

或通过ID号索引：

if(ID\_Edit\_Cut==pCmdUI->m\_nID) //通过ID号索引，菜单项和工具栏中的都可用

pCmdUI->Enable();

**8增加PopupMenu菜单**

Project->Add to Project->component and controls在vs2010中找不到，手动创建。

具体操作：

资源中新建一个菜单

CMenuView类中添加右键响应函数OnRButtonDown

CMenu menu;

menu.LoadMenu(IDR\_Popup);

CMenu \*pPopup=menu.GetSubMenu(0); //获取子菜单，只有一个子菜单

ClientToScreen(&point);

pPopup->TrackPopupMenu(TPM\_LEFTALIGN | TPM\_RIGHTBUTTON, point.x, point.y,

this); //point是屏幕坐标，所以要先ClientToScreen将客户区坐标转换为屏幕坐标

由于TrackPopupMenu调用的是this指针，因此弹出菜单中的菜单项的响应只能由view执行，框架类不能响应，为了使框架类也能指向响应操作，需将this换为其父窗口的指针GetParent()。

**9动态添加菜单**

**函数的使用**

1函数原型：BOOL AppendMenu（hMenu hMenu，UINT uFlags，UINT uIDNewltem，LPCTSTR lpNewltem）;

**hMenu**

　　将被修改的菜单条、下拉式菜单、子菜单、或快捷菜单的句柄。

**UFlag**

　　控制新菜单项的外观和性能的标志。此参数可以是备注里所列值的组合。

**UIDNewltem**

　　指定新菜单项的标识符，或者当uFlags设置为MF\_POPUP时，表示下拉式菜单或子菜单的句柄。

**LpNewltem**

　　指定新菜单项的内容。此参数的含义取决于参数uFlags是否包含MF\_BITMAP, MF\_OWNERDRAW或MF\_STRING标志，如下所示：

**MF\_BITMAP**

　　含有位图句柄。MF\_STRING：以`\O’结束的字符串的指针。

**MF\_OWNERDRAW**

　　含有被应用程序应用的32位值，可以保留与菜单项有关的附加数据。当菜单被创建或其外观被修改时，此值在消息WM\_MEASURE或WM\_DRAWITEM的参数IParam指向的结构，成员itemData里。

　　返回值：如果函数调用成功，返回非零值；如果函数调用失败，返回值是零。若想获得更多的错误信息，请调用GetLastError函数。

　　备注：一旦菜单被修改，无论它是否在显示窗口里，应用程序必须调用函数DrawMenuBar。

　　为了使键盘加速键能控制位留或自己绘制的菜单项，菜单的拥有者必须处理WM\_MENUCHAR消息。

　　参见自绘制菜单和WM\_MENUCHAR消息。

2函数原型：HMENU CreatePopupMenu（VOID）

函数功能：该函数创建一个下拉式菜单、子菜单或快捷菜单。此菜单最初是空的，但可用函数InsertMenultem来插入或追加菜单项。也可用函数InsertMenu来插人菜单项，用AppendMenu来追加菜单项。

　　参数：无。

　　返回值：如果函数调用成功，返回值是新创建菜单的句柄。如果函数调用失败，返回值是NULL。若想获得更多的错误信息，请调用GetLastError函数。

3 BOOL InsertMenu(HMENU hMenu,UINt uPosition,UINT uFlags,UINT uIDNewltem,LPCTSTR lpNewltem)

**具体操作**

CMenu menu;

menu.CreatePopupMenu();

// GetMenu()->AppendMenu(MF\_POPUP,(UINT)menu.m\_hMenu,\_T("WinSun")); //增加菜单，在最后面

GetMenu()->InsertMenu(2,MF\_BYPOSITION | MF\_POPUP,(UINT)menu.m\_hMenu, \_T("WinSun")); //插入菜单

menu.AppendMenu(MF\_STRING,111,\_T("Hello")); //插入菜单项

menu.AppendMenu(MF\_STRING,112,\_T("Weixin"));

menu.AppendMenu(MF\_STRING,113,\_T("Mybole"));

menu.Detach();

< xmlnamespace prefix ="v" ns ="urn:schemas-microsoft-com:vml" />

GetMenu()->GetSubMenu(0)->AppendMenu(MF\_STRING,114,"Welcome"); //在文件子菜单下加菜单项

GetMenu()->GetSubMenu(0)->InsertMenu(ID\_FILE\_OPEN,

MF\_BYCOMMAND | MF\_STRING,115， "维新"); //在新建和打开之间插入菜单项

// GetMenu()->DeleteMenu(1,MF\_BYPOSITION); //删除菜单

// GetMenu()->GetSubMenu(0)->DeleteMenu(2,MF\_BYPOSITION); //删除打开菜单项

**10 为动态创建的菜单增加消息响应的步骤**

a.在resource.h中增加#define IDM\_HELLO 123

b.在MainFrm.h中加入afx\_msg void OnHello();

c.MainFrm.cpp中加入ON\_COMMAND(IDM\_HELLO,OnHello)

d.最后加入

void CMainFrame::OnHello()

{

MessageBox("Hello!");

}

**11动态增加电话号码本步骤**

1 view类中处理WM\_Char消息。

void CMenu2View::OnChar(UINT nChar, UINT nRepCnt, UINT nFlags)

{

if(0x0d==nChar) //回车

{

if(0==++m\_nIndex) //如果第一次按回车，则创建菜单

{

m\_menu.CreatePopupMenu ();

GetParent()->GetMenu()->AppendMenuW(MF\_POPUP, (UINT)m\_menu.m\_hMenu, \_T("PhoneBook"));

//菜单属于框架类所以要GetParent()->GetMenu()来获取

GetParent()->DrawMenuBar();

//重绘菜单，否则不显示，又因为菜单属于框架类所以要GetParent()->

}

m\_menu.AppendMenuW (MF\_STRING, IDM\_PHONE1+m\_nIndex, m\_strLine.Left (m\_strLine.Find (' '))); //以字符串空格前的字符为名创建菜单项

m\_strArray.Add (m\_strLine); //m\_strArray为CstringArray类型的成员变量，存储CString

m\_strLine.Empty (); //清空字符串

Invalidate(); //窗口背景被擦除

}

else

{

m\_strLine+=char(nChar);

dc.TextOut(0,0,m\_strLine);

}

CView::OnChar(nChar, nRepCnt, nFlags);

}

2 添加菜单项的消息处理函数

Menu2View.h 中加入：

afx\_msg void OnPhone1();

afx\_msg void OnPhone2();

afx\_msg void OnPhone3();

Menu2View.cpp中加入：

ON\_COMMAND(IDM\_PHONE1, &CMenu2View::OnPhone1)

ON\_COMMAND(IDM\_PHONE2, &CMenu2View::OnPhone2)

ON\_COMMAND(IDM\_PHONE3, &CMenu2View::OnPhone3)

void CMenu2View::OnPhone1()

{

CClientDC dc(this);

dc.TextOutW (0,0,m\_strArray.GetAt (0)); //取出m\_strArray中数据

}

**12在CMainFrame类中截获消息处理函数**

CMainFrame中增加虚函数：

BOOL CMainFrame::OnCommand(WPARAM wParam, LPARAM lParam)

{

// TODO: 在此添加专用代码和或调用基类

int MenuCmdId=LOWORD(wParam);

CMenu2View \*pView=(CMenu2View\*)GetActiveView();

// GetActiveView()为CMainFrame类的成员函数，获取view类指针

if(MenuCmdId>=IDM\_PHONE1&&MenuCmdId<IDM\_PHONE1+pView->m\_strArray.GetSize())

{

//MessageBox(\_T("test"));

CClientDC dc(pView); //用View类指针创建dc

dc.TextOutW (0,0,pView->m\_strArray .GetAt (MenuCmdId-IDM\_PHONE1));

return true;

}

return CFrameWnd::OnCommand(wParam, lParam);

}

注意要增加两个头文件：

#include "Menu2Doc.h"

#include "Menu2View.h"

为了加深理解可以看一看“WM\_COMMAND消息路由”。