# 转 MFC窗口最小化到托盘

2013年10月13日 16:29:10 阅读数：3124

把程序放到托盘上的本质就是先在托盘区绘制一个图标，然后把程序隐藏不见，再对托盘的图标进行消息处理，就可以了。绘制图标以及确定图标所传送消息的函数只有一个，那就是 WINSHELLAPI BOOL WINAPI Shell_NotifyIcon( DWORD dwMessage, PNOTIFYICONDATA pnid ); 这个函数负责向系统传递消息，以添加、修改或删除托盘区的图标。它的返回值是个布尔类型的。就是说，如果返回0就是不成功，非0才成功。参数dwMessage 是表示这个函数的应用功能是哪一方面，是添加、删除，还是修改图标。如果是添加，则它的值为NIM_ADD；删除则是NIM_DELETE；而修改是NIM_MODIFY。参数pnid就是具体的和程序在托盘区的图标有关系的结构了。它的定义如下：

```cpp
typedef struct _NOTIFYICONDATA { DWORD cbSize;
HWND hWnd;
UINT uID;
UINT uFlags;
UINT uCallbackMessage;
HICON hIcon;
char szTip[64];
} NOTIFYICONDATA, *PNOTIFYICONDATA;
```

cbSize : 结构的长度，用“位”来做单位。一般在程序中，我们用(DWORD)sizeof(NOTIFYICONDATA) 给它赋值。
HWnd : 一个句柄，如果对托盘中的图标进行操作，相应的消息就传给这个句柄所代表的窗口。大多数情况下是this->m_hWnd。
uID： 在工程中定义的图标ID
uFlags : 这个成员标志着其他哪些成员的数据是有效的，分别为NIF_ICON, NIF_MESSAGE, NIF_TIP，分别代表着数据有效的成员是hIcon, uCallbackMessage, szTip。当然，三个值可以用“|”联系到一起。

下面分别对涉及到的成员进行阐述：
hIcon： 要增加，删除或修改的图标句柄。如果只知道个uID, 一般可能会用函数LoadIcon来得到句柄。例如LoadIcon ( AfxGetInstanceHandle() ,MAKEINTRESOURCE (IDR_MAINFRAME) )。
uCallbackMessage : 这在对托盘区的操作中，是比较重要的数据成员。这是个消息标志，当用鼠标对托盘区相应图标进行操作的时候，就会传递消息给Hwnd所代表的窗口。所以说，在uFlags中，一般都得标志它有效。这里一般都是自定义的消息。
szTip： 鼠标移动到托盘图标上时的提示文字。

## 托盘编程例子：

1、将程序最小化到系统托盘区的函数toTray()。

```cpp
void CTimeWakeDlg::toTray()
{
  NOTIFYICONDATA nid;
  nid.cbSize=(DWORD)sizeof(NOTIFYICONDATA);
  nid.hWnd=this->m_hWnd;  nid.uID=IDR_MAINFRAME;
  nid.uFlags=NIF_ICON|NIF_MESSAGE|NIF_TIP ;
  nid.uCallbackMessage=WM_SHOWTASK;//自定义的消息名称
  nid.hIcon=LoadIcon(AfxGetInstanceHandle(),MAKEINTRESOURCE(IDR_MAINFRAME));
  strcpy(nid.szTip,"计划任务提醒");//信息提示条为“计划任务提醒”
  Shell_NotifyIcon(NIM_ADD,&nid);//在托盘区添加图标
  ShowWindow(SW_HIDE);//隐藏主窗口
}
```

2、 程序已经最小化到托盘区了，但是对托盘图标的操作如何进行呢？这就体现了结构NOTIFYICONDATA的成员uCallbackMessage 的作用了。它所提供的作用就是，当用户用鼠标点击托盘区的图标的时候（无论是左键还是右键），会向hWnd所代表的窗口传送消息，如果是上例，消息的名称就是WM_SHOWTASK。
根据VC的消息机制，对自定义消息增加消息响应函数。
在头文件的//{{AFX_MSG和//}}AFX_MSG之间声明消息响应函数：
afx_msg LRESULT onShowTask(WPARAM wParam,LPARAM lParam);
然后在CPP文件中添加消息映射。在BEGIN_MESSAGE_MAP和END_MESSAGE_MAP 之间加入：
ON_MESSAGE(WM_SHOWTASK,onShowTask)将消息和消息响应函数映射起来。
然后就是在CPP文件中加入函数onShowTask的实现了：

```cpp
LRESULT CTimeWakeDlg::onShowTask(WPARAM wParam,LPARAM lParam)
//wParam接收的是图标的ID，而lParam接收的是鼠标的行为
{
  if(wParam!=IDR_MAINFRAME)
       return 1;
  switch(lParam)
  {
      case WM_RBUTTONUP://右键起来时弹出快捷菜单，这里只有一个“关闭”
       {
           LPPOINT lpoint=new tagPOINT;
           ::GetCursorPos(lpoint);//得到鼠标位置
           CMenu menu;
           menu.CreatePopupMenu();//声明一个弹出式菜单


          //增加菜单项“关闭”，点击则发送消息WM_DESTROY给主窗口（已隐藏），将程序结束。
          menu.AppendMenu(MF_STRING,WM_DESTROY,"关闭");
          //确定弹出式菜单的位置
          menu.TrackPopupMenu(TPM_LEFTALIGN,lpoint->x,lpoint->y,this);
         //资源回收
         HMENU hmenu=menu.Detach();
         menu.DestroyMenu();
         delete lpoint;
      } break;
     case WM_LBUTTONDBLCLK://双击左键的处理
     {
         this->ShowWindow(SW_SHOW);//简单的显示主窗口完事儿
     } break;
  }  return 0;
}
```

3. 调用
在窗口添加WM_SIZE的消息OnSize（）

```cpp
void CTimeWakeDlg::OnSize(UINT nType, int cx, int cy)
{
    CDialog::OnSize(nType, cx, cy);


    // TODO: 在此处添加消息处理程序代码
    if(nType==SIZE_MINIMIZED)   //判断是最小化按钮时，执行最小化到托盘函数
        ToTray();
}
```

注意：
定义消息名称以消息号，并注册消息，该步很重要！我就是因为没有注册消息，导致调试了很久都找不到问题所在。该步都是在Dlg.cpp(Dlg的实现中)中操作。定义消息名称和消息号：#define WM_ SHOWTASK (WM_USER+1001)，1001只是用于指定一个消息号，可以随便指定。注册则是在BEGIN_MESSAGE_MAP(Dlg,CDialog)和END_MESSAGE_MAP()之间添加ON_MESSAGE(WM_ SHOWTASK, onShowTask)。

文章标签： MFC 窗口 最小化 托盘

个人分类： 纯编程