转 MFC选项卡的实现

2013年10月16日 22:49:33 阅读数：3082

方案一

在对话框上放置一个Tab Control的 控件 ,再在对话框上放置所需的 控件 （本例放置了2个按钮，试图在每个标签中显示一个）。然后利用Class Wizard来为Tab Control 控件 创建一个 控件 变量，该变量是CTabCtrl类的，再为其他 控件 也创建相应的 控件 类。 在主对话框的初始函数中CProperty1Dlg::OnInitDialog()加入如下代码：

```cpp
//本例插入两个标签，实际运用中可通过循环插入所需个数的标签，运行后默认第一个标签被选中
m_tab.InsertItem( 0, _T("Tab1") );
m_tab.InsertItem( 1, _T("Tab2") );
//将不是第一个标签的控件隐藏掉，只留下你要的控件
m_button2.ShowWindow( SW_HIDE );
再利用ClassWizard处理Tab Control的 TCN_SELCHANGE 的消息。在消息处理函数中，利用CWnd::ShowWindow来使相应的控件显示和隐藏。
void CProperty1Dlg::OnSelchangeTab1(NMHDR* pNMHDR, LRESULT* pResult)
{
//GetCurSel返回当前被选中的标签的索引号（以0为基础算起）
int sel = m_tab.GetCurSel();
switch(sel)
{
case 0:
m_button1.ShowWindow( SW_SHOW );
m_button2.ShowWindow( SW_HIDE );
break;
case 1:
m_button2.ShowWindow( SW_SHOW );
m_button1.ShowWindow( SW_HIDE );
break;
}
*pResult = 0;
}
```

方案二

本这个方案中，我将使用MFC中现成的CPropertySheet和CPropertyPage类来完成将 控件 分散到各个对话框类中。

首先加入两个（或数个）对话框资源。修改各对话框资源的属性，将对话框的Caption属性改为你要在标签上所显示的文字。将对话框的Style属性改为：Child， Border属性改为：Thin， 只选中Title Bar复选框，去掉其他复选框。然后你可以在这些对话框中加入要分开显示的各个 控件 。

为上述对话框资源分别制作一个对话框类，该对话框类是从CPropertyPage继承。这样一来各子对话框类就好了，主对话框类可以直接使用CPropertySheet类。使用如下代码即可：

```cpp
CPropertySheet sheet("属性页对话框");
CPage1 page1;
CPage2 page2;
//加入子对话框作为一个属性页
sheet.AddPage(&page1);
sheet.AddPage(&page2);
//产生一个模态对话框，也可以使用Create方法来产生一个非模态对话框（具体参见MSDN）
sheet.DoModal();
```

如何在主对话框中放置其他 控 件 呢？如果直接使用CPropertySheet的话，是不可以的，但是别忘了我们可以从CPropertySheet类继承自己的类啊！

方案三

首先还是要创建那些要在属性页中的显示的子对话框类，创建步骤和方案二一样，都是从CPropertyPage继承。

这次我们将从CPropertySheet类继承自己的类（假设类名为CMySheet）。我们要在这里放上一个button 控 件 。那么现在先在CMySheet中加入一个CButton类的成员变量m_button。

在CMySheet类中的OnInitDialog()函数里，这样写：

```cpp
BOOL bResult = CPropertySheet::OnInitDialog();
//取得属性页的大小
CRect rectWnd;
GetWindowRect(rectWnd);
//调整对话框的宽度
SetWindowPos(NULL, 0, 0,rectWnd.Width() + 100,rectWnd.Height(),SWP_NOMOVE | SWP_NOZORDER | SWP_NOACTIVATE);
CRect rectButton(rectWnd.Width() + 25, 25,rectWnd.Width()+75, 75);
//用程序创建一个按钮
m_button.Create("Button", BS_PUSHBUTTON, CRect(rectWnd.Width(), 25,rectWnd.Width()+75, 50) , this, 1);
//显示这个按钮
m_button.ShowWindow( SW_SHOW );
CenterWindow();
return bResult;
```

使用方案三虽然能在主对话框中加入 控件 ，但是也比较麻烦，首先所加的 控件 只能在属性页的右边或下边。并且用 程序 来产生 控件 比较烦琐，位置与大小不易控制。那么还有其他方法，既能在对话框中加入属性页，又能在主对话框随意添加 控件 ？

方案四

不从CPropertySheet继承自己的类，还是直接使用它。各属性页的子对话框类还是需要的，创建方法和上述两个方案相同。

首先我们新建一个基于对话框的工程。在编辑已有的一个主对话框中可以自由加一些所需的 控件 ，但是得留出一定的空间用于放置属性页。

在主对话框类里加入一个CPropertySheet类的一个成员变量（m_sheet）代表整个属性页。再加入一些各子对话框类的实例作为成员变量（m_page1、m_page2……）。

在主对话框类的OnInitDialog()函数中加入：

```cpp
//加入标签，标签名由各个子对话框的标题栏决定
m_sheet.AddPage(&m_page1);
m_sheet.AddPage(&m_page2);
//用Create来创建一个属性页
m_sheet.Create(this, WS_CHILD | WS_VISIBLE, WS_EX_CONTROLPARENT);
RECT rect;
m_sheet.GetWindowRect(&rect);
int width = rect.right - rect.left;
int height = rect.bottom - rect.top;
//调整属性页的大小和位置
m_sheet.SetWindowPos(NULL, 20, 50, 0, 0, SWP_NOSIZE | SWP_NOZORDER | SWP_NOACTIVATE);
```

这个方案可以自由在主对话框中加一些必要的 控件 ，而且属性页中的 控件 也都分散在了各个子对话框类中，使用非常方便。

方案五

使用Tab Control，并且从CTabCtrl 控件 类继承自己的类（CTabSheet）来处理。

首先我先介绍一下如何使用CTabSheet。

先要制作子对话框类，这次的子对话框类不要从CPropertyPage继承，而是直接从CDialog继承。并且各个子对话框资源的属性应设置为：Style为Child， Border为None。

在主对话框资源中，加入一个Tab Control，并且适当调整位置和大小。利用ClassWizard来为这个Tab Control创建一个CTabSheet的 控件 变量。

在主对话框的OnInitDialog()加入：

m_sheet.AddPage("tab1", &m_page1, IDD_DIALOG1);

m_sheet.AddPage("tab2", &m_page2, IDD_DIALOG2);

m_sheet.Show();

就这样就可以在对话框上制作出一个完美的属性页了。效果和上图完全一样。

下面我就来讲讲CTabSheet类的细节内容。

CTabSheet是从CTabCtrl继承来的，用于Tab Control的 控件 类。在类中有一个成员变量用来记录各子对话框的指针CDialog* m_pPages[MAXPAGE]; MAXPAGE是该类所能加载的标签的最大值。

类中有一个AddPage方法，用于记录子对话框的指针和所使用对话框资源的ID号。

```cpp
BOOL CTabSheet::AddPage(LPCTSTR title, CDialog *pDialog,UINT ID)
{
if( MAXPAGE == m_nNumOfPages )
return FALSE;
//保存目前总的子对话框数
m_nNumOfPages++;
//记录子对话框的指针、资源ID、要在标签上显示的文字
m_pPages[m_nNumOfPages-1] = pDialog;
m_IDD[m_nNumOfPages-1] = ID;
m_Title[m_nNumOfPages-1] = title;
return TRUE;
}
在使用AddPage加入了若干子对话框后，必须调用CTabSheet的Show方法来真正生成标签和子对话框。
void CTabSheet::Show()
{
//利用CDialog::Create来创建子对话框，并且使用CTabCtrl::InsertItem来加上相应的标签
for( int i=0; i < m_nNumOfPages; i++ )
{
m_pPages[i]->Create( m_IDD[i], this );
InsertItem( i, m_Title[i] );
}
//由于对话框显示时默认的是第一个标签被选中，所以应该让第一个子对话框显示，其他子对话框隐藏
m_pPages[0]->ShowWindow(SW_SHOW);
for( i=1; i < m_nNumOfPages; i++)
m_pPages[i]->ShowWindow(SW_HIDE);
SetRect();
}
```

生成好标签和子对话框后，调用CTabSheet::SetRect来计算并调整属性页的大小。

```cpp
void CTabSheet::SetRect()
{
CRect tabRect, itemRect;
int nX, nY, nXc, nYc;
//得到Tab Control的大小
GetClientRect(&tabRect);
GetItemRect(0, &itemRect);
//计算出各子对话框的相对于Tab Control的位置和大小
nX=itemRect.left;
nY=itemRect.bottom+1;
nXc=tabRect.right-itemRect.left-2;
nYc=tabRect.bottom-nY-2;
//利用计算出的数据对各子对话框进行调整
m_pPages[0]->SetWindowPos(&wndTop, nX, nY, nXc, nYc, SWP_SHOWWINDOW);
for( int nCount=1; nCount < m_nNumOfPages; nCount++ )
m_pPages[nCount]->SetWindowPos(&wndTop, nX, nY, nXc, nYc, SWP_HIDEWINDOW);
}
```

在单击标签栏后，应该是相应的子对话框显示，正在显示的子对话框应该隐藏。因此利用ClassWizard来处理WM_LBUTTONDOWN消息。

```cpp
void CTabSheet::OnLButtonDown(UINT nFlags, CPoint point)
{
CTabCtrl::OnLButtonDown(nFlags, point);
//判断是否单击了其他标签
if(m_nCurrentPage != GetCurFocus())
{
//将原先的子对话框隐藏
m_pPages[m_nCurrentPage]->ShowWindow(SW_HIDE);
m_nCurrentPage=GetCurFocus();
//显示当前标签所对应的子对话框
m_pPages[m_nCurrentPage]->ShowWindow(SW_SHOW);
}
}
```

这样利用CTabSheet这个类就可以轻松地在对话框上放置自己的属性页了，并且 控 件 都分散在各子对话框类中，符合对象封装的思想。而且用这个方法来制作属性页就可以利用ClassWizard来轻松地生成消息映射处理Tab Control的消息了。例如：可以处理TCN_SELCHANGE消息来对切换了标签时进行一些动作。

方案五另一写法

思路:当我们调用InsertItem()这个函数的时候,选项卡控件将会添加一个标签页,这个时候,我们将自己的对话框的窗体的指针与此标签页关联起来,当用户进行标签页的切换的时候,我们根据当前是哪个标签页,显示哪个对话框,不是与当前标签页关联的对话框,我们将其隐藏即可.这样我们便可以实现选项卡控件.

第一步:新建一个自己的类CTabSheet继承CTabCtrl.

第二步:定义有用的成员变量

CDialog* m_dlgWnd[MAXTABPAGE]; //这个是存放对话框指针的指针数组
int m_curTabNumber; //记录当前用户添加了几个标签页
int m_selTabID; //当前用户点击的标签页的ID
第三步:添加成员函数
//通过这个函数,可以将一个对话框指针与添加的标签页关联起来,insWnd是创建的非模式对话框的指针,wndID是对话框的ID,pageText是标签页的标题
void CreateTabPage(CWnd *insWnd, int wndID,CString pageText)
//添加控件的点击事件的处理,当点击后得到当前点击的标签页的ID,然后将与此标签页相关的对话框显示,其它的隐藏即可
void OnLButtonDown(UINT nFlags, CPoint point)
通过添加以上的成员变量及成员函数即可实现一个简单的选项卡控件的用法
下面我将这两个成员函数的代码贴出来,并详细讲解

```cpp
        //创建并且增加一个标签页
    //创建并且增加一个标签页
void CTabSheet::CreateTabPage(CWnd *insWnd, int wndID,CString pageText)
{
     if (m_curTabNumber >= MAXTABPAGE)
     {
         MessageBox("标签页已经达到最大!","创建出错!",MB_OK);
         return;
     }
        //首先new一个对话框的指针,但是不要调用create函数,再将些指针当成参数传进来即可,创建已由此函数做完
     if (NULL == insWnd)
     {
         MessageBox("标签页为空","创建出错",MB_OK);
         return;
     }
                //创建对话框,并且增加标签页
     CDialog* curDlg = (CDialog*)insWnd;
     curDlg->Create(wndID,this);
     int suc = InsertItem(m_curTabNumber,pageText);
     if (-1 == suc)
     {
         MessageBox("插入标签页失败","失败",MB_OK);
         return;
     }
     curDlg->ShowWindow(SW_SHOW);
     //将这个对应的窗体指针存放起来
     m_dlgWnd[m_curTabNumber] = curDlg;
     //此时选择当前页面
     SetCurSel(m_curTabNumber);
     m_selTabID = m_curTabNumber;
     m_curTabNumber ++;
}
//点击左键事件,处理
void CTabSheet::OnLButtonDown(UINT nFlags, CPoint point)
{
    // TODO: Add your message handler code here and/or call default
    CTabCtrl::OnLButtonDown(nFlags, point);
    //得到当前用户点击的标签页的ID
    int curSelect = GetCurSel();
    //得到当前标签页的位置以便设置对话框显示的位置
    CRect curRect;
    GetClientRect(curRect);
    if (-1 == curSelect)
    {
        return;
    }
            //查找标签页,将与当前用户点击的标签页相关的对话框显示出来,其它的对话框隐藏
    for (int i = 0; i < m_curTabNumber; i ++)
    {
        if (i == curSelect)
        {
                                    m_dlgWnd[i]->SetWindowPos(NULL,0,20,curRect.Width(),curRect.bottom,SWP_SHOWWINDOW);
        }
        else
        {
            m_dlgWnd[i]->SetWindowPos(NULL,0,20,curRect.Width(),curRect.bottom,SWP_HIDEWINDOW);
        }
    }
    m_selTabID = curSelect;
    Invalidate();
    //CTabCtrl::OnLButtonDown(nFlags, point);
}
```

以上为关键的两个函数,下面介绍调用的方法
创建非模式的对话框
CTabSheet m_tabSheet;
CMyDlg* m_dlg = new CMyDlg;
m_tabSheet.CreateTabPage(m_dlg ,IDD_DLG_ID,"第一个标签页");
这样就可以产生一个标签页了,当然还可以继续调用此函数添加标签页

文章标签： MFC 选项卡 继承 实例

个人分类： 纯编程