**第7课 对话框**

**1 对话框的创建**

* **模态对话框**

资源添加对话框，双击对话框创建对话框类。

在menu中增加对话框菜单项，将其设为非popup的，并为其在view类下添加响应函数。View类中增加#include "TestDlg.h"。

在响应函数void CMyboleView::OnDialog()中：

CTestDlg dlg;

dlg.DoModal();

模态对话框即程序运行到此处被阻滞，停在此处，不可以执行其他内容。

* **非模态对话框**

CTestDlg m\_dlg;

m\_dlg .Create (IDD\_DIALOG1,this); //this为对话框父窗口的指针

m\_dlg .ShowWindow (SW\_SHOW);

注意：这里非模态对话框要设置为成员变量。另外非模态对话框点击非模态对话框的OnOK按纽时，它并没有关闭，而是隐藏了。需要调用destroyWindow()。

* **在对话框上动态创建一个button对象**

在对话框资源中从工具中拖一个button到对话框中，将其名称改为ADD，为其在CTestDlg类中添加响应函数。

void CTestDlg::OnBnClickedBtnAdd()

{

if(m\_blsCreate==false) // m\_blsCreate是bool型成员变量判断是否已经创建

{

m\_btn.Create (\_T("维新

"),BS\_DEFPUSHBUTTON|WS\_VISIBLE|WS\_CHILD,

CRect(0,0,100,100),this,123); //动态创建button，其中m\_btn为CBtton类型的成员变量

m\_blsCreate=true;

}

else

{

m\_btn.DestroyWindow ();

m\_blsCreate=false;

}

}

另外m\_blsCreate可以不定义为成员变量，而定义为static bool blsCreate=false;

每个窗口都有一个m\_hWnd的成员变量，当窗口被创建后m\_hWnd为非零值，当窗口被销毁后（窗口类对象仍然存在）值为零，因此可以通过判断m\_hWnd的方式来判断button对象是否被创建。

void CTestDlg::OnBnClickedBtnAdd()

{

if(!m\_btn.m\_hWnd )

{

m\_btn.Create (\_T("维新"), BS\_DEFPUSHBUTTON|WS\_VISIBLE|WS\_CHILD, CRect(0,0,100,100),this,123);

}

else

{

m\_btn.DestroyWindow ();

}

}

**2 对话框加法操作**

* **静态文本框**

在对话框资源文件中从工具箱拖出三个静态文本框Static Text，将其名称改为Number1，Number2，Number3，将ID号改为IDC\_NUMBER1, IDC\_NUMBER2, IDC\_NUMBER3,将Notify项设为True，如果不这么设置则无法接受相应消息。

在CTestDlg类中给文本框添加点击响应函数：

CString str;

if(GetDlgItem(IDC\_NUMBER1)->GetWindowText(str),str==\_T("Number1"))

{

GetDlgItem(IDC\_NUMBER1)->SetWindowText(\_T("数值："));

}

else

{

GetDlgItem(IDC\_NUMBER1)->SetWindowText(\_T("Number1"));

}

注释：GetDlgItem(IDC\_NUMBER1)通过对话框中的控件ID获取控件句柄Cwnd\*。

GetWindowText(str) 拷贝控件的名称到str。

SetWindowText(\_T("数值："))设置控件名称。

* **对话框实现加法操作**

在CTestDlg::OnBnClickedBtnAdd()函数中添加如下响应函数：

方法一：使用GetDlgItem(),GetWindowText(),SetWindowText()

int num1,num2,num3;

char ch1[10],ch2[10],ch3[10];

GetDlgItem(IDC\_EDIT1)->GetWindowText(ch1,10); //获取文本的最大个数是10

GetDlgItem(IDC\_EDIT2)->GetWindowText(ch2,10);

num1=atoi(ch1);

num2=atoi(ch2);

num3=num1+num2;

itoa(num3,ch3,10);

GetDlgItem(IDC\_EDIT3)->SetWindowText(ch3);

注释：atoi(ch1)将char\*类型转换为int，itoa将int转换为char\*，最后一个参数为转换进制数。

方法二：使用GetDlgItemText()和SetDlgItemText()

int num1,num2,num3;

char ch1[10],ch2[10],ch3[10];

GetDlgItemText(IDC\_EDIT1,ch1,10);

GetDlgItemText(IDC\_EDIT2,ch2,10);

num1=atoi(ch1);

num2=atoi(ch2);

num3=num1+num2;

itoa(num3,ch3,10);

SetDlgItemText(IDC\_EDIT3,ch3);

注释：GetDlgItemText()相当于GetDlgItem(),GetWindowText()的组合。

方法三：使用GetDlgItemInt()，SetDlgItemInt()

int num1,num2,num3;

num1=GetDlgItemInt(IDC\_EDIT1);

num2=GetDlgItemInt(IDC\_EDIT2);

num3=num1+num2;

SetDlgItemInt(IDC\_EDIT3,num3);

注释：GetDlgItemInt(

**int** *nIDDlgItem***,** // 控件id

**BOOL** **\****lpTranslated***,** // success state ，默认为NULL，不警告任何错误

**BOOL** *bSigned* // 有符号或无符号，默认为真有符号

**);**

BOOL SetDlgItemInt(int nIDDlgItem,UINT uValue,BOOL bSigned);

方法四：关联成员变量法

方法五：关联控件变量

方法六：发送消息SendMessage

方法七：发送消息SendDlgItemMessage

**3 关联成员变量**

资源文件中控件上右击添加变量，private，变量类型int，变量名：m\_num1,控件类别将control改为value。

可以看到在代码中增加了如下内容：

CTestDlg.h中：

private:

// 控件关联成员变量

int m\_num1;

int m\_num2;

int m\_num3;

CTestDlg.cpp中：

void CTestDlg::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Text(pDX, IDC\_EDIT1, m\_num1);

DDX\_Text(pDX, IDC\_EDIT2, m\_num2);

DDX\_Text(pDX, IDC\_EDIT3, m\_num3);

}

在CTestDlg::OnBnClickedBtnAdd()函数中添加如下响应函数：

UpdateData();

m\_num3=m\_num1+m\_num2;

UpdateData(false);

注释： UpdateData()参数只有一个，默认为TRUE。为TRUE时刷新控件的值到对应的变量(外部输入值交给内部变量)。为false时拷贝变量值到控件显示(变量的最终运算结果值交给外部输出显示)。

* **设定值的范围**

在函数DoDataExchange中DDX\_Text(pDX, IDC\_EDIT1, m\_num1);语句后添加：

DDV\_MinMaxInt(pDX,m\_num1,0,100); //m\_num限定在0到100

**4 关联控件变量**

资源文件中控件上右击添加变量，private，变量类型CEdit，变量名：m\_edit1,控件类别为control。

可以看到在代码中增加了如下内容：

CTestDlg.h中：

private:

// 控件关联成员变量

CEdit m\_edit1;

CEdit m\_edit2;

CEdit m\_edit3;

CTestDlg.cpp中：

void CTestDlg::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Control(pDX, IDC\_EDIT1, m\_edit1);

DDX\_Control(pDX, IDC\_EDIT2, m\_edit2);

DDX\_Control(pDX, IDC\_EDIT3, m\_edit3);

}

由于CEdit类型也是从Cwnd继承的，所以在CTestDlg::OnBnClickedBtnAdd()函数中添加如下响应函数：

int num1,num2,num3;

char ch1[10],ch2[10],ch3[10];

m\_edit1.GetWindowText(ch1,10);

m\_edit2.GetWindowText(ch2,10);

num1=atoi(ch1);

num2=atoi(ch2);

num3=num1+num2;

itoa(num3,ch3,10);

m\_edit3.SetWindowTextA (ch3);

**5 发送消息**

**使用SendMessage函数WM\_GETTEXT和WM\_SETTEXT消息**

在CTestDlg::OnBnClickedBtnAdd()函数中添加如下响应函数：

int num1,num2,num3;

char ch1[10],ch2[10],ch3[10];

//::SendMessageA(GetDlgItem(IDC\_EDIT1)->m\_hWnd , WM\_GETTEXT ，10, (LPARAM)ch1 );

//::SendMessageA(m\_edit1->m\_hWnd , WM\_GETTEXT ,10,(LPARAM)ch1);

//GetDlgItem(IDC\_EDIT1)->SendMessage (WM\_GETTEXT,10,(LPARAM)ch1);

m\_edit1.SendMessage (WM\_GETTEXT,10,(LPARAM)ch1);

m\_edit2.SendMessage (WM\_GETTEXT,10,(LPARAM)ch2); //第二个参数为发送数据量，第三个为接受数据的buffer

num1=atoi(ch1);

num2=atoi(ch2);

num3=num1+num2;

itoa(num3,ch3,10);

m\_edit3.SendMessage (WM\_SETTEXT,0,(LPARAM)ch3);

**使用SendDlgItemMessage函数**

在CTestDlg::OnBnClickedBtnAdd()函数中添加如下响应函数：

int num1,num2,num3;

char ch1[10],ch2[10],ch3[10];

SendDlgItemMessage(IDC\_EDIT1,WM\_GETTEXT,10,(LPARAM)ch1);

SendDlgItemMessage(IDC\_EDIT2,WM\_GETTEXT,10,(LPARAM)ch2);

num1=atoi(ch1);

num2=atoi(ch2);

num3=num1+num2;

itoa(num3,ch3,10);

SendDlgItemMessage(IDC\_EDIT3,WM\_SETTEXT,10,(LPARAM)ch3);

**使用EM\_SETSEL消息设置复选信息**

在上面代码后面加上：

SendDlgItemMessage(IDC\_EDIT3,EM\_SETSEL,1,3);

m\_edit3.SetFocus ();

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

SendDlgItemMessage开始位置为0，结束位置为-1，可将整个框内数据复选上。

**6 收缩扩展功能**

* **添加分割线**

在对话框窗口资源上加一个picture control画一条线，更改其属性，他的ID是static的，不能接受notify消息，将其改为IDC\_SEPARATOR，Sunken设置为true，visiable设置为false。

* **添加button控制收缩**

增加一个button，命名为”收缩<<”，并对其添加消息处理函数。

void CTestDlg::OnBnClickedButton2()

{

CString str;

if(GetDlgItemText(IDC\_BUTTON2,str),str=="收缩《")

{

SetDlgItemText(IDC\_BUTTON2,"扩展》");

}

else

{

SetDlgItemText(IDC\_BUTTON2,"收缩《");

}

static CRect rectLarge;

static CRect rectSmall;

if(rectLarge.IsRectNull())

{

CRect rectSeparator;

GetWindowRect(&rectLarge);

GetDlgItem(IDC\_SEPARATOR)->GetWindowRect(&rectSeparator);

rectSmall.left =rectLarge.left ;

rectSmall.top=rectLarge.top;

rectSmall.right =rectLarge.right;

rectSmall.bottom=rectSeparator.top ;

}

if(str=="收缩《")

{

SetWindowPos(NULL,0,0,rectSmall.Width (),rectSmall.Height(),

SWP\_NOMOVE|SWP\_NOZORDER); //重新设置大小

}

else

{

SetWindowPos(NULL,0,0,rectLarge.Width (),rectLarge.Height(),

SWP\_NOMOVE|SWP\_NOZORDER);

}

}

**7 按下回车键自动转到下一个输入框**

Ok按钮是缺省的按钮，属性中Default button为true。当按下回车自动相当于按下Ok键。Ok的ID号为IDOK，如果自己创建一个按钮ID号为IDC\_OK，虽然响应函数都为OnOk但不同。另外OK按钮不管存不存在对话框上，其响应都会存在。

增加ok消息响应函数，在其中屏蔽对基类CDialogEx::OnOK();的调用。

SetWindowLong()改变窗口的属性。

**方法1：**

改变控件的回调函数，注意IDC\_EDIT1的MultiLine要复选上，这样才能响应按键操作。

WNDPROC prevProc;

LRESULT CALLBACK WinSunProc(

HWND hwnd, // handle to window

UINT uMsg, // message identifier

WPARAM wParam, // first message parameter

LPARAM lParam // second message parameter

)

{

if(uMsg==WM\_CHAR && wParam==0x0d)

{

//::SetFocus(::GetNextWindow(hwnd,GW\_HWNDNEXT));

//SetFocus(::GetWindow(hwnd,GW\_HWNDNEXT));

CString str;

str.Format("%d",hwnd);

AfxMessageBox(str);//, UINT nType = MB\_OK, UINT nIDHelp = 0 );

// AfxGetApp()->

SetFocus(::GetNextDlgTabItem(::GetParent(hwnd),hwnd,FALSE));

return 1;

}

else

{

return prevProc(hwnd,uMsg,wParam,lParam);

}

}

BOOL CTestDlg::OnInitDialog()

{

CDialog::OnInitDialog();

// TOD Add extra initialization here

prevProc=(WNDPROC)SetWindowLong(GetDlgItem(IDC\_EDIT1)->m\_hWnd,GWL\_WNDPROC,

(LONG)WinSunProc);//设置回调函数

return TRUE; // return TRUE unless you set the focus to a control

// EXCEPTION: OCX Property Pages should return FALSE

}

**方法2：**

在OnOK响应函数中加入代码

//GetDlgItem(IDC\_EDIT1)->GetNextWindow()->SetFocus();

//GetFocus()->GetNextWindow()->SetFocus();

//GetFocus()->GetWindow(GW\_HWNDNEXT)->SetFocus();

GetNextDlgTabItem(GetFocus())->SetFocus(); //焦点可以依次传递，查找完一遍不会出问题