7.16

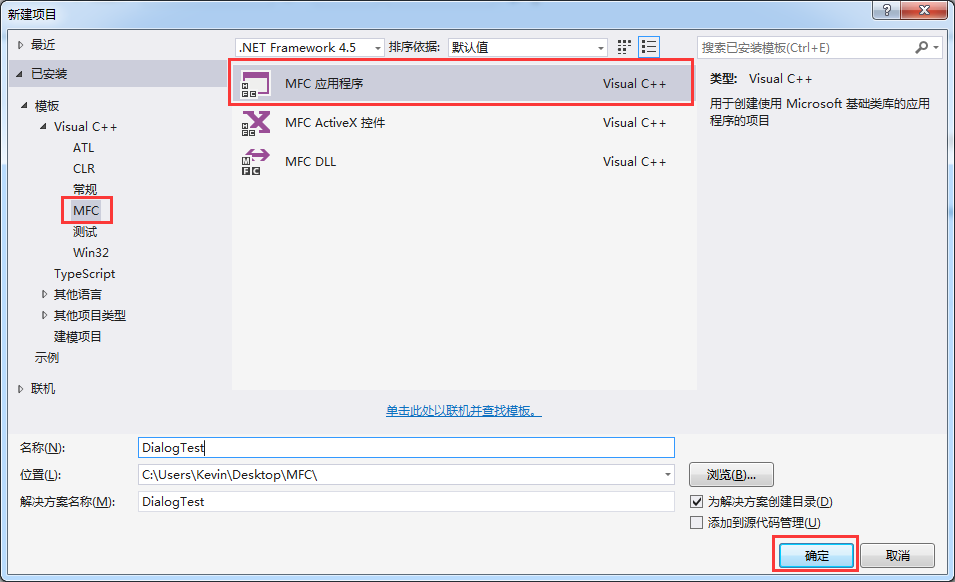
基于对话框编程

对话框是一种特殊类型的窗口，绝大多数Windows程序都通过对话框与用户进行交互。在Visual C++中，对话框既可以单独组成一个简单的应用程序，又可以成为文档/视图结构程序的资源。

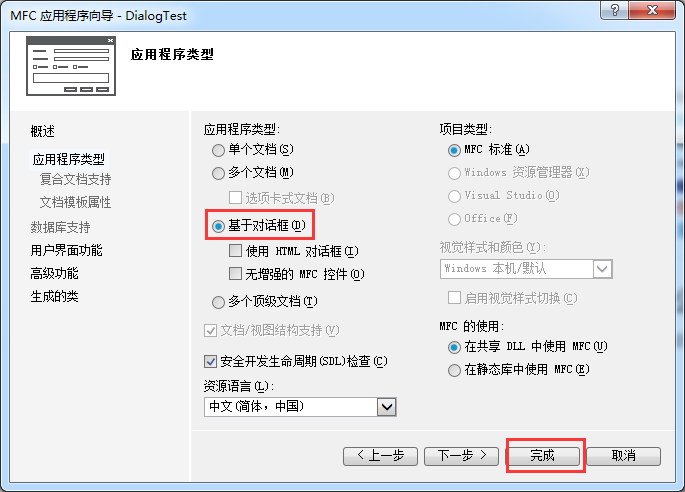
2.1 创建基于对话框的 MFC 应用程序框架

程序的创建过程：

* 选择“文件 | 新建 |　项目”菜单；
* 在“新建项目”对话框中，选择“ MFC 应用程序 ”，输入工程名称，选择“确定”。



* 选择“ 基于对话框”，即创建基于对话框的应用程序，选择“完成”。



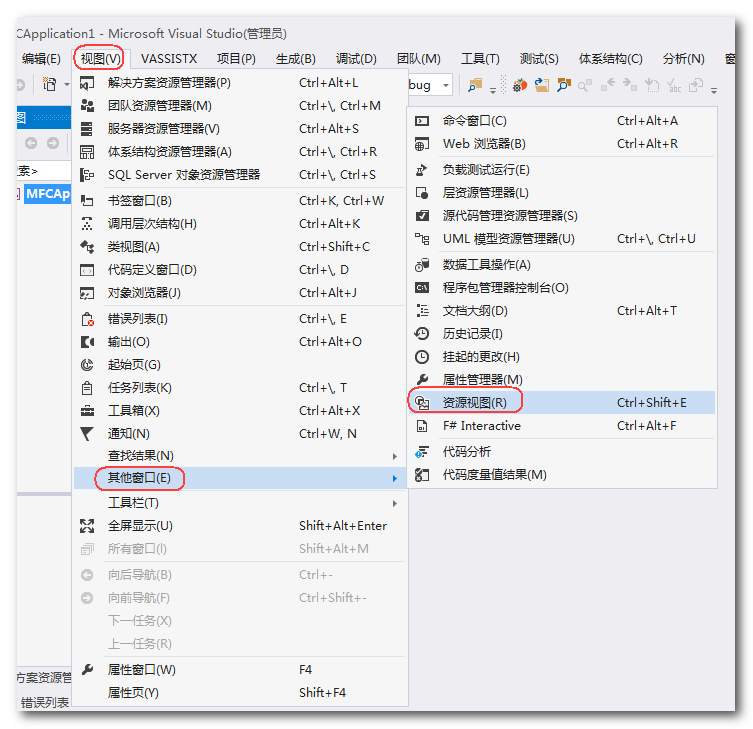
2.2 对话框应用程序框架介绍

2.2.1 资源视图

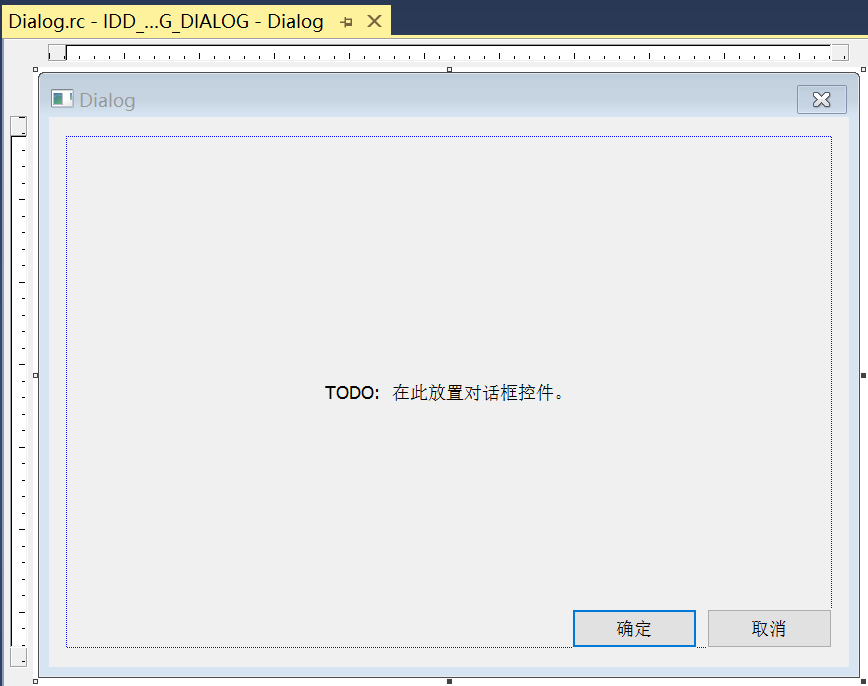
用 AppWizard 创建基于对话框的应用程序框架（假定工程名为 Dialog ）后，项目工作区上增加了一个“资源视图”选项卡。



或者，通过视图找到“资源视图”选项卡：



在 MFC中，与用户进行交互的对话框界面被认为是一种资源。展开“Dialog”，可以看到有一个ID为IDD\_ DIALOG \_DIALOG（中间部分（DIALOG）与项目名称相同）的资源，对应中间的对话框设计界面。不管在何时，只要双击对话框资源的ID，对话框设计界面就会显示在中间。



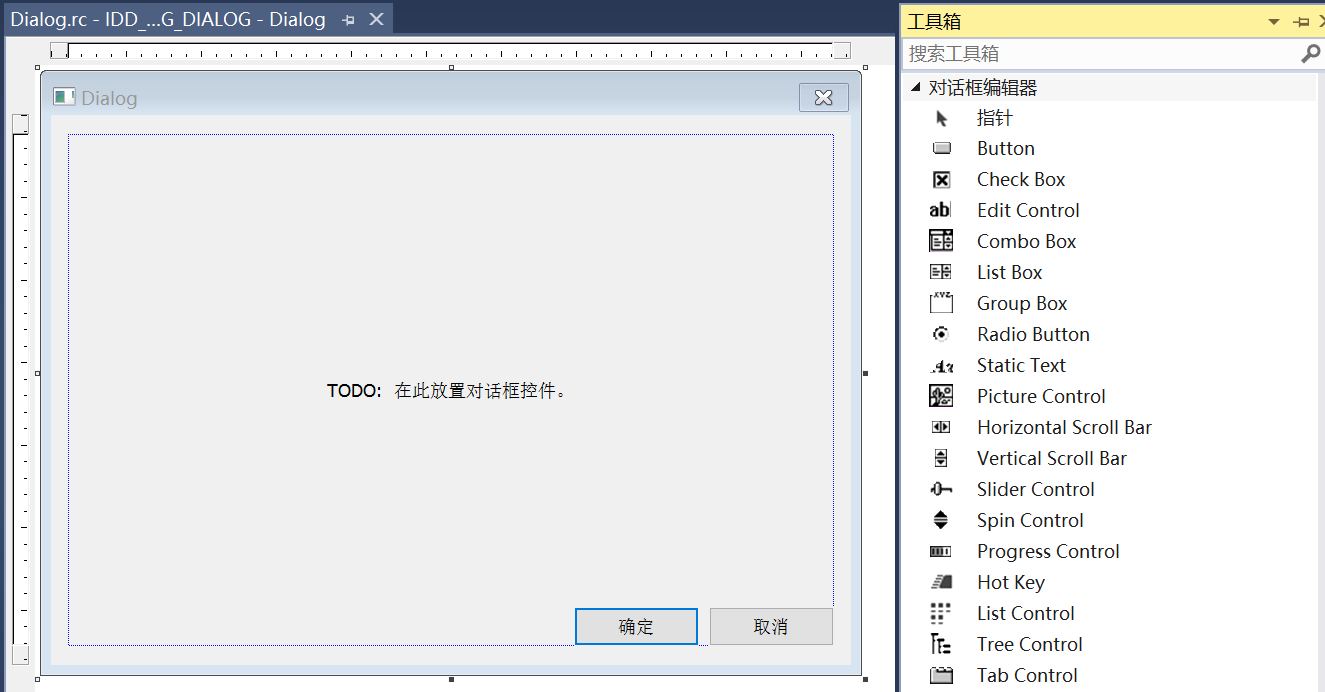
2.2.2 类视图

在类视图中，可以看到生成了3 个类：CAboutDlg、CDialogApp和CDialogDlg。



* **CAboutDlg**：对应生成的版本信息对话框。
* **CDialogApp**：应用程序类，从 CWinApp 继承过来，封装了初始化、运行、终止该程序的代码。
* **CDialogDlg**：对话框类，从CdialogEx继承过来的，在程序运行时看到的对话框就是它的一个具体对象。
  + **DoDataExchange函数**：该函数主要完成对话框数据的交换和校验。
  + **OnInitDialog函数**：相当于对对话框进行初始化处理。

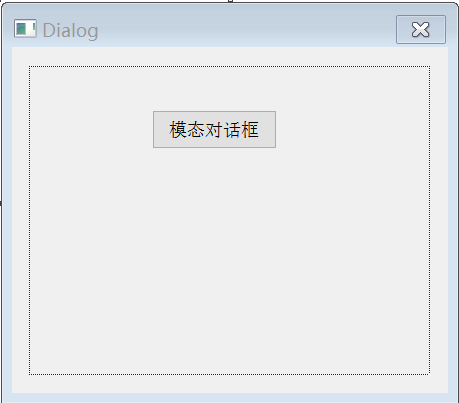
2.2.3 设计界面和工具箱



2.3 模态对话框

当模态对话框显示时，程序会暂停执行，直到关闭这个模态对话框之后，才能执行程序中的其他任务。

1）通过工具箱在界面上放一个Button，双击此按钮即可跳转到按钮处理函数：



//按钮处理函数

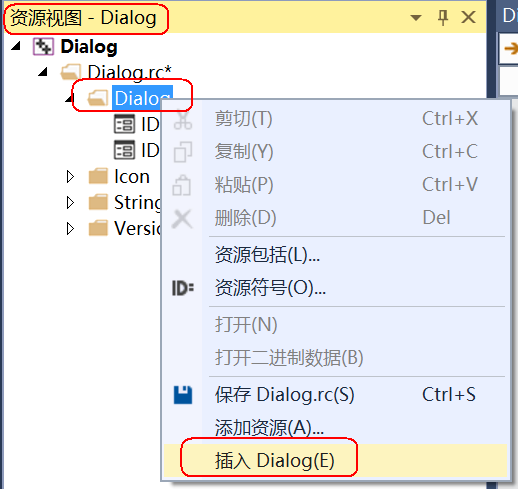
void CDialogDlg::OnBnClickedButton1()

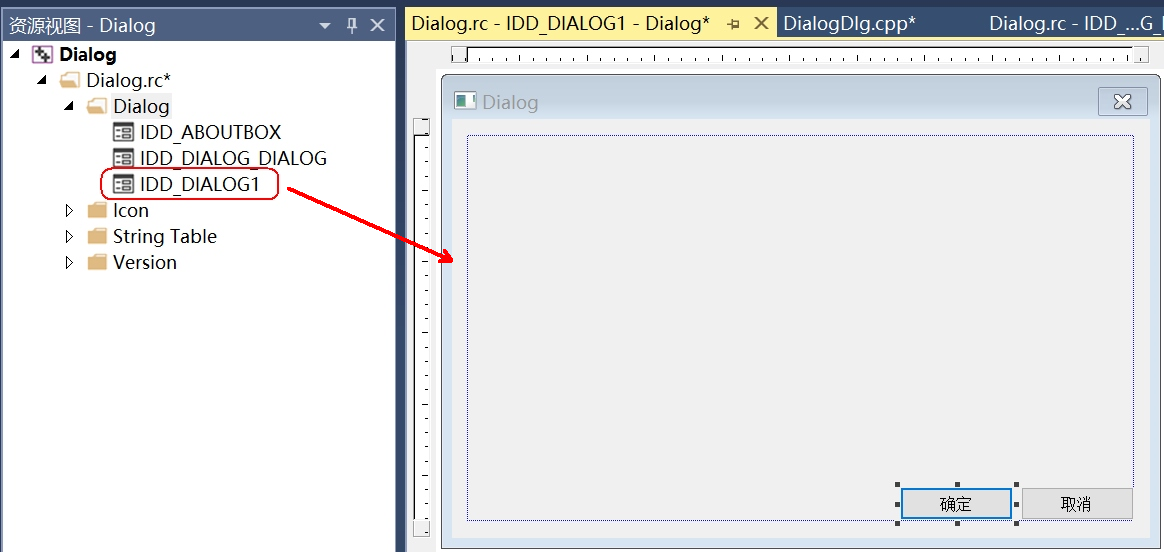
{

// TODO: 在此添加控件通知处理程序代码

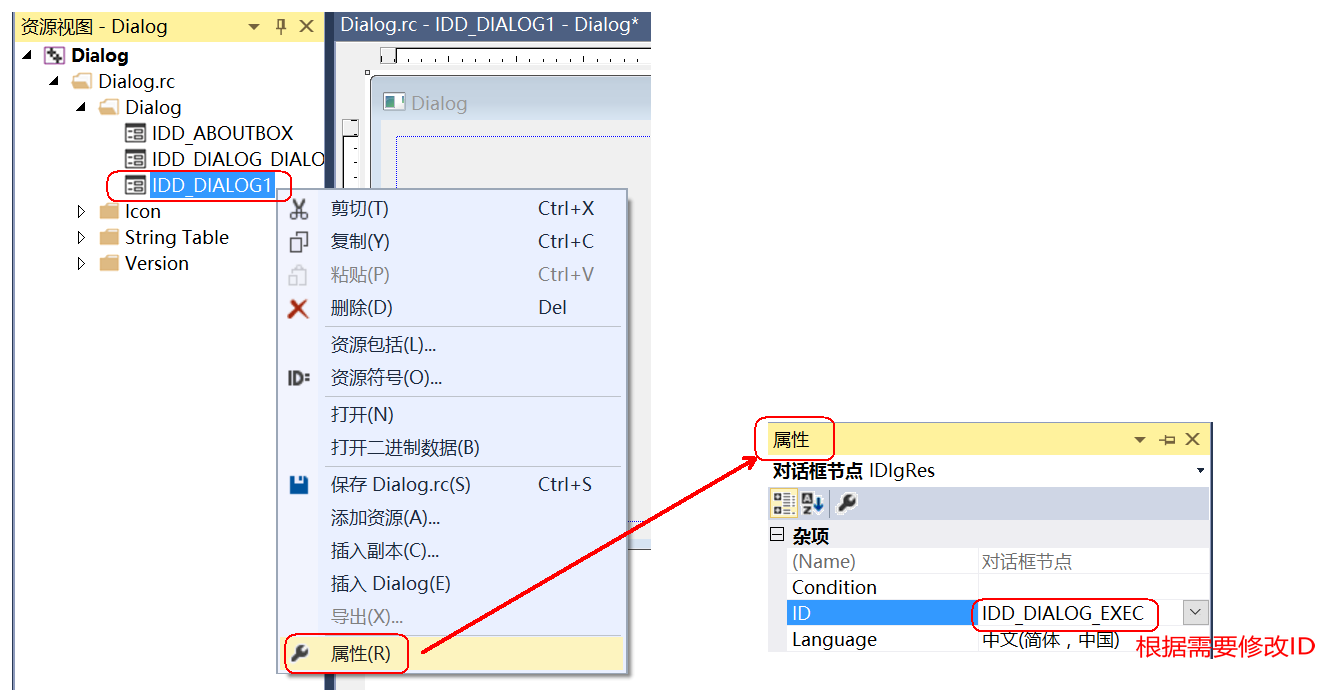
}

2）资源视图 -> Dialog -> 右击 -> 插入 Dialog：

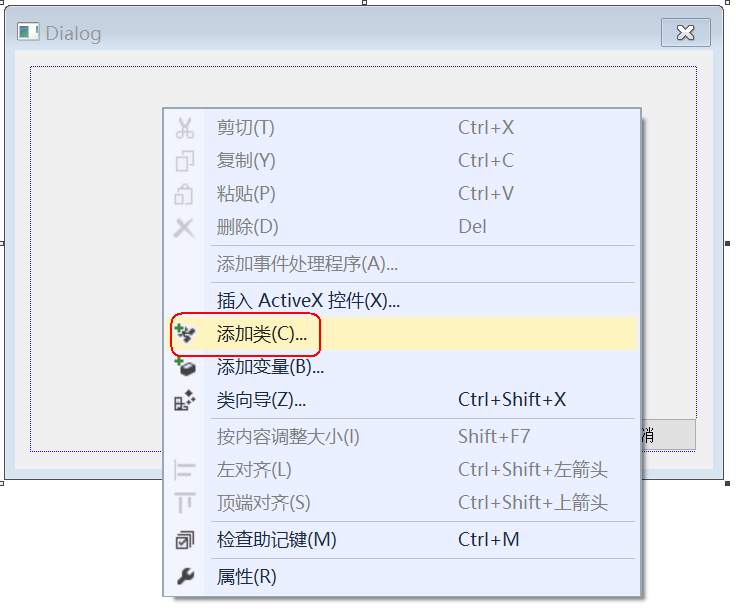


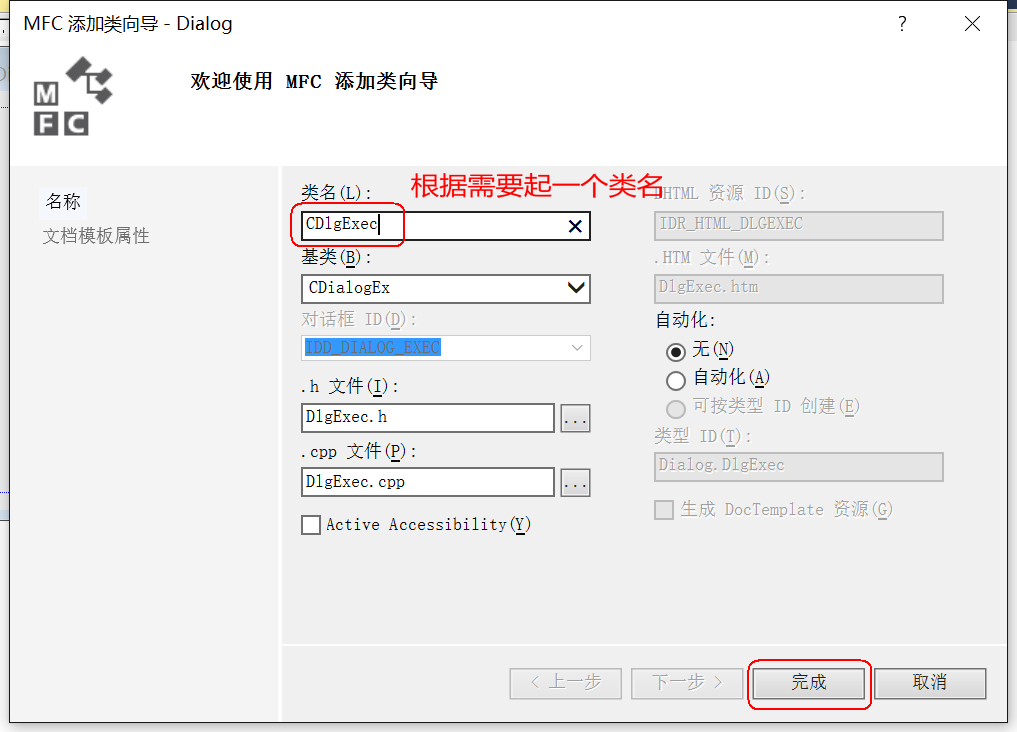


1. 修改对话框ID:

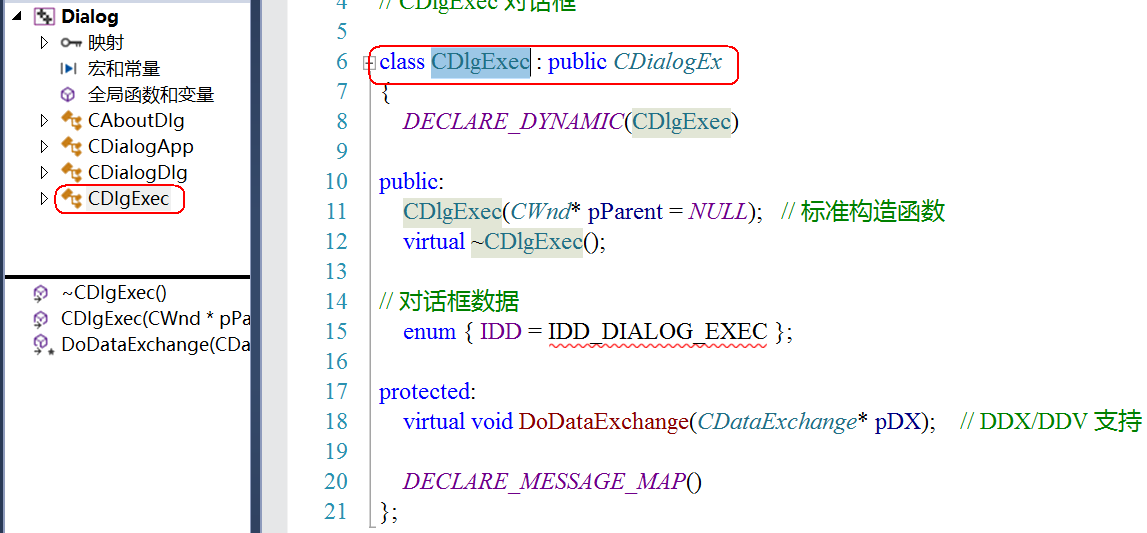


1. 点击对话框模板 -> 右击 -> 添加类：





1. 类视图中多了一个自定义类：



1. 按钮处理函数创建对话框，以模态方式运行。

实现模态对话框的创建需要调用CDialog类的成员函数CDialog::DoModel，该函数的功能就是创建并显示一个对话框:

//启动模态对话框按钮

void CDialogDlg::OnBnClickedButton1()

{

//需要包含头文件：#include "DlgExec.h"

CDlgExec dlg;

dlg.*DoModal*(); //以模态方式运行

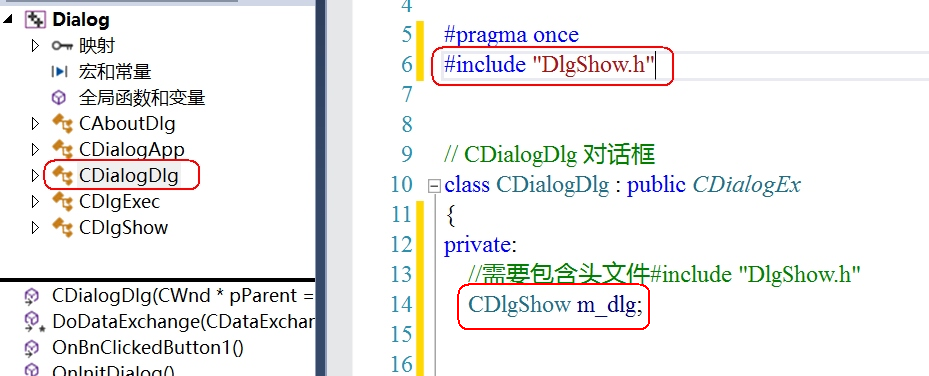
}

2.4 非模态对话框

当非模态对话框显示时，运行转而执行程序中的其他任务，而不用关闭这个对话框。

图形界面操作过程和模态对话框一样，只是，非模态对话框实现方式不一样，先创建(CDialog::Create)一次，然后再显示(CWnd::ShowWindow)。

1. 主对话框.h类中声明对话框对象：



1. 创建对话框放在主对话框类的构造函数或OnCreate()函数，目的只创建一次对话框：

//主对话框构造函数

CDialogDlg::CDialogDlg(*CWnd*\* pParent /\*=NULL\*/)

: *CDialogEx*(CDialogDlg::IDD, pParent)

{

m\_hIcon = *AfxGetApp*()->*LoadIcon*(IDR\_MAINFRAME);

m\_dlg.*Create*(IDD\_DIALOG\_SHOW); //IDD\_DIALOG\_SHOW为对话框ID

}

1. 按钮处理函数显示对话框：

//启动非模态对话框按钮

void CDialogDlg::OnBnClickedButton2()

{

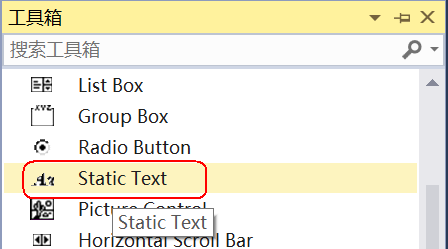
// TODO: 在此添加控件通知处理程序代码

m\_dlg.*ShowWindow*(*SW\_SHOWNORMAL*); //显示非模态对话框

}

3. 常用控件

3.1 静态文本框CStatic



静态文本框是最简单的控件，它主要用来显示文本信息，不能接受用户输入，一般不需要连接变量，也不需要处理消息。

静态文本框的重要属性有：

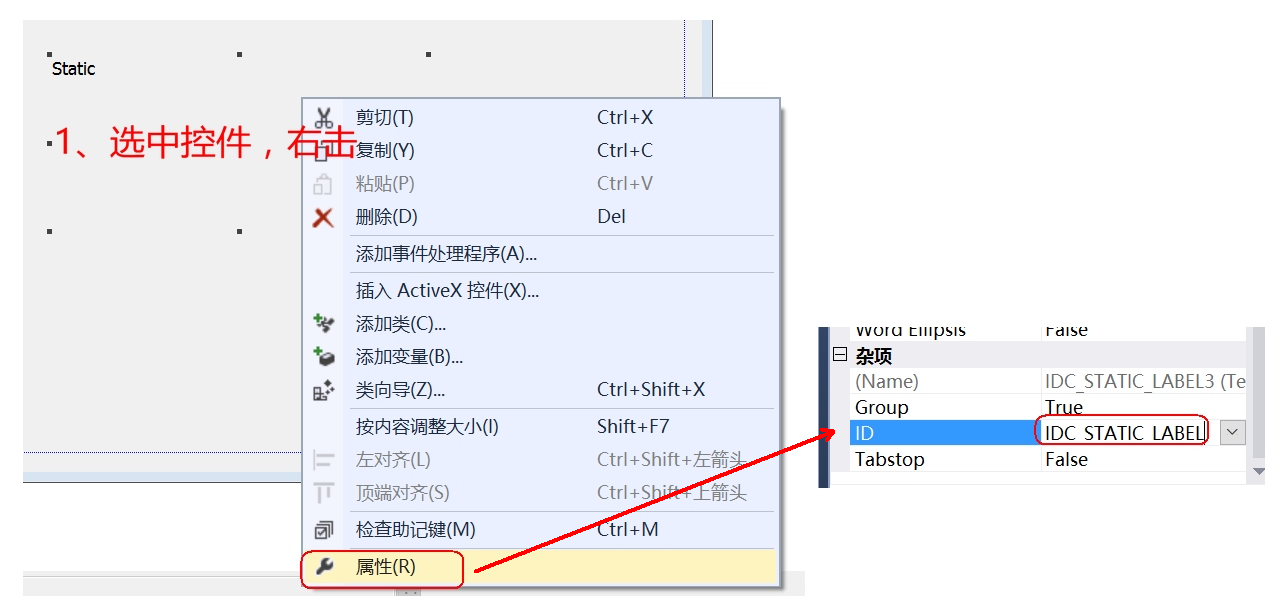
* ID：所有静态文本框的缺省ID都是IDC\_STATIC，静态ID，不响应任何消息（事件）
* Caption：修改显示的内容

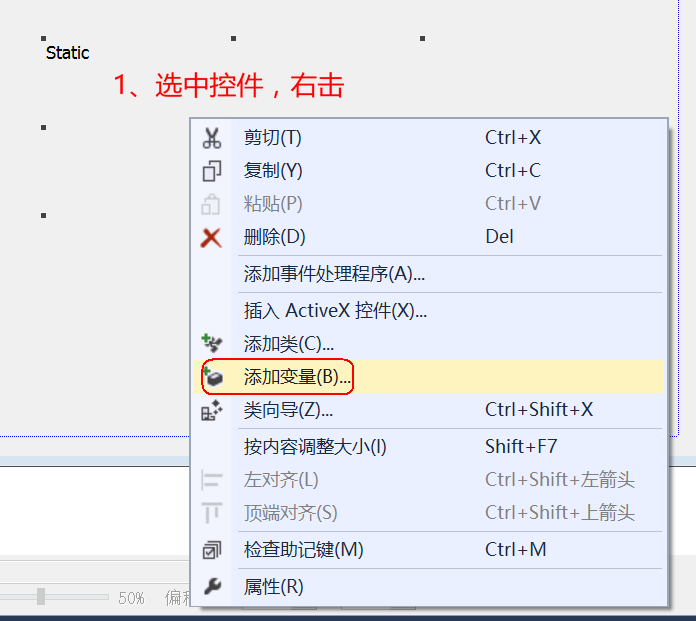
常用接口：

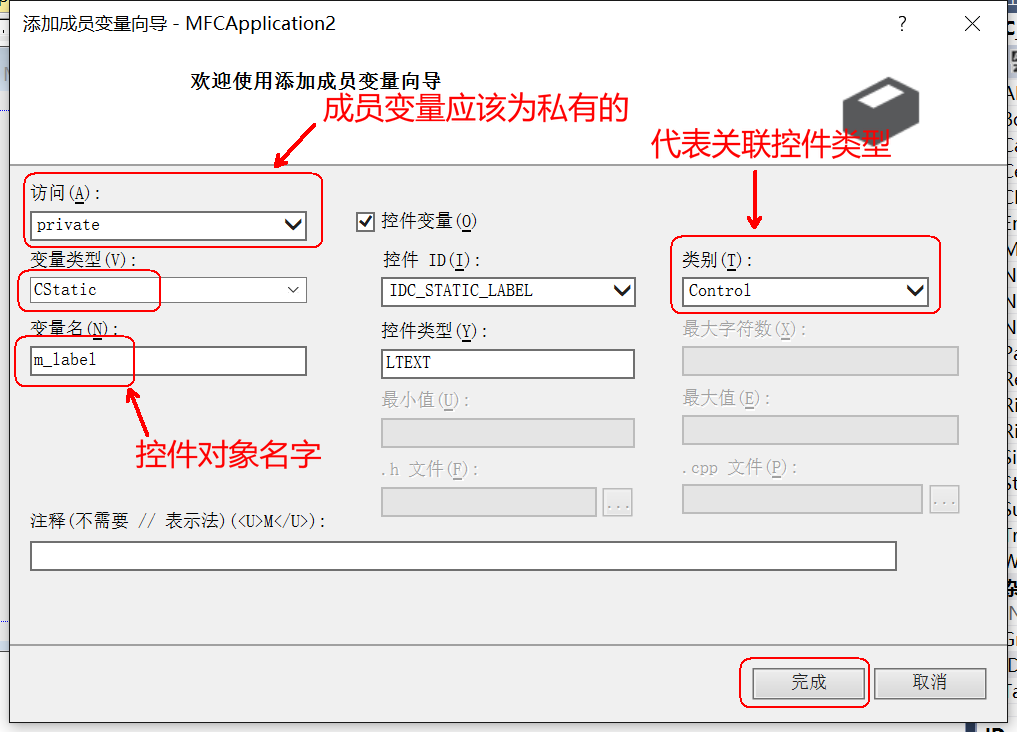
| **接口** | **功能** |
| --- | --- |
| CWnd::SetWindowText | 设置控件内容 |
| CWnd::GetWindowText | 获取控件内容 |
| CStatic::SetBitmap | 设置位图(后缀为bmp的图片) |

**关联控件变量：**

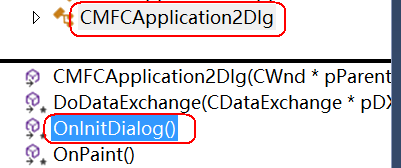
由于XXX\_STATIC静态ID是不能关联变量，故需把ID修改后，再关联变量：







在主对话框类OnInitDialog()中，完成相应接口测试：



//设置静态控件内容为Tom

m\_label.*SetWindowText*(*TEXT*("Tom"));

//获取静态控件的内容

*CString* str;

m\_label.*GetWindowText*(str);

*MessageBox*(str);

//设置静态控件窗口风格为位图居中显示

m\_label.*ModifyStyle*(0xf, *SS\_BITMAP* | *SS\_CENTERIMAGE*);

//通过路径获取bitmap句柄

#define HBMP(filepath,*width*,*height*) (*HBITMAP*)*LoadImage*(*AfxGetInstanceHandle*(),filepath,*IMAGE\_BITMAP*,*width*,*height*,*LR\_LOADFROMFILE*|*LR\_CREATEDIBSECTION*)

//静态控件设置bitmap

m\_label.*SetBitmap*(HBMP(*TEXT*("./1.bmp"), 300, 250));