# VS2010-MFC编程入门之前言

鸡啄米的C++[编程入门](http://www.jizhuomi.com/catalog.asp?tags=%E7%BC%96%E7%A8%8B%E5%85%A5%E9%97%A8)系列给大家讲了C++的编程入门知识，大家对C++语言在语法和设计思想上应该有了一定的了解了。但是教程中讲的例子只是一个个简单的例程，并没有可视化窗口。鸡啄米在这套VS2010/MFC编程入门教程中将会给大家讲解怎样使用VS2010进行可视化编程，也就是基于窗口的程序。

C++编程入门系列主要偏重于理论方面的知识，目的是让大家打好底子，练好内功，在使用VC++编程时不至于丈二和尚摸不着头脑。本套教程也会涉及到VC++的原理性的东西，同样更重视实用性，让大家学完本套教程以后，基本的界面程序都能很容易编写出来。

## **VC++简介**

VC++全称是Visual C++，是由微软提供的C++开发工具，它与C++的根本区别就在于，C++是语言，而VC++是用C++语言编写程序的工具平台。VC++不仅是一个编译器更是一个集成开发环境，包括编辑器、调试器和编译器等，一般它包含在Visual Studio中。Visual Studio包含了VB、VC++、C#等编译环境。当然我们在使用VC++6.0的时候为了轻便，总是只单独安装VC++ 6.0。但自微软2002年发布Visual Studio.NET以来，微软建立了在.NET框架上的代码托管机制，一个项目可以支持多种语言开发的组件，VC++同样被扩展为支持代码托管机制的开发环境，所以.NET Framework是必须的，也就不再有VC++的独立安装程序，不过可以在安装Visual Studio时只选择VC++进行安装。

## **VC++版本的选择:VS2010**

因为VC++ 6.0以后的版本不再有独立的安装程序，所以鸡啄米在教程中将不会称VC++ 6.0以后的版本为VC++ 7.0等等，而是用VC++所属的Visual Studio的版本名称代替，比如VS2003。

近些年VC++主要的版本包括：VC++ 6.0、VS2003、VS2005、VS2008和[VS2010](http://www.jizhuomi.com/software/27.html)。

VC++ 6.0占用的系统资源比较少，打开工程、编译运行都比较快，所以赢得很多软件开发者的青睐。但因为它先于C++标准推出，所以对C++标准的支持不太好。举个例子：

1. for(int i=0; i<5; i++)
2. {
3. a[i] = i;
4. }

for语句中声明的变量i，对于VC++ 6.0来说，出了for循环仍能使用。但很显然这与C++标准对于变量[生存期](http://www.jizhuomi.com/software/60.html)的规定不符合。

随着VC++版本的更新，对C++标准的支持越来越好，对各种技术的支持也越来越完善。但同时新版本所需的资源也越来越多，对处理器和内存的要求越来越高。到VS2010，光安装文件就2G多，安装后的文件占3G多空间，其运行也经常受处理器和内存等性能的限制。但鸡啄米还是推荐大家使用VS2010，毕竟它是最新版本，类库和开发技术都是最完善的，本教程也将使用VS2010为大家做例程的演示。当然如果系统配置确实比较低，可以选择VS2005，VS2005和VS2010相比还是要轻量级一些的。VC++ 6.0已经过时，奉劝大家尽量别用了。

## **VC++与MFC**

讲VC++免不了要提MFC，MFC全称Microsoft Foundation Classes，也就是微软基础类库。它是VC++的核心，是C++与Windows API的结合，很彻底的用C++封装了Windows SDK（Software Development Kit，软件开发工具包）中的结构和功能，还提供了一个应用程序框架，此应用程序框架为软件开发者完成了一些例行化的工作，比如各种窗口、工具栏、菜单的生成和管理等，不需要开发者再去解决那些很复杂很乏味的难题，比如每个窗口都要使用Windows API注册、生成与管理。这样就大大减少了软件开发者的工作量，提高了开发效率。

当然VC++不是只能够创建MFC应用程序，同样也能够进行Windows SDK编程，但是那样的话就舍弃了VC++的核心，放弃了VC++最强大的部分。MFC也不是只能用于VC++中，它同样也可以用在Borland C++等编译器中，当然没有几个人这样做。

本节旨在让大家对VC++、VS2010和MFC有基本的概念上的认识，后面鸡啄米会带大家进入VS2010/MFC的世界，让大家轻松的开发各种包含窗口、图形等的可视化程序。

# 一、VS2010与MSDN安装过程图解

上一讲中鸡啄米对VC++和MFC做了一些简单介绍。在本套教程中鸡啄米将使用VS2010为大家讲解如何使用VC++和MFC进行编程，所以首先要安装VS2010。

## **一.下载VS2010**

首先我们需要下载VS2010，大家可以在网上下载VS2010破解正式版，建议选择英文版，养成使用英文工具的习惯。鸡啄米使用VS2010旗舰试用版VS2010UltimTrial.iso为例介绍安装过程，旗舰试用版官方下载地址为：

<http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=12187>。

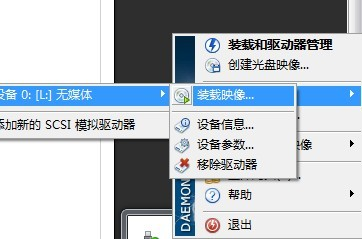
正式版的安装过程与试用版类似。

## **二.安装VS2010**

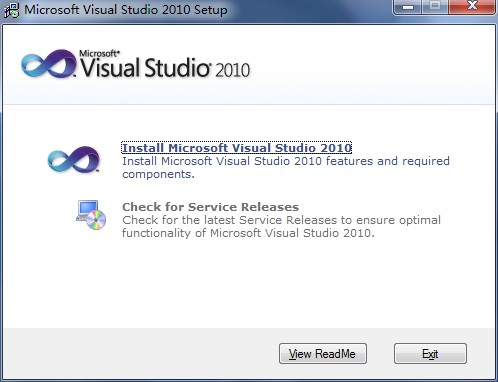
下载后进行安装。安装方法与一般的iso文件一样，可以使用虚拟光驱软件Daemon Tools安装，也可以将其解压后点击setup.exe进行安装。

鸡啄米为了让大家更直观的看到安装过程，我将在自己机子上再重新安装一次，并截图为大家讲解。

这里使用Daemon Tools安装VS2010。首先打开Daemon Tools，屏幕右下角会出现托盘图标，在图标上点右键，会弹出菜单，再把鼠标移到菜单项“虚拟设备”上，然后再移到子菜单项“设备 0：[L:] 无媒体”上，最后点击下一级子菜单项“装载映像”，弹出对话框选择VS2010UltimTrial.iso文件。



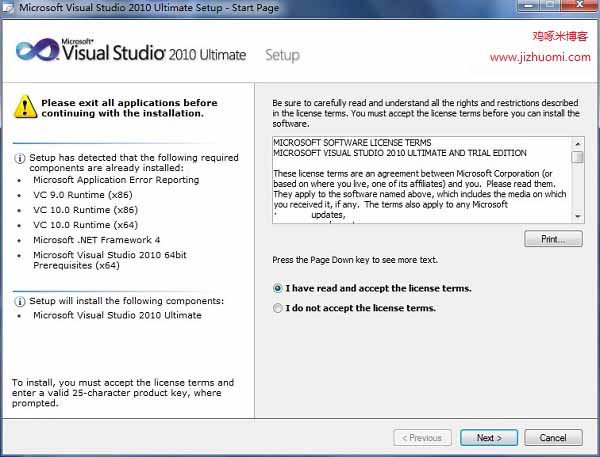
这样虚拟光驱就会打开此iso文件，弹出自动安装的提示，选择“运行autorun.exe”就可以了，如果没有弹出提示就通过资源管理器进入虚拟光驱，用setup.exe安装。接着会弹出下面的对话框：



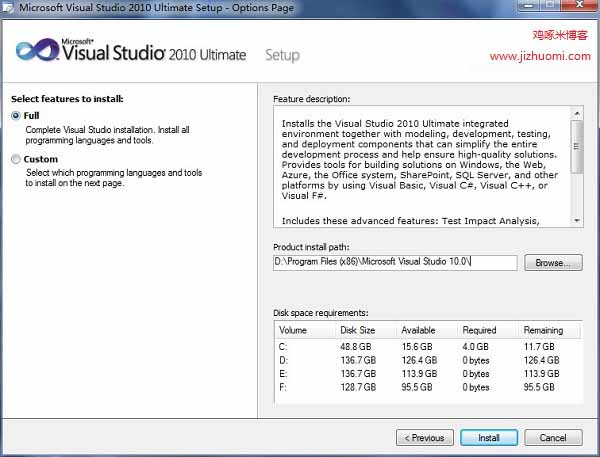
当然选择“Install Microsoft Visual Studio 2010”进入下一步，加载安装组件后如下显示：



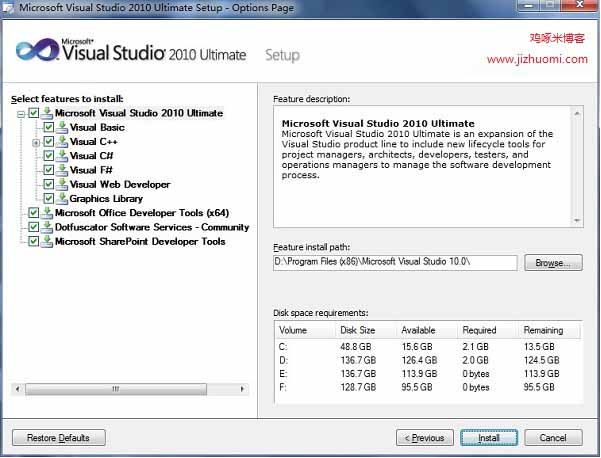
 点“Next”后：



选择“I have read and accept the license terms”后点“Next”弹出对话框：

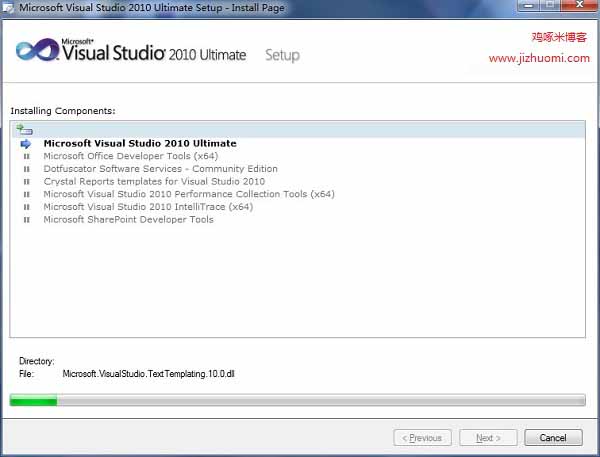


此处是让我们选择要安装的功能，有两种：Full（完全）和Custom（自定义）。Full选项表示安装所有编程语言和工具，Custom选择表示可以自定义要安装的编程语言和工具。右侧可以更改安装路径，鸡啄米建议不要安装到C盘，因为它占用的空间比较大。鸡啄米安装到了D盘，使用Full完全安装。如果选择Custom安装，点“Next”则出现如下画面：

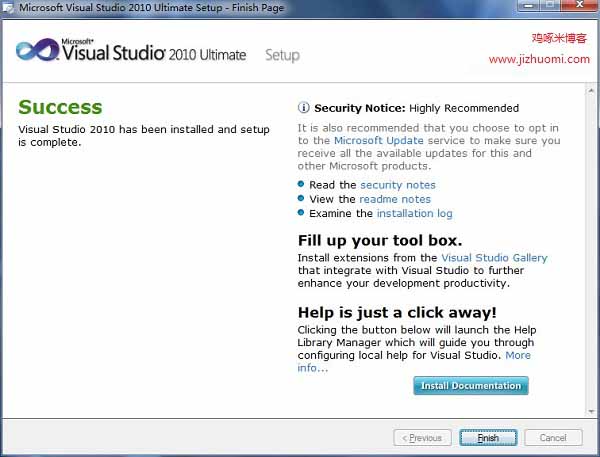


大家可以根据自己的需要取消某些语言或工具的安装，比如不想安装Visual C#，取消选择它就可以了。如果觉得以后都有可能会用到，那就像鸡啄米一样选择完全安装吧。

Full或Custom方式和安装路径设置好后，点“Install”进行安装：



可能正式版的安装文件在安装过程中会有重启过程。鸡啄米使用的试用版中间并没有重启。安装完成：

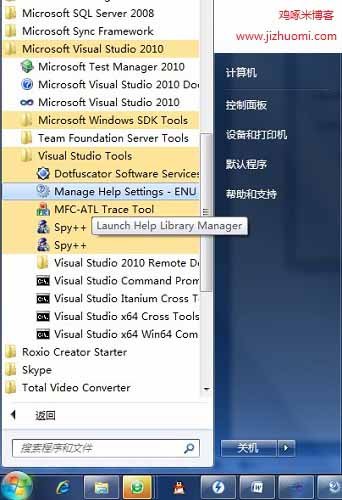


如果要继续安装MSDN，先不要卸载虚拟光驱映像。

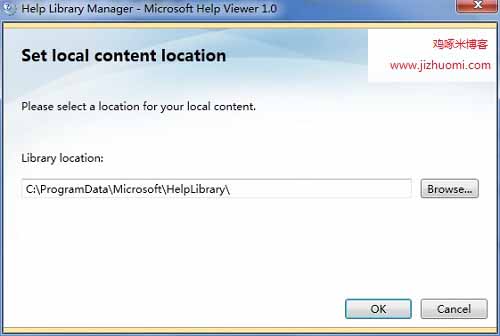
## **三.安装MSDN**

我们使用VS2010进行软件开发同样离不开帮助文档，即MSDN。在本地安装MSDN的方法如下：

在开始菜单的“所有程序”->“Microsoft Visual Studio 2010”->“Visual Studio Tools”下选择“Manage Help Settings - ENU”：



弹出对话框：



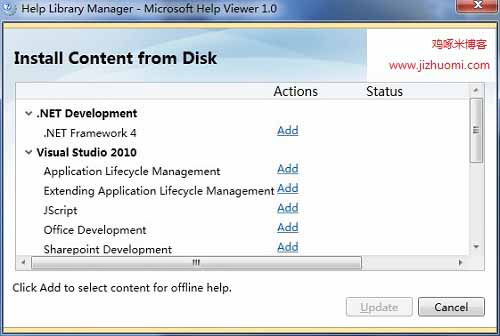
可以将帮助库存在默认路径，也可以修改存放路径。鸡啄米使用默认路径，点“OK”出现：



选择“Install Content From Disk”后弹出对话框选择帮助所在文件，这时需要在加载了VS2010的虚拟光驱中找，选择图中所示路径：



点OK后出现如下对话框，可以点“Add”选择要添加的帮助库，鸡啄米全部添加了。



点“Update”进行安装，等待其完成就可以了。

使用MSDN时点击开始菜单的“所有程序”->“Microsoft Visual Studio 2010”->“Microsoft Visual Studio 2010 Documentation”即可。

到此VS2010和MSDN的安装过程就结束了。以后就可以正式使用VS2010进行软件开发了。至于[VS2010的使用方法](http://www.jizhuomi.com/software/27.html)在鸡啄米的C++编程入门系列中已经介绍过，大家可以看看。

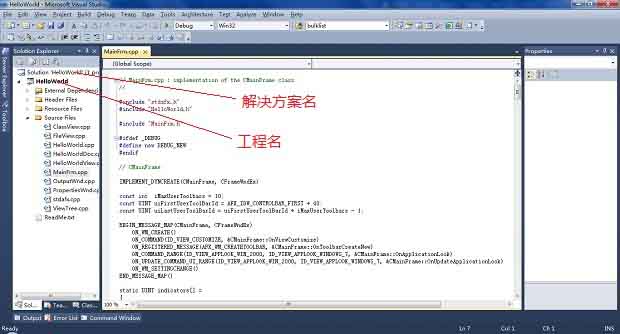
# 二、利用MFC向导生成单文档应用程序框架

上一讲中讲了[VS2010和MSDN如何安装](http://www.jizhuomi.com/software/139.html)，相信大家都已经安装好了。这一讲给大家一个简单的例子，演示如何生成单文档应用程序框架。

## **解决方案与工程**

鸡啄米在[VS2010的使用介绍](http://www.jizhuomi.com/software/27.html)中已经讲了解决方案与工程的概念，这里再重提一下。每个应用程序都作为一个工程来处理，它包含了头文件、源文件和资源文件等，这些文件通过工程集中管理。在VS2010中，工程都是在解决方案管理之下的。一个解决方案可以管理多个工程，可以把解决方案理解为多个有关系或者没有关系的工程的集合。VS2010提供了一个Solution Explorer解决方案浏览器视图，可以显示当前解决方案的内容，当新建一个工程时可以选择新建一个解决方案还是加入当前解决方案。

下图左侧面板中正在显示的视图就是Solution Explorer，视图中有一个解决方案-HelloWorld，此解决方案下有一个同名的工程-HelloWorld。



在应用程序向导生成应用程序后，VS2010会在用户设置的路径下，以解决方案名为名称建立一个目录，里面存放自动生成的文件。

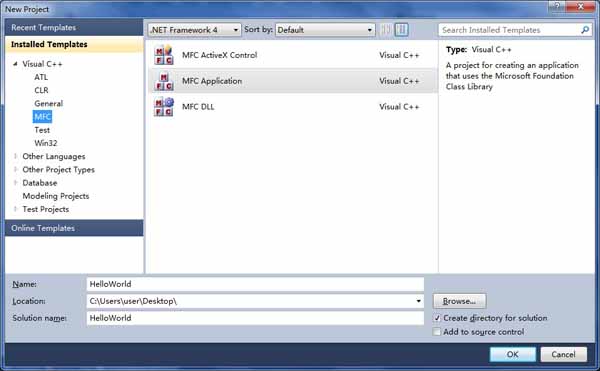
## **使用VS2010应用程序向导生成单文档应用程序框架**

鸡啄米这里简略演示下怎样生成单文档应用程序框架，让大家先有个直观的了解，有不理解的地方可以留着以后回来再看。下面按照操作步骤一步步讲解：

1.点菜单栏File->New->Project，弹出New Project对话框，我们可以选择工程类型。

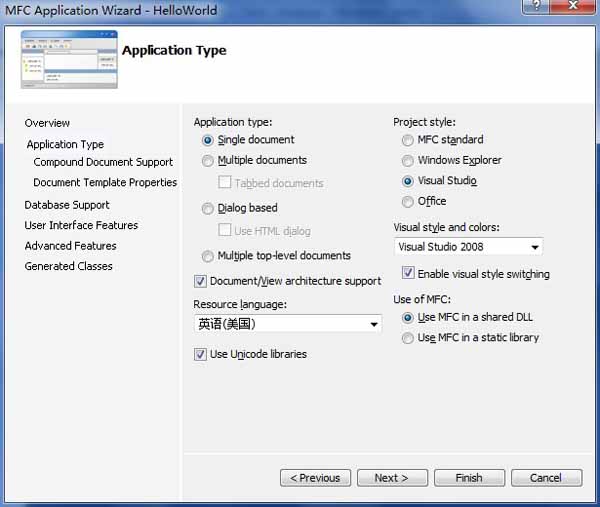
如果安装完VS2010以后第一启动时已经设置为VC++，则Installed Templates->Visual C++项会默认展开，而如果没有设置VC++，则可以展开到Installed Templates->Other Languages->Visual C++项。因为我们要生成的是MFC程序，所以在“Visual C++”下选择“MFC”，对话框中间区域会出现三个选项：MFC ActiveX Control、MFC Application和MFC DLL。MFC ActiveX Control用来生成MFC ActiveX控件程序。MFC Application用来生成MFC应用程序。MFC DLL用来生成MFC动态链接库程序。当然我们要选择MFC Application。

在对话框下部有Name、Location和Solution name三个设置项。意义如下：Name--工程名，Location--解决方案路径，Solution name--解决方案名称。这里Name我们设为“HelloWorld”，Location设置为“桌面”的路径，Solution name默认和Name一样，当然可以修改为其他名字，这里我们不作修改，也使用“HelloWorld”。点“OK”按钮。



2.这时会弹出“MFC Application Wizard”对话框，上部写有“Welcome to the MFC Application Wizard”，下面显示了当前工程的默认设置。第一条“Tabbed multiple document interface (MDI)”是说此工程是多文档应用程序。如果这时直接点下面的“Finish”按钮，可生成具有上面列出设置的多文档程序。但我们此例是要建立单文档应用程序，所以点“Next”按钮再继续设置吧。

3.接下来弹出的对话框上部写有“Application Type”，当然是让选择应用程序类型，我们看到有四种类型：Single document（单文档）、Multiple documents（多文档）、Dialog based（基于对话框）和Multiple top-level documents。我们选择Single document类型，以生成一个单文档应用程序框架。单文档应用程序运行时是一个单窗口界面。



此对话框的“Resource language”还提供语言的选择，这里默认选择英语。“Project style”可选择工程风格，我们选择默认的“Visual Studio”风格。“Use of MFC”有两个选项：Use MFC in a shared DLL（动态链接库方式使用MFC）和Use MFC in a static library（静态库方式使用MFC）。选择Use MFC in a shared DLL时MFC的类会以动态链接库的方式访问，所以我们的应用程序本身就会小些，但是发布应用程序时必须同时添加必要的动态链接库，以便在没有安装VS2010的机子上能够正常运行程序。选择Use MFC in a static library时MFC的类会编译到可执行文件中，所以应用程序的可执行文件要比上种方式大，但可以单独发布，不需另加包含MFC类的库。这里我们使用默认的Use MFC in a shared DLL。点“Next”按钮。

4.此时弹出上部写有“Compound Document Support”的对话框，可以通过它向应用程序加入OLE支持，指定OLE选项的复合文档类型。本例不需要OLE特性，使用默认值“None”。点“Next”按钮。

5.弹出的新对话框上部写有“Document Template Properties”。“File extension”可以设置程序能处理的文件的扩展名。对话框其他选项还可以更改程序窗口的标题。我们都使用默认设置，点“Next”按钮。

6.此时弹出的对话框主题是“Database Support”。用于设置数据库选项。此向导可以生成数据库应用程序需要的代码。它有四个选项：

None：忽略所有的数据库支持；

Header files only：只包含定义了数据库类的头文件，但不生成对应特定表的数据库类或视图类；

Database view without file support：创建对应指定表的一个数据库类和一个视图类，不附加标准文件支持；

Database view with file support：创建对应指定表的一个数据库类和一个视图类，并附加标准文件支持。

本例选择默认值“None”，不使用数据库特性。点“Next”按钮。

7.这时弹出的对话框是关于“User Interface Features”，即用户界面特性。我们可以设置有无最大化按钮、最小化按钮、系统菜单和初始状态栏等。还可以选择使用菜单栏和工具栏生成简单的应用程序还是使用ribbon。这里我们都选择默认设置。点“Next”进入下一步。

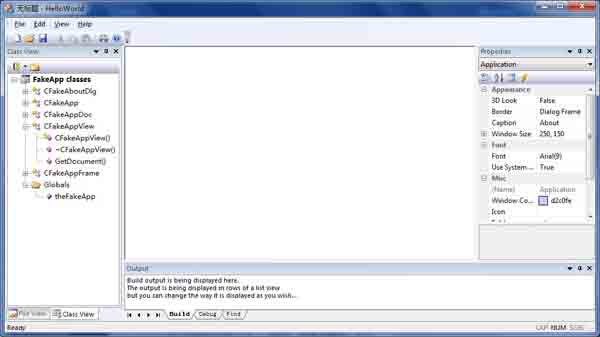
8.此时弹出“高级特性”对话框。可以设置的高级特性包括有无打印和打印预览等。在“Number of files on recent file list”项可以设置在程序界面的文件菜单下面最近打开文件的个数。我们仍使用默认值。点“Next”按钮。

9.弹出“生成类”对话框。在对话框上部的“生成类”列表框内，列出了将要生成的4 个类：一个视图类（CHelloWorldView）、一个应用类（CHelloWorldApp）、一个文档类（CHelloWorldDoc）和一个主框架窗口类（CMainFrame）。在对话框下面的几个编辑框中，可以修改默认的类名、类的头文件名和源文件名。对于视图类，还可以修改其基类名称，默认的基类是CView，还有其他几个基类可以选择。这里我们还是使用默认设置。点“Finish”按钮。

应用程序向导最后为我们生成了应用程序框架，并在Solution Explorer中自动打开了解决方案（见上面第一张图）。

## **编译运行生成的程序**

点菜单中的Build->Build HelloWorld编译程序，然后点Debug->Start Without Debugging（快捷键Ctrl+F5）运行程序，也可以直接点Debug->Start Without Debugging，这时会弹出对话框提示是否编译，选择“Yes”，VS2010将自动编译链接运行HelloWorld程序。结果页面如下所示：



终于看见界面了。鸡啄米在以后的教程中会继续讲解各种界面和控件的使用方法。欢迎到鸡啄米博客交流，您的关注是我前进的动力。

# 三、VS2010应用程序工程中文件的组成结构

鸡啄米在上一讲中为大家演示了如何[利用应用程序向导创建单文档应用程序框架](http://www.jizhuomi.com/software/141.html)。这一节将以上一讲中生成应用程序HelloWorld的文件结构为例，讲解[VS2010](http://www.jizhuomi.com/catalog.asp?tags=VS2010)应用程序工程中文件的组成结构。

用应用程序向导生成框架程序后，我们可以在之前设置的Location下看到以解决方案名命名的文件夹，此文件夹中包含了几个文件和一个以工程名命名的子文件夹，这个子文件夹中又包含了若干个文件和一个res文件夹，创建工程时的选项不同，工程文件夹下的文件可能也会有所不同。

如果已经以Debug方式编译链接过程序，则会在解决方案文件夹下和工程子文件夹下各有一个名为“Debug”的文件夹，而如果是Release方式编译则会有名为“Release”的文件夹。这两种编译方式将产生两种不同版本的可执行程序：Debug版本和Release版本。Debug版本的可执行文件中包含了用于调试的信息和代码，而Release版本则没有调试信息，不能进行调试，但可执行文件比较小。

**鸡啄米将所有文件分为6个部分：**解决方案相关文件、工程相关文件、应用程序头文件和源文件、资源文件、预编译头文件和编译链接生成文件。

## **1.解决方案相关文件**

解决方案相关文件包括解决方案文件夹下的.sdf文件、.sln文件、.suo文件和ipch文件夹。

.sdf文件和ipch目录一般占用空间比较大，几十兆甚至上百兆，与智能提示、错误提示、代码恢复和团队本地仓库等相关。如果你觉得不需要则可以设置不生成它们，方法是点击菜单栏Tools->Options，弹出Options对话框，选择左侧面板中Text Editor ->C/C++->Advanced，右侧列表中第一项Disable Database由False改为True就可以了，最后关闭VS2010再删除.sdf文件和ipch目录以后就不会再产生了。但关闭此选项以后也会有很多不便，例如写程序时的智能提示没有了。

.sln文件和.suo文件为MFC自动生成的解决方案文件，它包含当前解决方案中的工程信息，存储解决方案的设置。

## **2.工程相关文件**

工程相关文件包括工程文件夹下的.vcxproj文件和.vcxproj.filters文件。

.vcxproj文件是MFC生成的工程文件，它包含当前工程的设置和工程所包含的文件等信息。.vcxproj.filters文件存放工程的虚拟目录信息，也就是在解决方案浏览器中的目录结构信息。

## **3.应用程序头文件和源文件**

应用程序向导会根据应用程序的类型（单文档、多文档或基于对话框的程序）自动生成一些头文件和源文件，这些文件是工程的主体部分，用于实现主框架、文档、视图等。鸡啄米下面分别简单介绍下各个文件：

HelloWorld.h：应用程序的主头文件。主要包含由CWinAppEx类派生的CHelloWorldApp类的声明，以及CHelloWorldApp类的全局对象theApp的声明。

HelloWorld.cpp：应用程序的主源文件。主要包含CHelloWorldApp类的实现，CHelloWorldApp类的全局对象theApp的定义等。

MainFrm.h和MainFrm.cpp：通过这两个文件从CFrameWndEx类派生出CMainFrame类，用于创建主框架、菜单栏、工具栏和状态栏等。

HelloWorldDoc.h和HelloWorldDoc.cpp：这两个文件从CDocument类派生出文档类CHelloWorldDoc，包含一些用来初始化文档、串行化（保存和装入）文档和调试的成员函数。

HelloWorldView.h和HelloWorldView.cpp：它们从CView类派生出名为CHelloWorldView的视图类，用来显示和打印文档数据，包含了一些绘图和用于调试的成员函数。

ClassView.h和ClassView.cpp：由CDockablePane类派生出CClassView类，用于实现应用程序界面左侧面板上的Class View。

FileView.h和FileView.cpp：由CDockablePane类派生出CFileView类，用于实现应用程序界面左侧面板上的File View。

OutputWnd.h和OutputWnd.cpp：由CDockablePane类派生出COutputWnd类，用于实现应用程序界面下侧面板Output。

PropertiesWnd.h和PropertiesWnd.cpp：由CDockablePane类派生出CPropertiesWnd类，用于实现应用程序界面右侧面板Properties。

ViewTree.h和ViewTree.cpp：由CTreeCtrl类派生出CViewTree类，用于实现出现在ClassView和FileView等中的树视图。

## **4.资源文件**

一般我们使用MFC生成窗口程序都会有对话框、图标、菜单等资源，应用程序向导会生成资源相关文件：res目录、HelloWorld.rc文件和Resource.h文件。

res目录：工程文件夹下的res目录中含有应用程序默认图标、工具栏使用图标等图标文件。

HelloWorld.rc：包含默认菜单定义、字符串表和加速键表，指定了默认的About对话框和应用程序默认图标文件等。

Resource.h：含有各种资源的ID定义。

## **5.预编译头文件**

几乎所有的MFC程序的文件都要包含afxwin.h等文件，如果每次都编译一次则会大大减慢编译速度。所以把常用的MFC头文件都放到了stdafx.h文件中，然后由stdafx.cpp包含stdafx.h文件，编译器对stdafx.cpp只编译一次，并生成编译之后的预编译头HelloWorld.pch，大大提高了编译效率。

## **6.编译链接生成文件**

如果是Debug方式编译，则会在解决方案文件夹和工程文件夹下都生成Debug子文件夹，而如果是Release方式编译则生成Release子文件夹。

工程文件夹下的Debug或Release子文件夹中包含了编译链接时产生的中间文件，解决方案文件夹下的Debug或Release子文件夹中主要包含有应用程序的可执行文件。

关于应用程序工程文件的组成结构鸡啄米就先讲到这了。其中包含了很多专有名词，以后大家会慢慢熟悉的。欢迎来鸡啄米博客交流。谢谢。

# 四、MFC应用程序框架分析

上一讲鸡啄米讲的是[VS2010应用程序工程中文件的组成结构](http://www.jizhuomi.com/software/143.html)，可能大家对工程的运行原理还是很模糊，理不出头绪，毕竟跟C++编程入门系列中的例程差别太大。这一节鸡啄米就为大家分析下MFC应用程序框架的运行流程。

## **一.SDK应用程序与MFC应用程序运行过程的对比**

程序运行都要有入口函数，在之前的C++教程中都是main函数，而Windows应用程序的入口函数是WinMain函数，MFC程序也是从WinMain函数开始的。下面鸡啄米就给出用Windows SDK写的“HelloWorld”程序，与应用程序框架进行对比，这样能更好的了解框架是怎样运行的。Windows SDK开发程序就是不使用MFC类库，直接用Windows API函数进行[软件开发](http://www.jizhuomi.com/)。鸡啄米不是要讲解SDK开发，只是为了对比而简单介绍，至于SDK开发可以在大家学完MFC以后选择是否要研究，一般来说有简单了解就可以了。

### **SDK应用程序**

首先，给出Windows SDK应用程序“HelloWorld”的源码：

C++代码

1. include <windows>
3. int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR szCmdLine, int iCmdShow)
4. {
5. const static TCHAR appName[] = TEXT("Hello world");
6. WNDCLASSEX myWin;
7. myWin.cbSize = sizeof(myWin);
8. myWin.style = CS\_HREDRAW | CS\_VREDRAW;
9. myWin.lpfnWndProc = myWndProc;
10. myWin.cbClsExtra = 0;
11. myWin.cbWndExtra = 0;
12. myWin.hInstance = hInstance;
13. myWin.hIcon = 0;
14. myWin.hIconSm  = 0;
15. myWin.hCursor = 0;
16. myWin.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);
17. myWin.lpszMenuName = 0;
18. myWin.lpszClassName = appName;
19. //Register
20. if (!RegisterClassEx(&myWin)) return 0;
21. const HWND hWindow = CreateWindow(
22. appName,
23. appName,
24. WS\_OVERLAPPEDWINDOW,
25. CW\_USEDEFAULT,
26. CW\_USEDEFAULT,
27. CW\_USEDEFAULT,
28. CW\_USEDEFAULT,
29. 0,
30. 0,
31. hInstance,
32. 0);
33. ShowWindow(hWindow,iCmdShow);
34. UpdateWindow(hWindow);
35. {
36. MSG msg;
37. while(GetMessage(&msg,0,0,0))
38. {
39. TranslateMessage(&msg);
40. DispatchMessage(&msg);
41. }
42. return (int)msg.wParam;
43. }
44. }
46. LRESULT CALLBACK myWndProc(HWND hWindow, UINT msg, WPARAM wParam, LPARAM lParam)
47. {
48. if (msg==WM\_PAINT)
49. {
50. PAINTSTRUCT ps;
51. const HDC hDC = BeginPaint(hWindow,&ps);
52. RECT rect;
53. GetClientRect(hWindow,&rect);
54. DrawText(hDC,TEXT("HELLO WORLD"),-1,&rect, DT\_SINGLELINE | DT\_CENTER | DT\_VCENTER);
55. EndPaint(hWindow,&ps);
56. return 0;
57. }
58. else if (msg==WM\_DESTROY)
59. {
60. PostQuitMessage(0);
61. return 0;
62. }
63. return DefWindowProc(hWindow,msg,wParam,lParam);
64. }

上面的程序运行的流程是：进入WinMain函数->初始化WNDCLASSEX，调用RegisterClassEx函数注册窗口类->调用ShowWindow和UpdateWindow函数显示并更新窗口->进入消息循环。关于消息循环再简单说下，Windows应用程序是消息驱动的，系统或用户让应用程序进行某项操作或完成某个任务时会发送消息，进入程序的消息队列，然后消息循环会将消息队列中的消息取出，交予相应的窗口过程处理，此程序的窗口过程函数就是myWndProc函数，窗口过程函数处理完消息就完成了某项操作或任务。本例是要显示“HELLO WORLD”字符串，UpdateWindow函数会发送WM\_PAINT消息，但是此消息不经过消息队列而是直接送到窗口过程处理，在窗口过程函数中最终绘制了“HELLO WORLD”字符串。

### **MFC应用程序**

下面是MFC应用程序的运行流程，通过MFC库中代码进行分析：

首先在HelloWorld.cpp中定义[全局对象](http://www.jizhuomi.com/software/60.html)theApp：CHelloWorldApp theApp;。调用CWinApp和CHelloWorldApp的构造函数后，进入WinMain函数（位于appmodul.cpp中）。

C++代码

1. extern "C" int WINAPI
2. \_tWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,\_In\_ LPTSTR lpCmdLine, int nCmdShow)
3. #pragma warning(suppress: 4985)
4. {
5. // call shared/exported WinMain
6. return AfxWinMain(hInstance, hPrevInstance, lpCmdLine, nCmdShow);
7. }

在TCHAR.h中，有此定义：#define \_tWinMain   WinMain，所以这里的\_tWinMain就是WinMain函数。它调用了AfxWinMain函数（位于WinMain.cpp中）。

C++代码

1. int AFXAPI AfxWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,LPTSTR lpCmdLine, int nCmdShow)
2. {
3. .............略
4. // App global initializations (rare)
5. if (pApp != NULL && !pApp->InitApplication())
6. goto InitFailure;
8. if (!pThread->InitInstance())
9. {
10. .........略
11. }
13. // Run函数位于THRDCORE.cpp中，由此函数进入消息循环
14. nReturnCode = pThread->Run();
16. ..............略
18. return nReturnCode;
19. }

上面InitInstance函数的代码如下：

C++代码

1. BOOL CTestApp::InitInstance()
2. {
3. .............略
4. CSingleDocTemplate\* pDocTemplate;
5. pDocTemplate = new CSingleDocTemplate(
6. IDR\_MAINFRAME,
7. RUNTIME\_CLASS(CTestDoc),
8. RUNTIME\_CLASS(CMainFrame),      // main SDI frame window
9. RUNTIME\_CLASS(CTestView));
10. if (!pDocTemplate)
11. return FALSE;
12. AddDocTemplate(pDocTemplate);
13. // Parse command line for standard shell commands, DDE, file open
15. CCommandLineInfo cmdInfo;
16. ParseCommandLine(cmdInfo);
18. //ProcessShellCommand位于AppUI2.cpp中，注册并创建窗口
19. if (!ProcessShellCommand(cmdInfo))
20. return FALSE;
22. m\_pMainWnd->ShowWindow(SW\_SHOW);
23. m\_pMainWnd->UpdateWindow();
25. return TRUE;
26. }

InitInstance中的ProcessShellCommand函数又调用了CMainFrame的LoadFrame函数注册并创建了窗口，执行完ProcessShellCommand函数以后，调用了m\_pMainWnd的ShowWindow和UpdateWindow函数显示并更新框架窗口。这些是不是与上面的SDK程序十分类似？

接下来该是消息循环了，上面的AfxWinMain函数中调用了pThread的Run函数（位于THRDCORE.cpp中），在Run中包含了消息循环。Run函数的代码如下：

C++代码

1. int CWinThread::Run()
2. {
3. .............略
4. // phase2: pump messages while available
5. do
6. {
7. // pump message, but quit on WM\_QUIT
8. if (!PumpMessage())
9. return ExitInstance();
11. // reset "no idle" state after pumping "normal" message
12. if (IsIdleMessage(&m\_msgCur))
13. {
14. bIdle = TRUE;
16. lIdleCount = 0;
18. }
19. } while (::PeekMessage(&m\_msgCur, NULL, NULL, NULL, PM\_NOREMOVE));
20. ..............略
21. }
23. BOOL CWinThread::PumpMessage()
24. {
25. return AfxInternalPumpMessage();
26. }
28. BOOL AFXAPI AfxInternalPumpMessage()
29. {
30. \_AFX\_THREAD\_STATE \*pState = AfxGetThreadState();
32. if (!::GetMessage(&(pState->m\_msgCur), NULL, NULL, NULL))
33. {
34. .............略
35. }
36. ...............略
37. if (pState->m\_msgCur.message != WM\_KICKIDLE && !AfxPreTranslateMessage(&(pState->m\_msgCur)))
38. {
39. ::TranslateMessage(&(pState->m\_msgCur));
40. ::DispatchMessage(&(pState->m\_msgCur));
41. }
43. return TRUE;
44. }

我们看到PumpMessage中通过调用GetMessage、TranslateMessage、DispatchMessage等建立了消息循环并投递消息。

窗口过程函数AfxWinProc形式如下：

C++代码

1. LRESULT CALLBACK AfxWndProc(HWND hWnd,UINT nMsg,WPARAM wParam, LPARAM lParam)
2. {
3. ……
4. CWnd\*pWnd=CWnd::FromHandlePermanent(hWnd);
5. ReturnAfxCallWndProc(pWnd,hWnd,nMsg,wParam,lParam);
6. }

两者运行过程对比

到此，通过对比可以发现，MFC应用程序的运行流程与SDK程序是类似的，都是先进行一些初始化过程，再注册并创建窗口，然后显示、更新窗口，最后进入消息循环，消息都由窗口过程函数处理。现在大家是不是觉得有些头绪了？在运行流程上有基本的掌握即可。

## **二.MFC应用程序框架主要类之间的关系**

在第二讲中，给大家演示了如何[利用应用程序向导生成单文档应用程序框架](http://www.jizhuomi.com/software/141.html)，可以看到程序的基本框架和必要的代码都自动生成了，上一讲又讲解了文件组成结构，实际上在前面自动生成的框架中比较重要的类包括以下几个：CHelloWorldApp、CMainFrame、CHelloWorldDoc和CHelloWorldView，至于其他的类比如CClassView、CFileView等都是在框架窗口（CMainFrame）上创建的面板等，不是必要的。

鸡啄米就四个主要类的关系简单讲下，CHelloWorldApp类处理消息，将收到的消息分发给相应的对象。CMainFrame是视图CHelloWorldView的父窗口，视图CHelloWorldView就显示在CMainFrame的客户区中。视图类CHelloWorldView用来显示文档类CHelloWorldDoc中的数据，并根据对视图类的操作修改文档类的数据。一个视图类只能跟一个文档类相联系，而一个文档类可以跟多个视图类相联系。关于视图类和文档类的关系后面会详细讲解。

本节VC++/MFC[编程入门](http://www.jizhuomi.com/catalog.asp?tags=%E7%BC%96%E7%A8%8B%E5%85%A5%E9%97%A8)教程内容比较多，主要是让大家对MFC应用程序的运行原理有大概的了解。对于以后的MFC开发有很多好处。如果有问题请在鸡啄米博客留言交流。谢谢。

# 五、MFC消息映射机制概述

上一讲鸡啄米为大家简单分析了[MFC应用程序框架](http://www.jizhuomi.com/software/145.html)，这一讲是关于MFC消息映射机制的内容。

前面已经说过，Windows应用程序是消息驱动的。在MFC[软件开发](http://www.jizhuomi.com/)中，界面操作或者线程之间通信都会经常用到消息，通过对消息的处理实现相应的操作。比较典型的过程是，用户操作窗口，然后有消息产生，送给窗口的消息处理函数处理，对用户的操作做出响应。

### **什么是消息**

窗口消息一般由三个部分组成：(1)一个无符号整数，是消息值；(2)消息附带的WPARAM类型的参数；(3)消息附带的LPARAM类型的参数。其实我们一般所说的消息是狭义上的消息值，也就是一个无符号整数，经常被定义为宏。

### **什么是消息映射机制**

MFC使用一种消息映射机制来处理消息，在应用程序框架中的表现就是一个消息与消息处理函数一一对应的消息映射表，以及消息处理函数的声明和实现等代码。当窗口接收到消息时，会到消息映射表中查找该消息对应的消息处理函数，然后由消息处理函数进行相应的处理。SDK编程时需要在窗口过程中一一判断消息值进行相应的处理，相比之下MFC的消息映射机制要方便好用的多。

### **Windows消息分类**

先讲下Windows消息的分类。Windows消息分为系统消息和用户自定义消息。Windows系统消息有三种：

1.标准Windows消息。除WM\_COMMAND外以WM\_开头的消息是标准消息。例如，WM\_CREATE、WM\_CLOSE。

2.命令消息。消息名为WM\_COMMAND，消息中附带了标识符ID来区分是来自哪个菜单、工具栏按钮或加速键的消息。

3.通知消息。通知消息一般由列表框等子窗口发送给父窗口，消息名也是WM\_COMMAND，其中附带了控件通知码来区分控件。

CWnd的派生类都可以接收到标准Windows消息、通知消息和命令消息。命令消息还可以由文档类等接收。

用户自定义消息是实际上就是用户定义一个宏作为消息，此宏的值应该大于等于WM\_USER，然后此宏就可以跟系统消息一样使用，窗口类中可以定义它的处理函数。

### **消息映射表**

除了一些没有基类的类或CObject的直接派生类外，其他的类都可以自动生成消息映射表。下面的讲解都以前面例程HelloWorld的CMainFrame为例。消息映射表如下：

C++代码

1. BEGIN\_MESSAGE\_MAP(CMainFrame, CFrameWndEx)
2. ON\_WM\_CREATE()
3. ON\_COMMAND(ID\_VIEW\_CUSTOMIZE, &CMainFrame::OnViewCustomize)
4. ON\_REGISTERED\_MESSAGE(AFX\_WM\_CREATETOOLBAR, &CMainFrame::OnToolbarCreateNew)
5. ON\_COMMAND\_RANGE(ID\_VIEW\_APPLOOK\_WIN\_2000, ID\_VIEW\_APPLOOK\_WINDOWS\_7, &CMainFrame::OnApplicationLook)
6. ON\_UPDATE\_COMMAND\_UI\_RANGE(ID\_VIEW\_APPLOOK\_WIN\_2000, ID\_VIEW\_APPLOOK\_WINDOWS\_7, &CMainFrame::OnUpdateApplicationLook)
7. ON\_WM\_SETTINGCHANGE()
8. END\_MESSAGE\_MAP()

在BEGIN\_MESSAG\_MAP和END\_MESSAGE\_MAP之间的内容成为消息映射入口项。消息映射除了在CMainFrame的实现文件中添加消息映射表外，在类的定义文件MainFrm.h中还会添加一个宏调用：

DECLARE\_MESSAGE\_MAP()

一般这个宏调用写在类定义的结尾处。

### **添加消息处理函数**

如何添加消息处理函数呢？不管是自动还是手动添加都有三个步骤：

1.在类定义中加入消息处理函数的函数声明，注意要以afx\_msg打头。例如MainFrm.h中WM\_CREATE的消息处理函数的函数声明：afx\_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);。

2.在类的消息映射表中添加该消息的消息映射入口项。例如WM\_CREATE的消息映射入口项：ON\_WM\_CREATE()。

3.在类实现中添加消息处理函数的函数实现。例如，MainFrm.cpp中WM\_CREATE的消息处理函数的实现：

1. int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
2. {
3. ......
4. }

通过以上三个步骤以后，WM\_CREATE等消息就可以在窗口类中被消息处理函数处理了。

### **各种Windows消息的消息处理函数**

标准Windows消息的消息处理函数都与WM\_CREATE消息类似。

命令消息的消息映射入口项形式如：ON\_COMMAND (ID\_VIEW\_CUSTOMIZE, &CMainFrame::OnViewCustomize)，消息为ID\_VIEW\_CUSTOMIZE，消息处理函数为OnViewCustomize。

如果想要使用某个处理函数批量处理某些命令消息，则可以像CMainFrame消息映射表中的ON\_COMMAND\_RANGE (ID\_VIEW\_A PPLOOK\_WIN\_2000,ID\_VIEW\_APPLOOK\_WINDOWS\_7, &CmainFrame ::OnApplicationLook)一样添加消息映射入口项，这样值在ID\_VIEW\_APPLOOK\_WIN\_2000到ID\_VIEW\_ APPLOOK\_ WINDOWS\_7之间的菜单项等的命令消息都由CMainFrame的OnApplicationLook函数处理。函数原型为afx\_msg void OnApplicationLook(UINT id);，参数id为用户操作的菜单项等的ID。

在操作列表框等控件时往往会给父窗口发送WM\_NOTIFY通知消息。WM\_NOTIFY消息的wParam参数为发送通知消息的控件的ID，lParam参数指向一个结构体，可能是NMHDR结构体，也可能是第一个元素为NMHDR结构体变量的其他结构体。NMHDR结构体的定义如下（仅作了解）：

1. Typedef sturct tagNMHDR{
2. HWND hwndFrom;
3. UINT idFrom;
4. UINT code;
5. } NMHDR;

hwndFrom为发送通知消息控件的句柄，idFrom为控件ID，code为要处理的通知消息的通知码，例如NM\_CLICK。

通知消息的消息映射入口项形式如：

ON\_NOTIFY(wNotifyCode,id,memberFxn)

wNotifyCode为要处理的通知消息通知码，例如：NM\_CLICK。id为控件标识ID。MemberFxn为此消息的处理函数。

通知消息的处理函数的原型为：

afx\_msg void memberFxn( NMHDR \* pNotifyStruct, LRESULT \* result);

如果需要使用用户自定义消息，首先要定义消息宏，如：#define WM\_UPDATE\_WND (WM\_USER+1)，再到消息映射表中添加消息映射入口项：ON\_MESSAGE(WM\_UPDATE\_WND, &CMainFrame::OnUpdateWnd)，然后在MainFrm.h中添加消息处理函数的函数声明：afx\_msg LRESULT OnUpdateWnd(WPARAM wParam, LPARAM lParam);，最后在MainFrm.cpp中实现此函数。

鸡啄米本节对MFC消息映射机制只是做了比较简单的讲解，让大家对它有一定的认识，[编程入门](http://www.jizhuomi.com/catalog.asp?tags=%E7%BC%96%E7%A8%8B%E5%85%A5%E9%97%A8)者不必强求完全掌握。在以后的教程中会经常涉及到消息的使用，大家会逐渐熟悉MFC的消息映射机制。

# 六、对话框：创建对话框模板和修改对话框属性

鸡啄米在上一讲中介绍了[MFC的消息映射机制](http://www.jizhuomi.com/software/147.html)，属于原理方面的知识。对于[VC++编程入门](http://www.jizhuomi.com/catalog.asp?tags=%E7%BC%96%E7%A8%8B%E5%85%A5%E9%97%A8)学习者来说可能有些抽象，鸡啄米会把消息映射的知识渗透到后面的教程中。本节开始为大家讲解偏应用的知识-创建对话框。

对话框，大家应该很熟悉了，在我们常用的软件中大多都有对话框界面，例如，360安全卫士的主界面其实就是个对话框，只是它做了很多美工方面的工作，将其大大美化了。

创建对话框主要分两大步，第一，创建对话框资源，主要包括创建新的对话框模板、设置对话框属性和为对话框添加各种控件；第二，生成对话框类，主要包括新建对话框类、添加控件变量和控件的消息处理函数等。鸡啄米在本节中先讲讲怎样创建对话框模板和设置对话框属性。

## **创建基于对话框的应用程序框架**

之前鸡啄米创建的HelloWorld程序是[单文档应用程序](http://www.jizhuomi.com/software/141.html)，生成了多种窗口，如果用它来讲创建对话框的话可能有些复杂，对大家单纯理解对话框有点影响，所以这里鸡啄米就再创建一个基于对话框的应用程序，用来实现加法运算的功能。创建步骤同单文档应用程序大同小异，简单步骤如下：

1.选择菜单项File->New->Project，弹出“New Project”对话框。

2.左侧面板中Installed Templated的Visual C++下选择MFC，中间窗口中选择MFC Application，然后在下面的Name编辑框中键入工程名称，本例取名“Addition”，在Location编辑框中设置工程的保存路径。点“OK”。

3.点“Next”到“Application Type”对话框，在Application type下选择Dialog based，其他使用默认设置，点“Finish”。

我们可以在Solution Explorer视图中看到，此工程的文件要比单文档应用程序少的多，在Class View中主要有三个类：CAboutDlg、CAdditionApp和CAdditionDlg。CAboutDlg是应用程序的“关于”对话框类，CAdditionApp是由CWinApp派生的类，CAdditionDlg是主对话框类，主对话框也就是此应用程序运行后显示的主要界面。

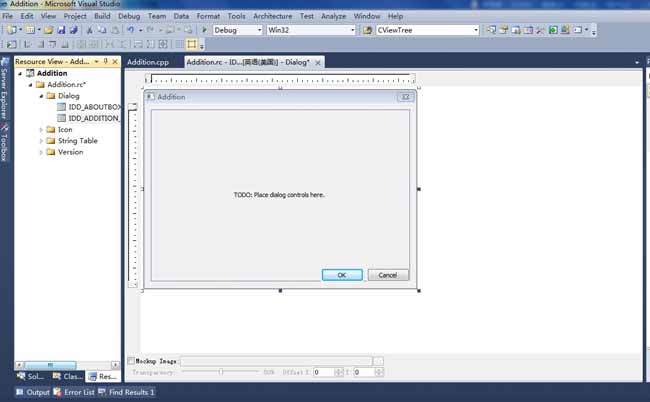
**注：**如果在VS2010中找不到Solution Explorer或Class View等视图，可以在菜单项View下找到对应视图选项选择即可。在[VS2010的使用介绍](http://www.jizhuomi.com/software/27.html)中已经有讲解。

在Resource View视图中可以看到工程Addition的资源树，展开Addition.rc，下面有四个子项：Dialog（对话框）、Icon（图标）、String Table（字符串表）和Version（版本）。然后展开Dialog项，下面有两个对话框模板，其ID分别为：IDD\_ABOUTBOX和IDD\_ADDITION\_DIALOG，前者是“关于”对话框的模板，后者是主对话框的模板。ID是资源的唯一标识，本质上是一个无符号整数，一般ID代表的整数值由系统定义，我们无需干涉。

## **对话框模板**

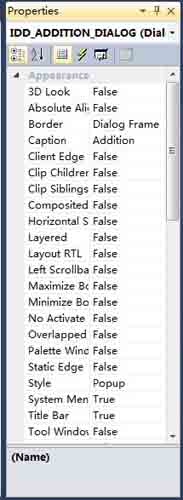
可见对于主对话框来说，创建对话框第一步中的创建新的对话框模板已经由系统自动完成了。而如果是再添加对话框需要创建新的对话框模板时，需要在Resource View的“Dialog”节点上点右键，在右键菜单中选择“Insert Dialog”，就会生成新的对话框模板，并且会自动分配ID。

在Resource View的资源树中双击某个ID，可在中间区域内显示相应的资源界面。双击IDD\_ADDITION\_DIALOG时，中间区域就会显示Addition对话框模板。如下图：



## **设置对话框属性**

在Addition对话框模板上点右键，然后在右键菜单中选择Properties，则在右侧面板中会显示对话框的属性列表。如下图：



鸡啄米在这里对经常使用的几个属性作简单说明，并对Addition对话框进行属性设置说明。

1.ID：对话框ID，唯一标识对话框资源，可以修改。此处为IDD\_ADDITION\_DIALOG，我们不修改它。

2.Caption：对话框标题。此处默认为Addition，我们将其修改为“加法计算器”。

3.Border：边框类型。有四种类型：None、Thin、Resizing和Dialog Frame。我们使用默认的Dialog Frame。

  4.Maximize：是否使用最大化按钮。我们使用默认的False。

5.Minimize：是否使用最小化按钮。同样我们使用默认的False。

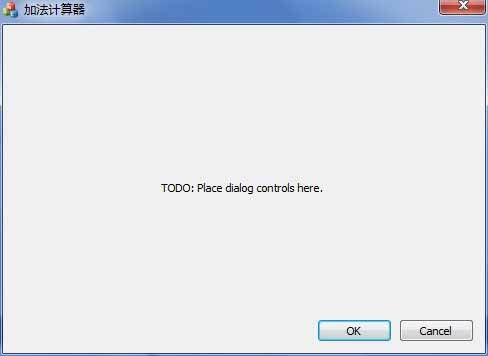
6.Style：对话框类型。有三种类型：Overlapped（重叠窗口）、Popup（弹出式窗口）和Child（子窗口）。弹出式窗口比较常见。我们使用默认的Popup类型。

7.System Menu：是否带有标题栏左上角的系统菜单，包括移动、关闭等菜单项。我们使用默认的True。

8.Title Bar：是否带有标题栏。我们使用默认的True。

9.Font(Size)：字体类型和字体大小。如果将其修改为非系统字体，则Use System自动改为False。而如果Use System原来为False，将其修改为True，则Font(Size)自动设置为系统字体。这里我们使用默认的系统字体。

根据以上说明，其实我们只修改了标题属性。这时我们运行此程序后的界面如下：



这一讲就先讲到这里了，对于创建对话框第一步中的为对话框添加各种控件下一讲为大家演示。欢迎来鸡啄米博客交流学习。

# 七、对话框：为对话框添加控件

创建对话框资源需要创建对话框模板、修改对话框属性、为对话框添加各种控件等步骤，前面一讲中鸡啄米已经讲了创建对话框模板和修改对话框属性，本节继续讲如何为对话框添加控件。

上一讲中鸡啄米创建了一个名为“Addition”的工程，目的是生成一个实现加法运算的应用程序。实现加法计算有几个必要的因素：被加数、加数、和。被加数和加数需要输入，和需要输出显示。那么这几个因素都需要相应的控件来输入或显示，下面鸡啄米就一步步讲解如何添加这些控件。

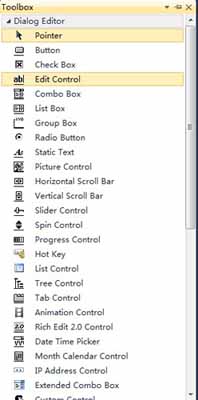
## 1、**添加静态文本框**

**1.为对话框添加一个静态文本框（Static Text），用于显示字符串--“被加数”。**

上一讲中生成的资源模板中自动添加了一个标题为“TODO:Place dialog controls here.”的静态文本框，我们可以修改它的标题继续使用，也可以删掉它。这里为了从头讲解静态文本框的添加过程，将它删掉，继续添加新的静态文本框。

删除控件时，可以使用鼠标左键点击选中它，选中后控件的周围会出现虚线框，然后按Delete键就可以将其删除了。在“Addition”工程的Resource View中打开上一讲中创建的对话框模板IDD\_ADDITION\_DIALOG，自动添加的静态文本框就可以使用这种方法删除。

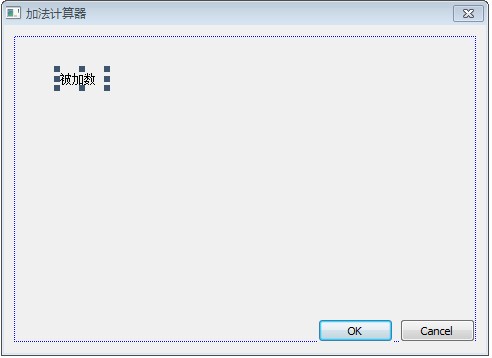
在添加新的静态文本框以前，先看看Toolbox视图是否显示了，如果没有显示，在菜单栏上点击View->Toolbox即可。Toolbox视图如下图：



Toolbox中列出了一些常用控件，其中有一个是Static Text，即是我们要添加的控件。在Toolbox中的Static Text上点下鼠标左键不放开，并拖到IDD\_ADDITION\_DIALOG对话框模板上，模板上会出现一个虚线框，我们找到合适的位置松开鼠标左键放下它。

用鼠标左键选中控件后周围出现虚线框，然后鼠标移到虚线框上几个黑点的位置会变成双向箭头的形状，此时就可以按下鼠标左键并拖动来改变控件大小了。我们可以这样改变新添加的静态文本框控件的大小，以更好的显示标题。当然，整个对话框模板也可以用这种方法改变大小。

接下来就该修改静态文本框的文字了。鼠标右键点击静态文本框，在右键菜单中选择“Properties”，Properties面板就会显示出来，在面板上修改Caption属性为“被加数”，ID修改为IDC\_SUMMAND\_STATIC。此时模板如下图：

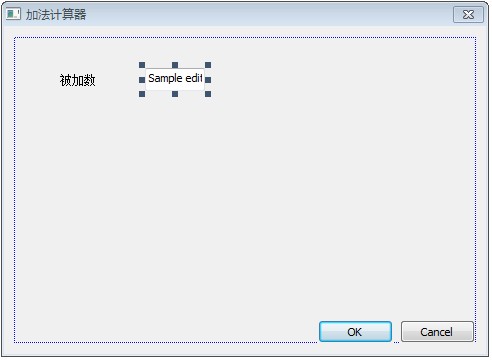


## **2、添加编辑框**

**2.为对话框添加一个编辑框（Edit Control），用来输入被加数。**

添加编辑框的过程与静态文本框类似，在Toolbox中选中Edit Control控件拖到对话框模板上，并使其与之前的静态文本框水平对齐（为了美观），然后调整其大小使之适合被加数的输入。

在编辑框上点右键，仍然在右键菜单中选择“Properties”显示出属性（Properties）面板，修改其ID为IDC\_SUMMAND\_EDIT。此时模板如下图：



3.按照1的方法添加一个标题为“加数”的静态文本框，用于显示字符串--“加数”。并将其ID改为IDC\_ADDEND\_STATIC。

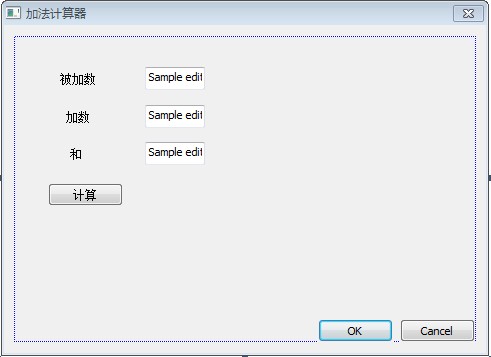
4.按照2的方法添加一个ID为IDC\_ADDEND\_EDIT的编辑框，用来输入加数。

5.按照1的方法添加一个标题为“和”的静态文本框，用于显示文字--“和”。并修改其ID为IDC\_SUM\_STATIC。

6.按照2的方法添加一个ID为IDC\_SUM\_EDIT的编辑框，用来显示最终的加和。

7.类似的添加按钮（Button）控件到对话框模板，用于在被点击后触发加法计算。修改其标题为“计算”，ID为IDC\_ADD\_BUTTON。

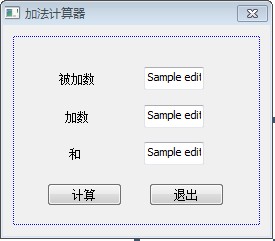
到此，对话框模板如图：



8.删除OK按钮。打开Cancel按钮的属性面板，将标题改为“退出”，并使其与“计算”按钮水平对齐。

9.根据控件的布局，适当调整整个对话框模板的大小，使其相对控件布局来说大小合适，界面美观。

这样在对话框模板中就把我们在本例中需要用到的控件就添加完了。最终效果如下：



至此，我们的对话框资源就基本创建完了。应用程序运行后的界面效果已经很清楚了。后面鸡啄米会讲如何在对话框类中实现加法计算功能，并能很好的和界面交互。欢迎继续到鸡啄米博客交流。

# 八、对话框：创建对话框类和添加控件变量

前两讲中鸡啄米为大家讲解了如何创建对话框资源。创建好对话框资源后要做的就是生成对话框类了。鸡啄米再声明下，生成对话框类主要包括新建对话框类、添加控件变量和控件的消息处理函数等。

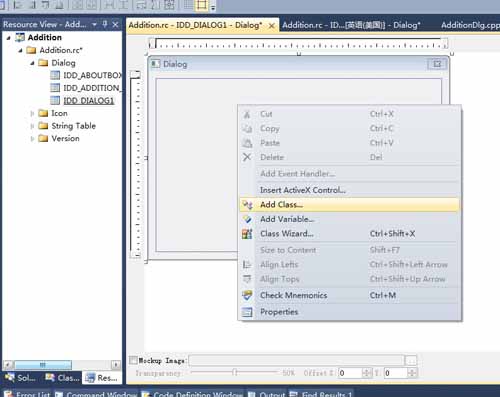
因为鸡啄米给大家的例程Addition是基于对话框的程序，所以程序自动创建了对话框模板IDD\_ADDITION\_DIALOG，并自动生成了对话框类CAdditionDlg，它是从CDialogEx类派生的。大家用过VC++ 6.0的可能记得，我们定义的对话框类都是从CDialog类派生的，但在VS2010中，一般对话框类都是继承自CDialogEx类。

## **创建对话框类**

如果是自己新添加的对话框模板，怎样为它创建对话框类呢？

1.首先鸡啄米就按第六讲：创建对话框模板和修改对话框属性中说的那样，在Resource View的“Dialog”节点上右键，然后在右键菜单中选择“Insert Dialog”创建一个新的对话框模板，ID就使用默认的IDD\_DIALOG1。

2.在中间区域会显示新建的对话框模板，然后选中此对话框模板，点右键，在右键菜单中选择Add Class。



3.选择“Add Class”后会弹出一个对话框，在对话框中“Class name”下的编辑框中写入自定义的类名就可以了，例如CMyDialog。

4.最后点“Finish”完成。

最终你就可以在Class View中看到新生成的对话框类CMyDialog了，并且在Solution Explorer中有相应的MyDialog.h头文件和MyDialog.cpp源文件生成。CMyDialog类同样派生于CDialogEx类。

注意，一般类名都以C打头，又比如，CTestDlg。

## **为对话框中的控件添加变量**

在上一讲中为对话框添加了几个控件，包括三个静态文本框，三个编辑框，一个按钮控件。程序自动生成的Cancel按钮保留，作为退出按钮，而OK按钮删除掉了。

静态文本框只是为了说明后面紧跟的编辑框中数据的意义，是被加数、加数还是和，所以它们是不会变的，我们就不为它们添加变量了。按钮控件是用来操作的，这里也不为它们添加变量。编辑框中的数据可能会经常变化，有必要为它们每个控件关联一个变量。

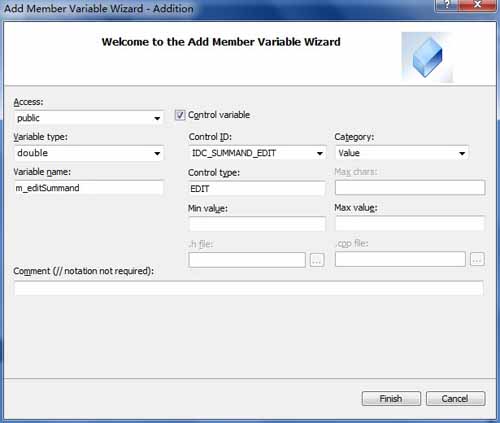
首先为被加数的编辑框IDC\_SUMMAND\_EDIT添加变量。

1.在编辑框上点右键，在右键菜单中选择“Add Variable”。弹出添加成员变量的向导对话框。

2.我们想为其添加值变量而不是控件变量，所以对话框中“Category”下的组合框中选择Value。

3.“Variable type”下的组合框此时默认选中的是“CString”，CString是字符串类，显然不能进行加法运算。我们可以选择double、float、int等。这里我们选择double，即编辑框关联一个double类型的变量。

4.在“Variable name”中写入自定义的变量名。鸡啄米为其取名m\_editSummand。



5.点“Finish”完成。

**注意，类的成员变量名一般以m\_打头，以标识它是一个成员变量。**

参照此方法，再分别为加数的编辑框IDD\_ADDEND\_EDIT添加double型变量m\_editAddend、和的编辑框IDD\_SUM\_EDIT添加double型变量m\_editSum。

## **对话框类的数据交换和检验**

在程序运行界面中，用户往往会改变控件的属性，例如，在编辑框中输入字符串，或者改变组合框的选中项，又或者改变复选框的选中状态等。控件的属性改变后MFC会相应修改控件关联变量的值。这种同步的改变是通过MFC为对话框类自动生成的成员函数DoDataExchange()来实现的，这也叫做对话框的数据交换和检验机制。

我们为三个编辑框添加了变量以后，在AdditionDlg.cpp中CAdditionDlg的DoDataExchange()函数的函数体中多了三条DDX\_Text调用语句。下面是函数体代码和鸡啄米添加的注释。

C++代码

1. void CAdditionDlg::DoDataExchange(CDataExchange\* pDX)
2. {
3. // 处理MFC默认的数据交换
4. CDialogEx::DoDataExchange(pDX);
5. // 处理控件IDC\_SUMMAND\_EDIT和变量m\_editSummand之间的数据交换
6. DDX\_Text(pDX, IDC\_SUMMAND\_EDIT, m\_editSummand);
7. // 处理控件IDC\_ADDEND\_EDIT和变量m\_editAddend之间的数据交换
8. DDX\_Text(pDX, IDC\_ADDEND\_EDIT, m\_editAddend);
9. // 处理控件IDC\_SUM\_EDIT和变量m\_editSum之间的数据交换
10. DDX\_Text(pDX, IDC\_SUM\_EDIT, m\_editSum);
11. }

鸡啄米再以Addition程序为例简单说下数据交换机制。如果我们在程序运行界面中输入被加数，则通过CAddition的DoDataExchange()函数可以将输入的值保存到m\_editSummand变量中，反之如果程序运行中修改了变量m\_editSummand的值，则通过CAddition的DoDataExchange()函数也可以将新的变量值显示到被加数的编辑框中。

但是这种数据交换机制中，DoDataExchange()并不是被自动调用的，而是需要我们在程序中调用CDialogEx::UpdateData()函数，由UpdateData()函数再去自动调用DoDataExchange()的。

CDialogEx::UpdateData()函数的原型为：

BOOL UpdateData(BOOL bSaveAndValidate = TRUE);

参数：bSaveAndValidate用于指示数据传输的方向，TRUE表示从控件传给变量，FALSE表示从变量传给数据。默认值是TRUE，即从控件传给变量。

返回值：CDialogEx::UpdateData()函数的返回值表示操作是否成功，成功则返回TRUE，否则返回FALSE。

在下一讲中鸡啄米将具体演示CDialogEx::UpdateData()函数如何使用。

鸡啄米本节主要讲的是新建对话框类和添加控件变量，控件的消息处理函数将在下一讲详细介绍。依然欢迎大家常回鸡啄米博客学习和讨论。

# 九、对话框：为控件添加消息处理函数

创建对话框类和添加控件变量在上一讲中已经讲过，这一讲的主要内容是如何为控件添加消息处理函数。

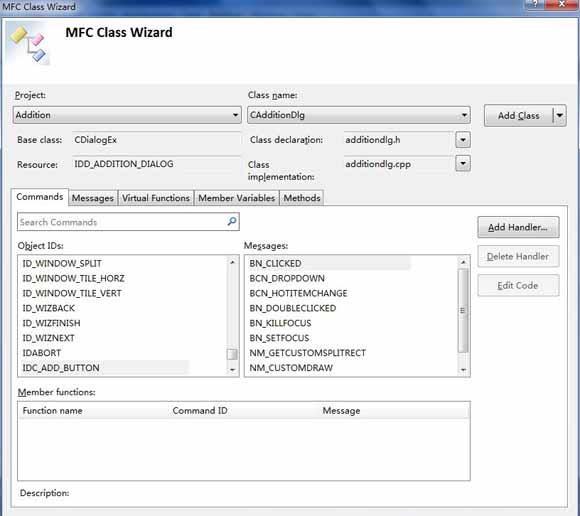
MFC为对话框和控件等定义了诸多消息，我们对它们操作时会触发消息，这些消息最终由消息处理函数处理。比如我们点击按钮时就会产生BN\_CLICKED消息，修改编辑框内容时会产生EN\_CHANGE消息等。一般为了让某种操作达到效果，我们只需要实现某个消息的消息处理函数。

## **一、添加消息处理函数**

鸡啄米仍以前面的加法计算器的程序为例，说明怎样为“计算”按钮控件添加消息处理函数。添加方法列出4种：

### **1.使用Class Wizard添加消息处理函数**

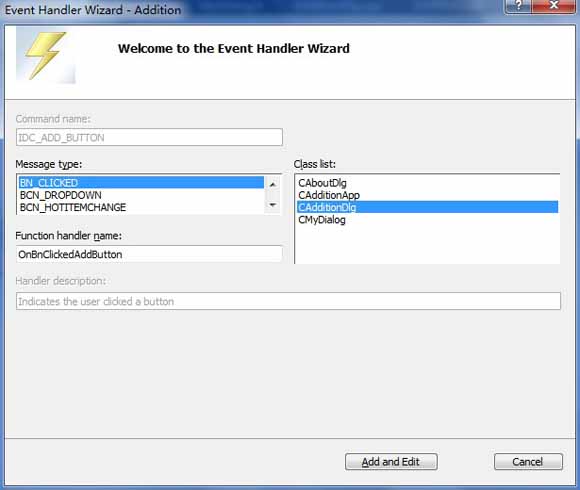
用过的VC++ 6.0的朋友应该对Class Wizard很熟悉了，添加类、消息处理函数等经常会用到它，可以说是一个很核心的功能。但从VS2002开始就见不到Class Wizard了，大部分功能都集成到对话框和控件等的属性中了，使用很方便。到VS2010，久违的Class Wizard又回来了。但鸡啄米已经习惯了使用属性中的功能了，对于从VC++ 6.0直接转VS2010的朋友可能觉得还是使用Class Wizard比较习惯。



大家应该记得，“计算”按钮的ID为IDC\_ADD\_BUTTON，上图中Commands标签下，Oject IDs列表中有此ID，因为我们是想实现点击按钮后的消息处理函数，所以在Messages列表中选择BN\_CLICKED消息，然后点右上方的Add Handler就可以添加BN\_CLICKED消息处理函数OnClickedAddButton了。当然你也可以改名，但一般用的默认的就可以。

### **2.通过“Add Event Handler...”添加消息处理函数**

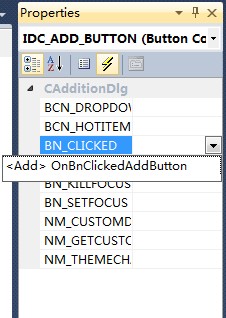
在“计算”按钮上点右键，然后在右键菜单中选择菜单项“Add Event Handler...”，弹出“Event Handler Wizard”对话框，如下图：



可见“Message type”中默认选中的就是BN\_CLICKED消息，函数名和所在类都已经自动给出，直接点“Add and Edit”就可以了。

### **3.在按钮的属性视图中添加消息处理函数**

上面说过，从VS2002开始就主要从属性视图添加消息处理函数了。我们在“计算”按钮上点右键，在右键菜单中选择“Properties”，右侧面板中会显示按钮的属性视图。



我们可以像上图中那样，点属性视图的“Control Events”按钮（类似闪电标志），下面列出了“计算”按钮的所有消息。我们要处理的是BN\_CLICKED消息，点其右侧空白列表项，会出现一个带下箭头的按钮，再点此按钮会出现“<Add> OnBnClickedAddButton”选项，最后选中这个选项就会自动添加BN\_CLICKED处理函数了。

### **4.双击按钮添加消息处理函数**

最直接最简单的方法就是，双击“计算”按钮，MFC会自动为其在CAdditionDlg类中添加BN\_CLICKED消息的处理函数OnBnClickedAddButton()。

## **二.在消息处理函数中添加自定义功能**

在我们使用任意一种方法添加了消息处理函数以后，都只能得到一个空的OnBnClickedAddButton()函数的函数体，要实现我们想要的功能，还需要在函数体中加入自定义功能代码。

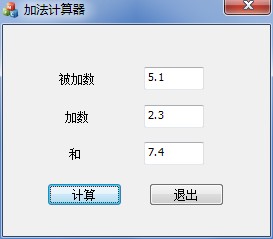
在加法计算器程序中，我们想要“计算”按钮实现的功能是，获取被加数和加数的数值，然后计算它们的和并显示到和的编辑框里。那么，OnBnClickedAddButton()的函数体就应修改为：

C++代码

1. void CAdditionDlg::OnBnClickedAddButton()
2. {
3. // TODO: Add your control notification handler code here
4. // 将各控件中的数据保存到相应的变量
5. UpdateData(TRUE);
7. // 将被加数和加数的加和赋值给m\_editSum
8. m\_editSum = m\_editSummand + m\_editAddend;
10. // 根据各变量的值更新相应的控件。和的编辑框会显示m\_editSum的值
11. UpdateData(FALSE);
12. }

鸡啄米在上面的代码中已经添加注释，大家应该很容易理解了。对于UpdateData()函数的说明在上一讲中已经介绍过，如果忘了可以再回上一讲了解了解。

接下来我们运行下此应用程序。在运行结果界面中，输入被加数5.1，加数2.3，然后点“计算”：



 在上图中可以看到，点“计算”按钮后，和的编辑框中显示了正确结果：7.4。

鸡啄米简单分析下运行过程：输入被加数和加数，点“计算”按钮后产生点击消息，从而调用OnBnClickedAddButton()函数。进入此函数后，首先由UpdateData(TRUE)函数将被加数的值5.1和加数的值2.3分别保存到变量m\_editSummand和m\_editAddend，然后通过语句m\_editSum = m\_editSummand + m\_editAddend;计算出被加数和加数的和为7.4，并把7.4赋值给m\_editSum。最后调用UpdateData(FALSE)根据被加数、加数、和的值更新三个编辑框的显示值，就得到了上图中的结果。

到此，一个具有简单的加法运算功能的加法计算器应用程序就基本完成了。如果大家想实现其他功能，可以修改控件资源和消息处理函数来练习下。本节就讲到这里了，有问题欢迎到鸡啄米博客或者我们的编程入门qq群讨论。

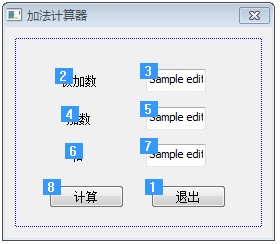
# 十、对话框：设置对话框控件的Tab顺序

前面几节鸡啄米为大家演示了加法计算器程序完整的编写过程，本节主要讲对话框上控件的Tab顺序如何调整。

上一讲为“计算”按钮添加了消息处理函数后，加法计算器已经能够进行浮点数的加法运算。但是还有个遗留的小问题，就是对话框控件的Tab顺序问题。

运行加法计算器程序，显示对话框后不进行任何操作，直接按回车，可以看到对话框退出了。这是因为“退出”按钮是Tab顺序为1的控件，也就是第一个接受用户输入的控件。但是按照我们的输入习惯，应该是被加数的编辑框首先接受用户输入，然后是加数编辑框，再接下来是“计算”按钮，最后才是“退出”按钮。

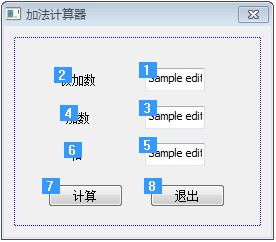
我们先来直观的看看各个控件的Tab顺序吧。打开“Resource View”视图，然后在资源中找到对话框IDD\_ADDITION\_DIALOG，双击ID后中间客户区域出现其模板视图。在主菜单中选择“Format”->"Tab Order"，或者按快捷键Ctrl+D，对话框模板上就会显示各个控件的Tab顺序数字。如下图：



上图中每个控件左上角都有一个数字，这就是它的Tab响应顺序。对话框刚打开时输入焦点就在Tab顺序为1的“退出”按钮上，不做任何操作按下Tab键，输入焦点就会转移到Tab顺序为2的“被加数”静态文本框上，但是因为静态文本框不接受任何输入，所以输入焦点继续自动转移到Tab顺序为3的被加数编辑框，再按Tab键，输入焦点又会转移到Tab顺序为4的“加数”静态文本框上，同样由于它是静态文本框，输入焦点不停留继续转移到加数编辑框，后面的控件同理。

我们认为这个顺序不合理，那怎么修改呢？很简单，从自己认为Tab顺序应该为1的控件开始依次单击，随着单击的完成，各控件的Tab响应顺序也按我们的想法设置好了。

例如，此例中我们可以依次单击被加数编辑框、“被加数”静态文本框、加数编辑框、“加数”静态文本框、和编辑框、“和”静态文本框、“计算”按钮和“退出”按钮。设置完后如下图：



最后按ESC键，确认设置并退出对话框模板的Tab顺序设置状态。

现在我们再运行程序，可以看到对话框打开后最初的输入焦点在被加数编辑框上，然后我们按Tab键，输入焦点移到加数编辑框上，继续多次按Tab键时，输入焦点会按“和编辑框--‘计算’按钮--‘退出’按钮--被加数编辑框--加数编辑框--和编辑框......”的顺序循环转移。这样就达到了我们的目的。

本节教程内容比较简单，相信大家很快就能掌握。依然欢迎大家在鸡啄米博客留言或到我们的编程入门群讨论。

# 十一、对话框：模态对话框及其弹出过程

加法计算器对话框程序大家照着做一遍后，相信对基于对话框的程序有些了解了，有个好的开始对于以后的学习大有裨益。趁热打铁，鸡啄米这一节讲讲什么是模态对话框和非模态对话框，以及模态对话框怎样弹出。

## **一、模态对话框和非模态对话框**

## **二、模态对话框是怎样弹出的**

## **三、添加一个新对话框并弹出它**

# 十二、对话框：非模态对话框的创建及显示

# 十三、对话框：属性页对话框及相关类的介绍

# 十四、对话框：向导对话框的创建及显示

# 十五、对话框：一般属性页对话框的创建及显示

# 十六、对话框：消息对话框

# 十七、对话框：文件对话框

# 十八、对话框：字体对话框

# 十九、对话框：颜色对话框

# 二十、常用控件：静态文本框

# 二十一、常用控件：编辑框Edit Control

# 二十二、常用控件：按钮控件Button、Radio Button和Check Box

# 二十三、常用控件：按钮控件的编程实例

# 二十四、常用控件：列表框控件ListBox

# 二十五、常用控件：组合框控件Combo Box

# 二十六、常用控件：滚动条控件Scroll Bar

# 二十七、常用控件：图片控件Picture Control

# 二十八、常用控件：列表视图控件List Control 上

# 二十九、常用控件：列表视图控件List Control 下

# 三十、常用控件：树形控件Tree Control 上

# 三十一、常用控件：树形控件Tree Control 下

# 三十二、常用控件：标签控件Tab Control 上

# 三十三、常用控件：标签控件Tab Control 下

# 三十四、菜单：VS2010菜单资源详解

上一节讲了[标签控件Tab Control](http://www.jizhuomi.com/software/207.html)以后，常用控件的内容就全部讲完了，当然并没有包括所有控件，主要是一些很常用很重要的控件。本节开始鸡啄米将为大家讲解菜单的概念及使用。

## **菜单简介**

菜单在界面设计中是经常使用的一种元素，包括Windows系统中的窗口、智能终端设备的应用界面等都会经常见到菜单的身影。我们在对可视化窗口操作时，菜单确实提供了很大方便。

菜单可以分为下拉式菜单和弹出式菜单。

下拉式菜单一般在窗口标题栏下面显示，大家还记得我们在[VS2010/MFC编程入门之二（利用MFC向导生成单文档应用程序框架）](http://www.jizhuomi.com/software/141.html)中创建的HelloWorld单文档工程吗？它的运行结果窗口的标题栏下就是下拉式菜单。下拉式菜单通常是由主菜单栏、子菜单及子菜单中的菜单项和分隔条所组成的。

弹出式菜单一般可以通过单击鼠标右键等操作显示。它的主菜单不可见，只显示子菜单。

# 三十五、菜单：菜单及CMenu类的使用

# 三十六、工具栏：工具栏资源及CToolBar类

# 三十七、工具栏：工具栏的创建、停靠与使用