



Visual C++使用方法简介

Visual C++ 菜单栏

Visual C++6.0的菜单分成9大类:

[File] 文件	[Edit] 编辑	[View] 视图
[Insert] 插入	[Project] 工程	[Build] 组建
[Tools] 工具	[Window] 窗口	[Help] 帮助

在程序运行时， Visual C++ 6.0的菜单栏可以动态改变，还有 [Layout] 和 [Debug]。

如在调试状态下， [Build] 变成了 [Debug]。

[File] 文件菜单

[File] 菜单共有14选项，分成6组：

- 1、[New]新建一个一般文件，工程，工作区，其他文档。
[Open] 打开、[Close] 关闭
- 2、Workspace工作区操作，打开、保存和关闭工作区。
- 3、有三个菜单项，用于文件保存。
- 4、有二个菜单项，用于文件打印。
- 5、用于打开以前打开过的文件或工作区。
- 6、一个菜单项Exit，用于退出Visual C++ 6.0。

[Edit]编辑菜单

[Edit] 菜单分成7组:

- 1、撤销编辑结果，或重复前次编辑过程。
- 2、提供常见的编辑功能。
- 3、字符串查找和替换。
- 4、Go to和Bookmark编辑行定位和书签定位。
- 5、Advanced (高级)，一些其他编辑手段。
- 6、Breakpoints，与调试有关，主要用于设置断点。
- 7、成员列表、函数参数信息、类型信息，及自动完成功能。

[View]视图菜单

[View]菜单共有9个选项，分成6组（初始时没有1和7）：

- 1、ClassWizard（或Ctrl+W），激活MFC ClassWizard 类向导工具，用来管理类、消息映射等。
- 2、Resource Symbols 对工程所定义的所有资源标号，进行浏览和管理。
- 3、Resource Includes 用于设定资源ID的包含头文件。
- 4、Full Screen 全屏显示，按Esc退出全屏显示。
- 5、Workspace 显示工作区窗口。
- 6、Output 显示输出窗口。
- 7、Debug Windows 在调试状态下控制一些调试窗口。
- 8、Refresh 刷新当前显示窗口。
- 9、Properties 查看和修改当前窗口所显示的对象属性。5

[Insert]插入菜单

[Insert] 菜单共有6个选项：

- 1、New Class 添加新类（MFC 、Generic、Form三种不同类型的类）。
- 2、New Form 添加Form Class。
- 3、Resource添加资源。
- 4、Resource Copy添加资源复制件。
- 5、File As Text 插入选定的文本文件。
- 6、New ATL Object添加ATL对象。

[Project]工程菜单

[Project]菜单共有6个选项:

- 1、Set Active Project 在多个工程中选定当前活动工程。
- 2、Add to Project 向当前工程添加文件、文件夹、数据连接、Visual C组件, 以及ActiveX控件。
- 3、source Control源代码控制具。
- 4、Dependencies设置工程间的依赖关系。
- 5、Settings 设置工程属性(调试版本、发布版本和共同部分)。
- 6、Export Makefile 导出应用程序的Make (*.mak)文件。

[Build]构建菜单

[Build]菜单共有13个选项:

- 1、Compile 编译当前文件。
- 2、Build 创建工程的可执行文件，但不运行。
- 3、Rebuild All重新编译所有文件，并连接生成可执行文件。
- 4、Batch Build成批编译、连接工程的不同设置。
- 5、Clean把编译、连接生成的中间文件和最终可执行文件删除。
- 6、Start Debug->Go 开始调试，到断点处暂停。
- 7、Start Debug->Step Into单步调试，遇函数进入函数体。
- 8、Start Debug ->Run to Cursor开始调试，到光标处停止。
- 9、Debugger Remote Connection用于远程连接调试。
- 10、Execute运行可执行目标文件。
- 11、Set Active Configuration选择Build配置方式（Debug、Release）。
- 12、Configuration增加或删除工程配置方式。
- 13、Profile工程构建过程的描述文件。

[Tools] 工具菜单

[Tools] 菜单中是Visual C++附带的各种工具。

其中常用的工具有

ActiveX Control Test Container（测试一个ActiveX控件的容器）、

Spy++（用于程序运行时以图形化方式查看系统进程、线程、窗口、窗口信息等），

以及MFC Tracer（用于程序跟踪）等。

还有一些常用的设置：Customize, Options。

[Windows]窗口菜单

[Windows]菜单主要工能如下:

- 1、New Window新建一个窗口，内容与当前窗口同。
- 2、Split 分割当前窗口成四个，内容全相同。
- 3、Docking View控制当前窗口是否成为浮动视图。
- 4、Cascade编辑窗口层叠放置。
- 5、Tile Horizontally编辑窗口横向平铺显示。
- 6、Tile Vertically编辑窗口纵向平铺显示。
- 7、Windows对已经打开的窗口进行集中管理。

[Help]窗口菜单

[Help]菜单中的4个选项Contents、Search、Index和Technical Support 都会弹出帮助窗口，叫做MSDN Library Visual Studio6.0。

MSDN库提供的帮助工能很丰富，可以以目录、索引和搜索三种方式提供帮助。浏览方式多样，甚至可以连接到Web网站查找信息。

另有两个选项：

Keyboard Map选项打开快捷键列表；

Tip of the Day选项打开Tip of the Day对话框，介绍Visual C++6.0的使用知识和技巧。

工具栏

工具栏由多个操作按钮组成，这些操作一般都与某个菜单项对应。主要工具栏如下：

- 1、Standard提供最基本功能：文件操作、编辑、查找等。
- 2、Build工程的编译、连接、修改活动配置、运行调试程序。
- 3、Build MiniBar由部分按钮组成的工具栏。
- 4、Resource添加各种类型的资源。
- 5、Edit剪切、复制和粘贴等功能。
- 6、Debug用于调试状态的若干操作
- 7、Browse源程序浏览操作
- 8、Database跟数据库有关的操作。

Visual C++组件一览

- Developer Studio开发环境
- 编辑器
- 编译器
- 链接器
- Wizard实用程序
- 调试器
- 其他实用工具

Developer Studio 开发环境

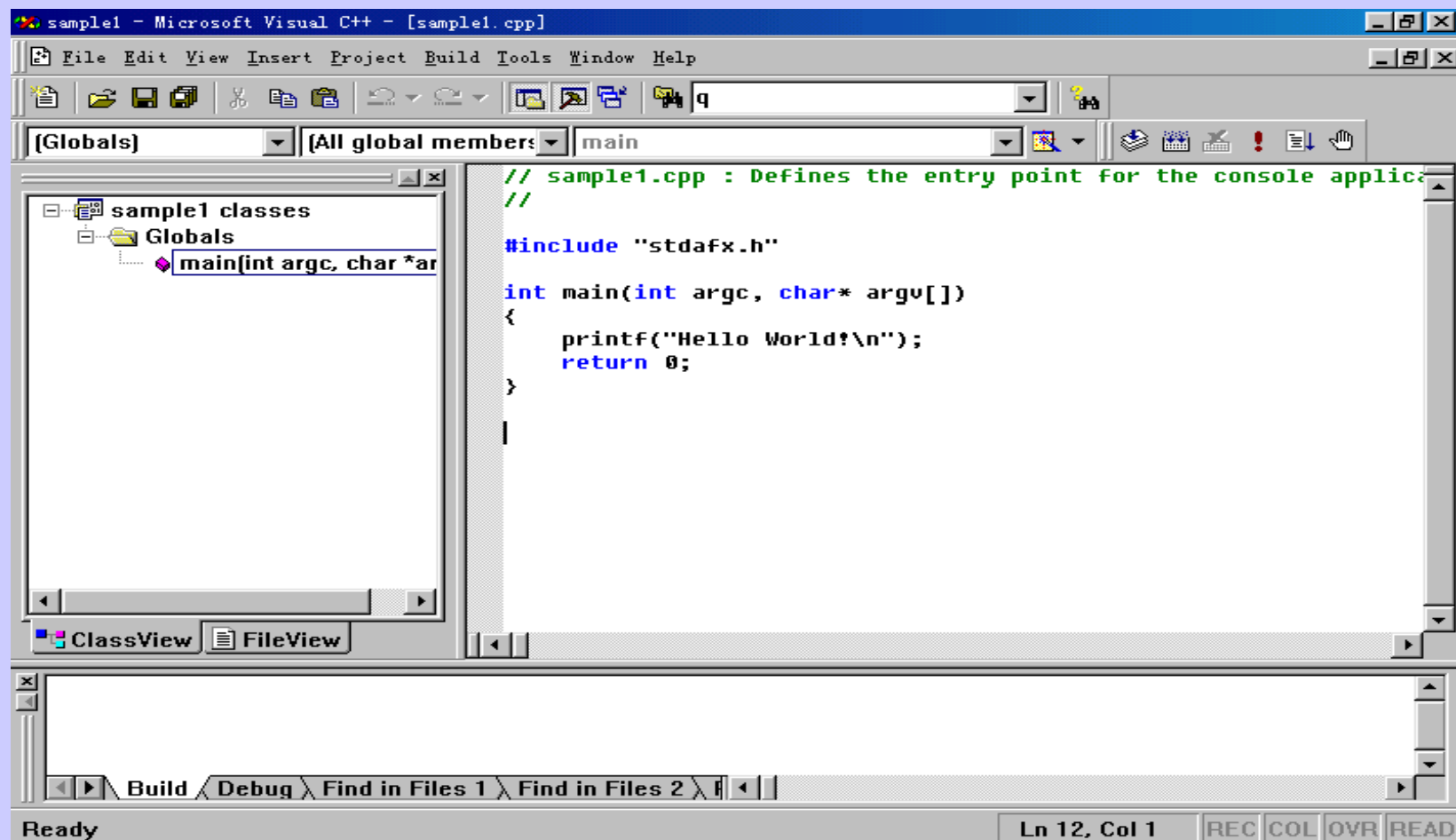


图1

了解Developer Studio

- Workspace（工作空间）窗口
- Output（输出）窗口
- 编辑窗口
- 调试窗口

工作空间窗口

Workspace窗口显示了项目各个方面的信息。在窗口底端选择相应的选项卡来按不同视图显示项目的列表。

- **ClassView:** 列出项目中的类和成员函数。双击列表中的类或函数，即可在Visual C++文本编辑器中打开该类的源文件。
- **ResourceView:** 列出项目的资源数据，双击列表中的数据项会打开合适的编辑器并加载资源。
- **FileView:** 列出项目的源文件，头文件。

工作空间和项目

- 工作空间 (workspace): 工作空间是一个包含用户的所有相关项目和配置的实体。
- 项目 (project): 项目定义为一个配置和一组文件, 用以生成最终的程序或二进制文件。一个工作空间可以包含多个项目, 这些项目既可以是同一类型的项目, 也可以是由不同类型的项目 (如Visual C++和Visual J++项目)。

编辑窗口

编辑窗口为开发者提供了编辑文件和资源的手段。通过编辑窗口，开发者可以编辑和修改源程序和各种类型的资源。

资源

资源包括菜单、对话框、图标、字体、快捷键等。开发者可以通过编辑资源来定义WINDOWS程序的界面部分。

资源的定义是以文本的形式存放在资源定义文件中，并由编译器编译为二进制代码。

在VC++中，提供了一个资源编辑器，使开发者能在图形方式下对各种资源进行编辑。

资源编辑器

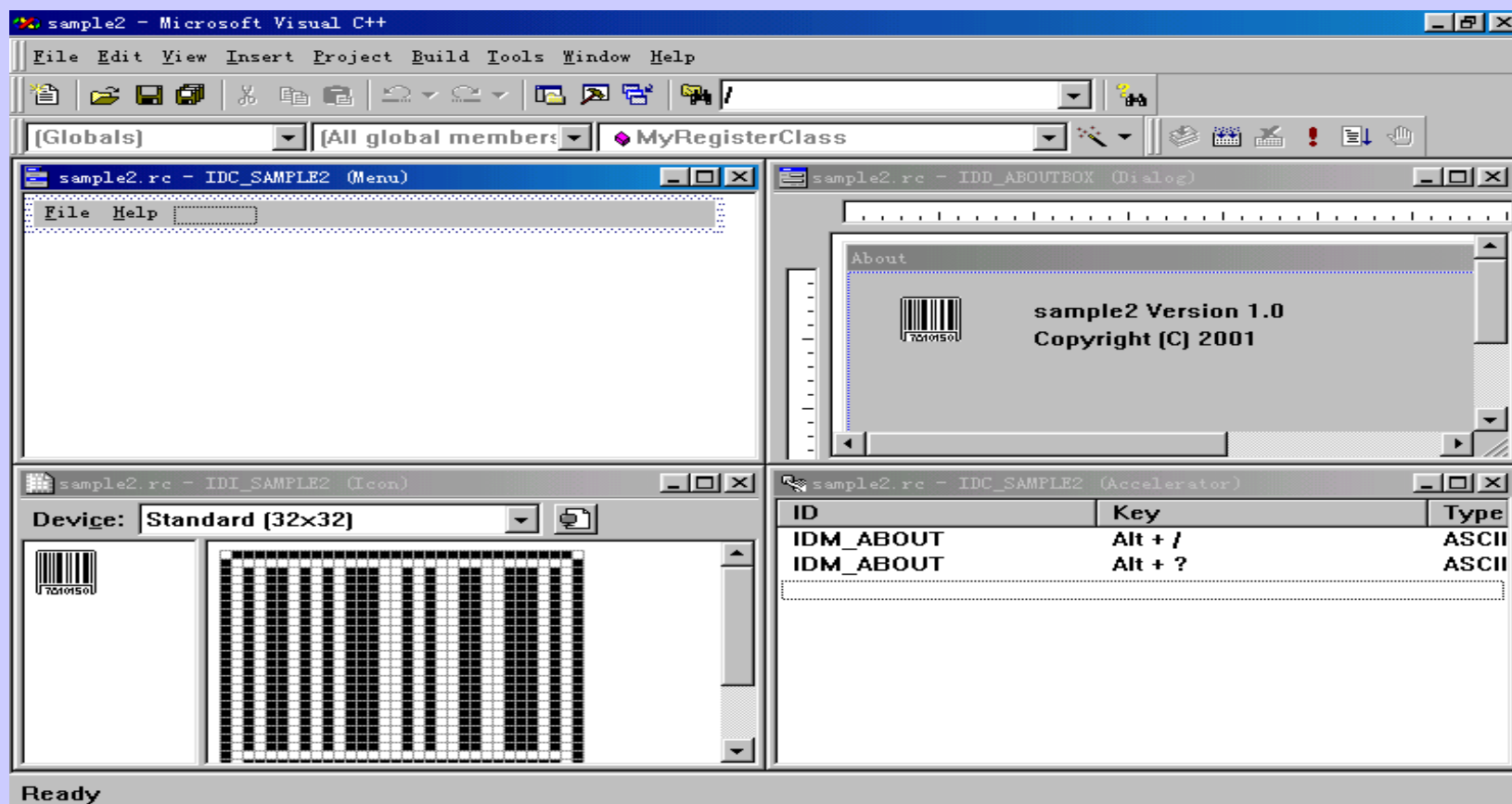


图2

输出窗口

输出窗口输出一些用户操作后的反馈信息，它由一些页面组成，每个页面输出一种信息，输出的信息种类主要有：

- 编译信息：在编译时输出，主要是编译时的错误和警告。
- 调试信息：在对程序进行调试时输出，主要是程序当前的运行状况。
- 查找结果：在用户从多个文件中查找某个字符串时产生，显示查找结果的位置。

调试窗口

调试窗口包括一组窗口，在调试程序时分别显示各种信息，这些窗口主要包括：

- 变量查看窗口 (WATCH)
- 过程调用查看窗口 (CALL STACK)
- 内存查看窗口 (MEMORY)
- 寄存器查看窗口 (REGISTER)

用VC++环境下运行一个新程序的上机操作步骤

1. 打开VC++窗口，单击菜单[File]->[New]
2. 在弹出窗口上选择标签Project，选中Win32 Console Application项，并在Project name框输入工程名；在Location框输入保存源程序的路径名；单击OK。
3. 在弹出窗口选择An Empty Project，单击[Finish]按钮；在下一窗口单击OK。
4. 再在VC++菜单条中单击[File]->[New]。

5. 在弹出窗口选择标签File, 选C++ Source File, 并在File框输入文件名。(此时, Add to project框应该是刚输入的工程名, Location框应是刚输入的路径名).
 6. 在编辑窗口输入源程序, 单击[File]->[Save], 存盘.
 7. 再在VC++菜单条中单击[Build]->[Compile]; 无编辑错, 单击[Build] -> [Build]; 无连接错, 单击[Build]->[Execute]执行程序; 运行结束, 按任意键退回VC++窗口。
 8. 如果再键入另一个新程序, 单击[File]->[Close Workspace]; 然后再重复上述步骤。
- 编译一个已打开的程序也可简单地单击[Build All]的标志符。执行编译好的程序, 单击执行标志! 。

如要打开已存在的C++ .CPP源程序，可按以下步骤打开：

1. 打开VC++窗口，单击 [File] -> [Open]；
2. 在弹出对话框找文件所在文件夹，选中文件，单击打开按钮，把文件调入VC++编辑窗。
3. 在VC++菜单条中单击 [Build] -> [Compile]，单击是按钮。
4. 无编辑错，单击 [Build] -> [Build]；无连接错，单击 [Build] -> [Execute] 执行程序；运行结束，按任意键退回VC++窗口。

编译一个已打开的程序也可简单地单击 [Build All] 的标志符。执行编译好的程序，单击！

用VC++开发程序的过程

- 用App Wizard新建一个工程
- 编辑代码
- 编译代码
- 调试和执行程序

App Wizard

- App Wizard（应用程序生成器）是Visual C++自带的一个工具，通过它，可以方便地生成各种类型的程序的框架。
- 选择菜单中的File---->New...，即可以使用App Wizard来新建程序。
- 可以新建的内容包括File、Project、Workspaces、Other Documents四个页面，每个页面下有各种类型的工程或文件。
- 选定类型之后，即进入Wizard（向导），让用户选择一些可选项，完成之后，程序的框架即生成。

用App Wizard新建一个工程

1. 在File菜单上，点New，选择Projects标签。
2. 从列表中选择项目类型。
3. 点Create New Workspace (新建工作区) 或Add to Current Workspace (加入到当前工作区中)。
4. 要使新工程为子工程，可以选择Dependency of 检查框，并从列表中选择一个工程。
5. 在Project Name框中，输入新工程名，确保该名字必须与工作区中的别的工程名字不重名 。
6. 在Location框中，指定工程存放的目录：可以直接输入路径名，也可以按旁边的Browse按钮，浏览选择一个路径 。
7. 点Platform框中的相应检查框，指定工程的开发平台 。
8. 输入完以上内容并按OK按钮后，根据所选的工程类型，会出现相应的Wizard（向导）。通过一系列的对话框输入，快速生成工程的框架。

新建一个C++程序

1. 在File菜单上，点节New，选择File标签。
2. 从列表中选择C/C++源程序。
3. 在编辑窗键入源程序 。
4. 文件保存，源程序文件的名必需为.C或.CPP
5. 点节编译命令 。
6. 若程序有错，重新编辑源程序，改正错误后，重新编译。
7. 编译没有错误后，点节执行命令，程序开始执行。

继续新建一个C++程序

1. 在File菜单上，点节Close关闭编辑窗，点节Close Workspace关闭工作空间。
2. 然后重复新建一个C/C++程序的全部工作。

修正一个已有的C++程序

- 1、在File菜单上，点节Open，打开一个已有的C++程序。
- 2、在编辑窗修改源程序。
- 3、点节编译命令。
- 4、若程序有错，重新编辑源程序，改正错误后，重新编译。
- 5、编译没有错误后，点节执行命令，程序开始执行。

开始实践--第一个VC程序

1. 新建一个工程，在项目类型中选“Win32 Console Application”。
2. 在Project Name框中输入test1，将Create New Workspace选择框选中。
3. 按OK按钮。
4. 在出现的Wizard对话框中选择A Simple Application，然后按Finish按钮。
5. 在接下来出现的对话框中按OK。

编译运行程序

1. 选择菜单中的Build---->Build test1.exe
2. 在输出窗口会出现 “test1.exe - 0 error(s), 0 warning(s)”, 说明编译通过。
3. 选择菜单中的Build---->Execute test1.exe 。
4. 出现运行结果。

程序运行结果

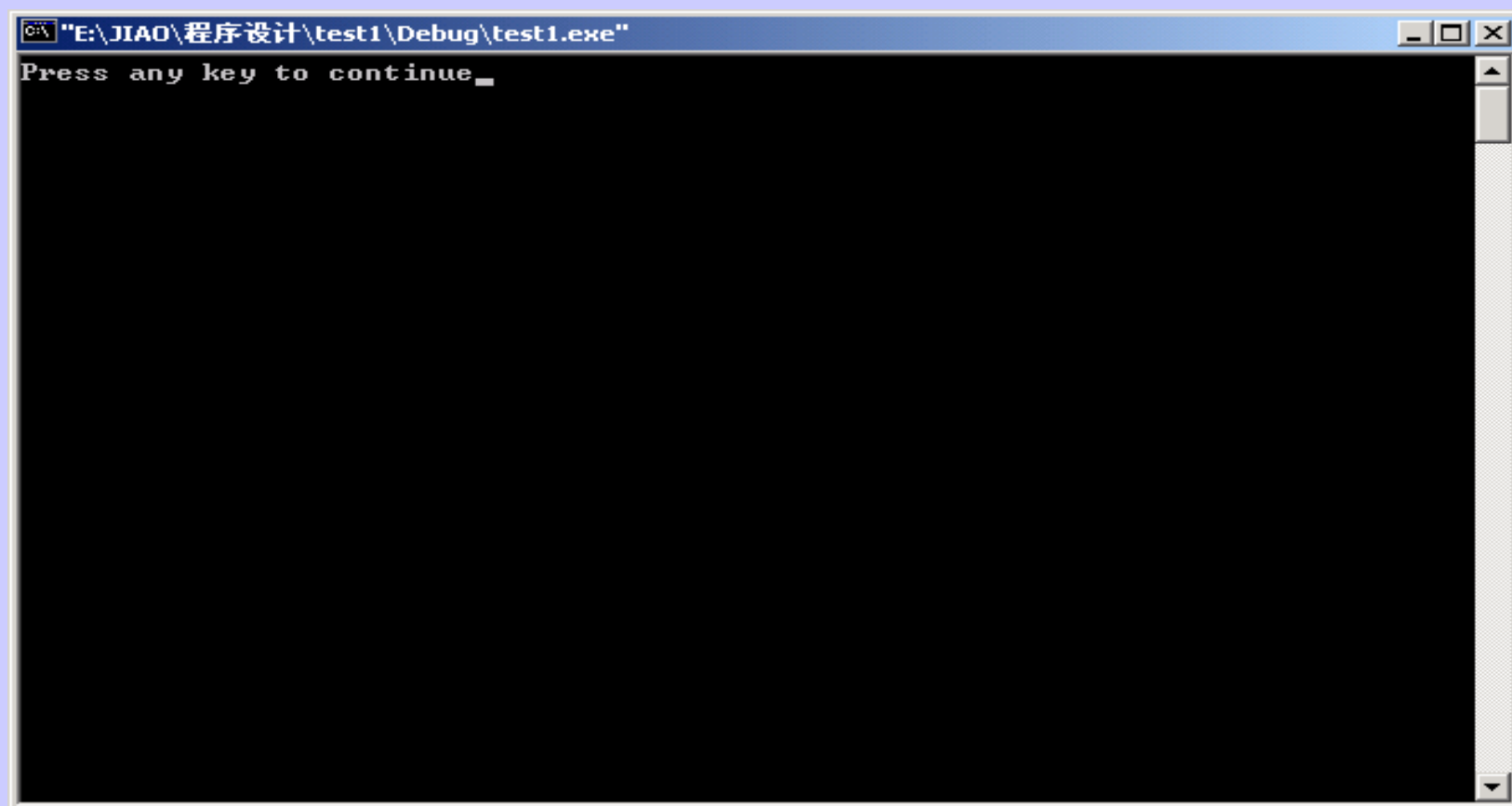


图3

编辑代码

1. 在工作空间窗口中选 ClassView 页面。
2. 双击Global下的main方法，右边的编辑窗口显示了main方法所在源文件的内容。
3. 在编辑窗口中的return 0 的前面插入一行
“`printf("this is my firstprogram!\n");`”。
4. 在#include “stdafx.h” 那行之后插入一行
`#include "stdio.h"`
5. 保存文件

重新编译并运行

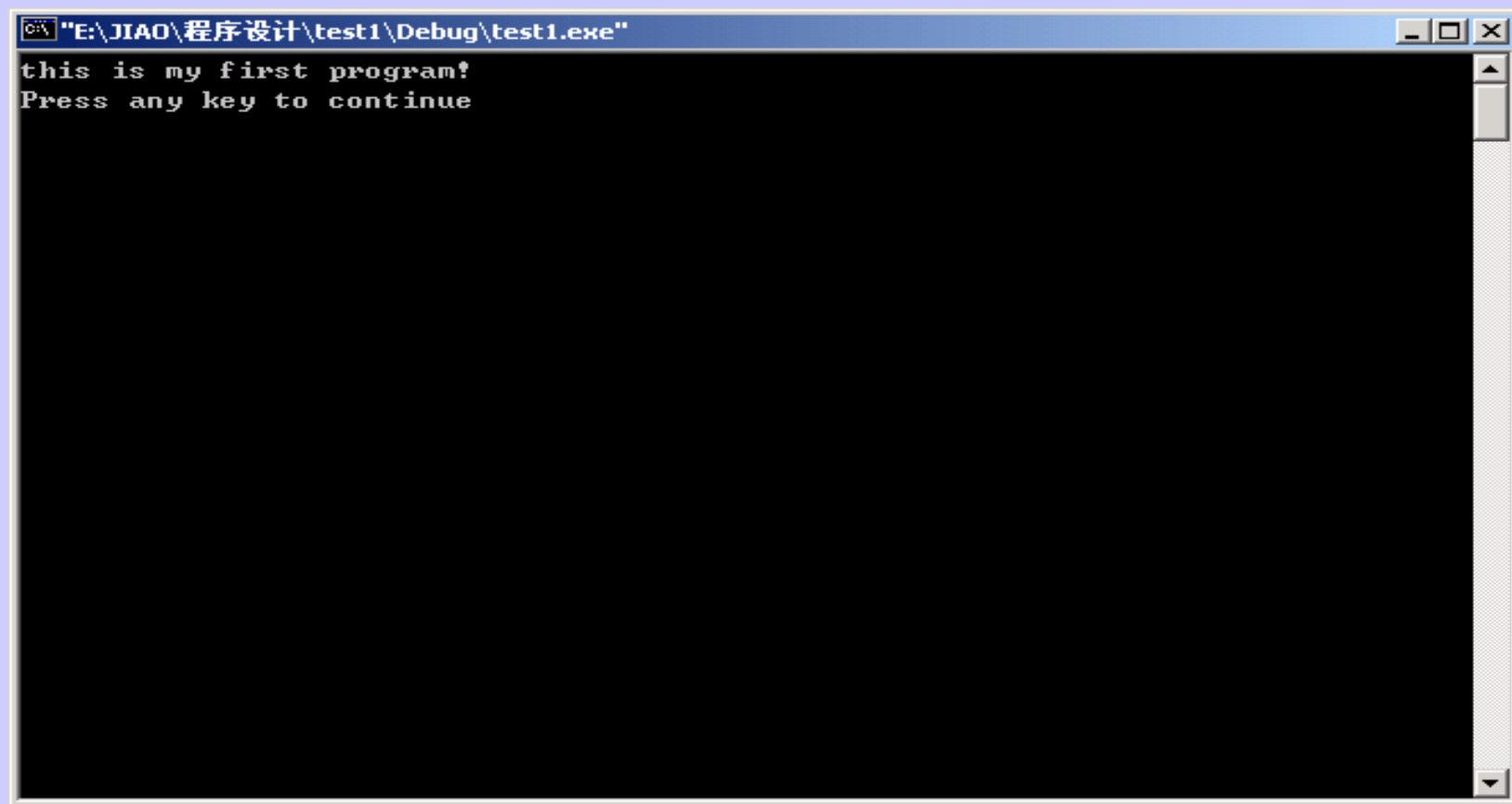
1. 选择菜单中的Build---->Build test1.exe
2. 如果在输出窗口出现

test1.exe - 0 error(s), 0 warning(s)

说明编译通过；如果显示有错误，则需要修改源文件直到编译通过。

1. 选择菜单中的Build---->Execute test1.exe 。
2. 出现运行结果。

修改过的程序运行结果



A screenshot of a Windows command prompt window. The title bar at the top reads "E:\JIAO\程序设计\test1\Debug\test1.exe". The window has standard Windows window controls (minimize, maximize, close) on the right. The main area is black with white text. The text displayed is:

```
this is my first program!  
Press any key to continue
```

The text is in a monospaced font. The window is open, and the text is visible on the first two lines.

图4

调试程序

编译错误是因为程序代码不符合C++语法、单词拼写错误、函数调用参数使用不当等，通过编译和检查程序能比较方便地改正。

若运行结果与预期结果不同，则需要用调试程序来找到程序中错误的地方，并排除所有的错误。

• 选择菜单中的Build--->Start Debug，启用调试器。
调试器有四个子菜单：

[Go]

[Step Into]

[Run to cursor]

[Attach to process]

- >Go 从当前语句开始执行，直到遇到断点，或程序执行结束。用Go启动调试器，从头开始执行程序。
- >Step Into 单步执行每一程序行，遇到函数调用进入函数体内单步执行。
- >Run to cursor 运行程序至当前光标位置。
- >Attach to process 将调试器与当前运行的某个进程联系起来，可跟踪进入进程内部，调试运行中的进程。

调试命令

菜单项	快捷键	作用
Go	F5	运行程序至断点，或程序结束
Restart	Ctrl+Shift+F5	重新载入程序，并启动执行
Stop Deb.	Shift+F5	关闭调试会话
Break		从当前位置退出，终止程序执行
Step Into	F11	单步执行，并进入调用函数
Step Over	F10	单步执行，但不进入函数
Step Out	Shift+F11	跳出当前函数，回到调用处
Run to Cursor	Ctrl+F10	运行至当前光标处
Exceptions		设置异常，可以选择遇到异常处停止， 或遇到未处理的异常处停止
Threads		线程调试，可以挂起、恢复、切换线程
Step Into Specific Function		直接进入函数，用于调试多层嵌套的函数

在View菜单下还提供一个Debug Windows菜单的几个子菜单，用于隐藏或显示与调试工作相关的一些窗口。

菜单项	快捷键	作用
Watch	Alt+3	显示窗口，用于观察和设置变量值
Variables	Alt+4	观察与当前函数相关的变量
Registers	Alt+5	观察微处理器的寄存器
Memory	Alt+6	观察未使用的内存块
Call Stack	Alt+7	显示调用栈，观察调用的函数
Disassembly	Alt+8	打开窗口显示汇编程序代码

断点是程序调试过程中暂时停止执行的地方。在断点处，可以观察、设置变量的值，检查程序是否按所期望的逻辑执行。

插入断点 在源程序窗口内任一程序行上按鼠标右键，从右键快捷菜单中选择[Insert/Remove Breakpoint]菜单项，就可以将当前语句行作为一个断点。在该语句行左边，有一个红色实心圆指示该行是一个断点。

删除断点 在有断点的语句行上按右键弹出快捷菜单，选择[Remove Breakpoint]菜单项，就可删除该断点。

禁止断点 在断点处的右键快捷菜单上选择[Disable Breakpoint]菜单项，暂时禁止该断点，该断点可能以后再用。该位置将变为用空心圆标记。

恢复断点 在禁止断点处，用右键快捷菜单选择[Enable Breakpoint]菜单项，恢复起用曾被禁止的断点。

开发窗口程序

1. 新建一个新工程，在项目类型中选“Win32 Application”。
2. 在Project Name框中输入test2，将Create New Workspace选择框选中。
3. 按OK按钮。
4. 在出现的Wizard对话框中选择A Typical Hello world Application，然后按Finish按钮。
5. 在接下来出现的New Project Information对话框中按OK。
6. 编译代码、运行代码。

窗口程序运行结果

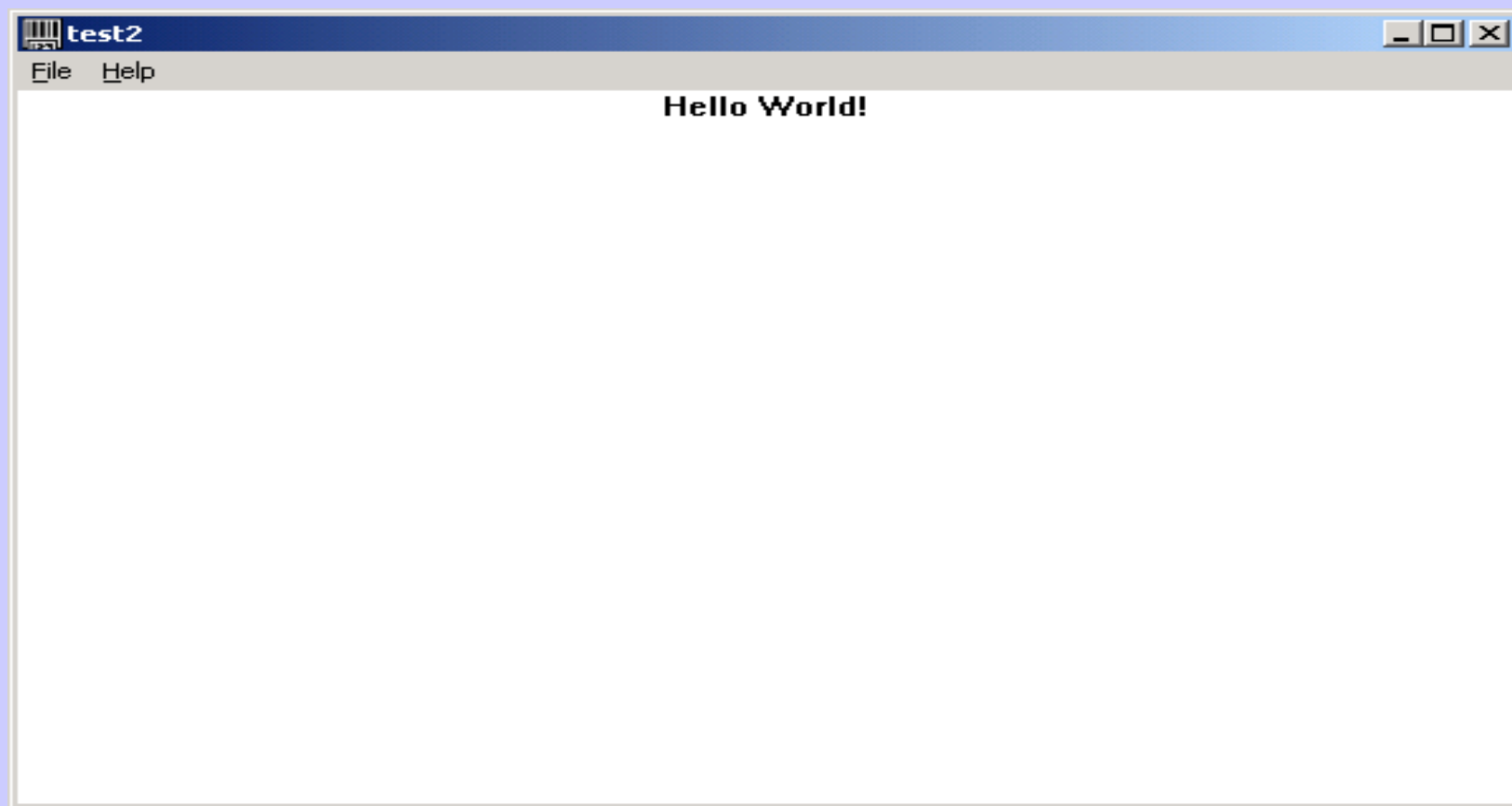


图5

获取帮助

- 为什么要获取帮助
 - 不可能也没有必要去记住所有的知识
 - 帮助文档可以使程序员事半功倍
- 获取帮助的途径
 - MSDN联机帮助
 - 参考书
 - INTERNET

MSDN联机文档



图6

使用MSDN

- Contents（目录）选项卡
 - 当你头脑中有一个大概的主题，而你又想要看一看这个主题有哪些文档时，目录表是非常有用的。
- Index（索引）选项卡
 - Index（索引）选项卡显示了整套MSDN文件的全面索引，只要打入关键字，就可以立刻找到你要找的东西。
- Search（搜索）选项卡
 - MSDN帮助文件集包含一个搜索引擎，它能进行全文本搜索来确定哪些主题文件包含了指定的词或短语。
- Favorites（收藏）选项卡
 - 它维护着一个标记选择过的文章的书签列表，使你能迅速找到你曾经看过的内容。

基于MFC 的GUI编程

1.Windows编程-1

- 事件驱动和消息机制是Windows编程的基础。
- Windows是消息驱动（或事件驱动）的操作系统。消息驱动意味着操作系统的每个部分与其它部分，以及应用程序之间通过Windows消息进行通信。例如当我们移动鼠标或按下键盘上某个键，Windows就会捕捉到这个消息，并存储到消息队列中等待处理（通常是转发给相应的操作系统某个处理程序或者应用程序）。

1.Windows编程-1

- 以前的软件开发者采用SDK (Windows Software Development Kit 软件开发工具包) 进行Windows编程。
- SDK的核心内容是API (应用程序编程接口)、事件驱动和消息循环。
- API是一个程序包, 其内有一组函数, 供程序员用它创建其他程序。Windows正是利用API实现图形用户界面 (GUI)。

1.Windows编程-2

- 选择为开发Windows应用而设计的精美的应用程序框架（Application framework）。Application framework是一个完整的程序模型，具备标准应用软件所需的一切基本功能，如文件存取、打印预览。
- Application framework提供了标准的程序模型，我们只需要按个人需要添加一些材料：在派生类中改写虚拟函数，或在派生类中加上新的程序函数。微软的MFC是一种十分成功的Application framework。

1.Windows编程-2

- MFC (Microsoft Foundation Classes 微软基础类) 是一个建立在Windows API基础上的C++类库，目的是使Windows程序设计过程更有效率。我们可以把MFC看作一个零组件“超级市场”。这些零组件（类）功能以及彼此间的关系都已经定义好，我们可以从中选择自己需要的零件构造出一个应用程序。
- 静态情况下MFC是一组类，在程序执行期间就生成一组有活动力的对象组。MFC程序没有main函数这样的入口，它的执行由application object（一个派生自MFC CWinApp的全局对象）引发，引发后我们选用的MFC类就依次实例化开始运行了。

2. MFC AppWizard

- AppWizard (Application Wizard) 俗称“应用程序创建向导”。
- 使用MFC AppWizard可以创建基于MFC类库的Windows应用程序（可执行文件.exe或者动态连接库.dll）。MFC AppWizard提供一系列对话框，用户可以选择满足需要的选项。定义完应用程序和项目选项后，MFC AppWizard生成创建应用程序所需要的初始框架文件。
- MFC AppWizard可以创建三种类型的应用程序：
 1. Single documents（单文档 SDI）：一次只允许打开一个子文档窗体。
 2. Multiple documents（多文档 MDI）：允许打开多个子文档窗体。
 3. Dialog-Based（基于对话框）：基于对话框的应用程序将显示一个对话框供用户输入或者进行选择。
- 三种类型应用程序的示例如下：

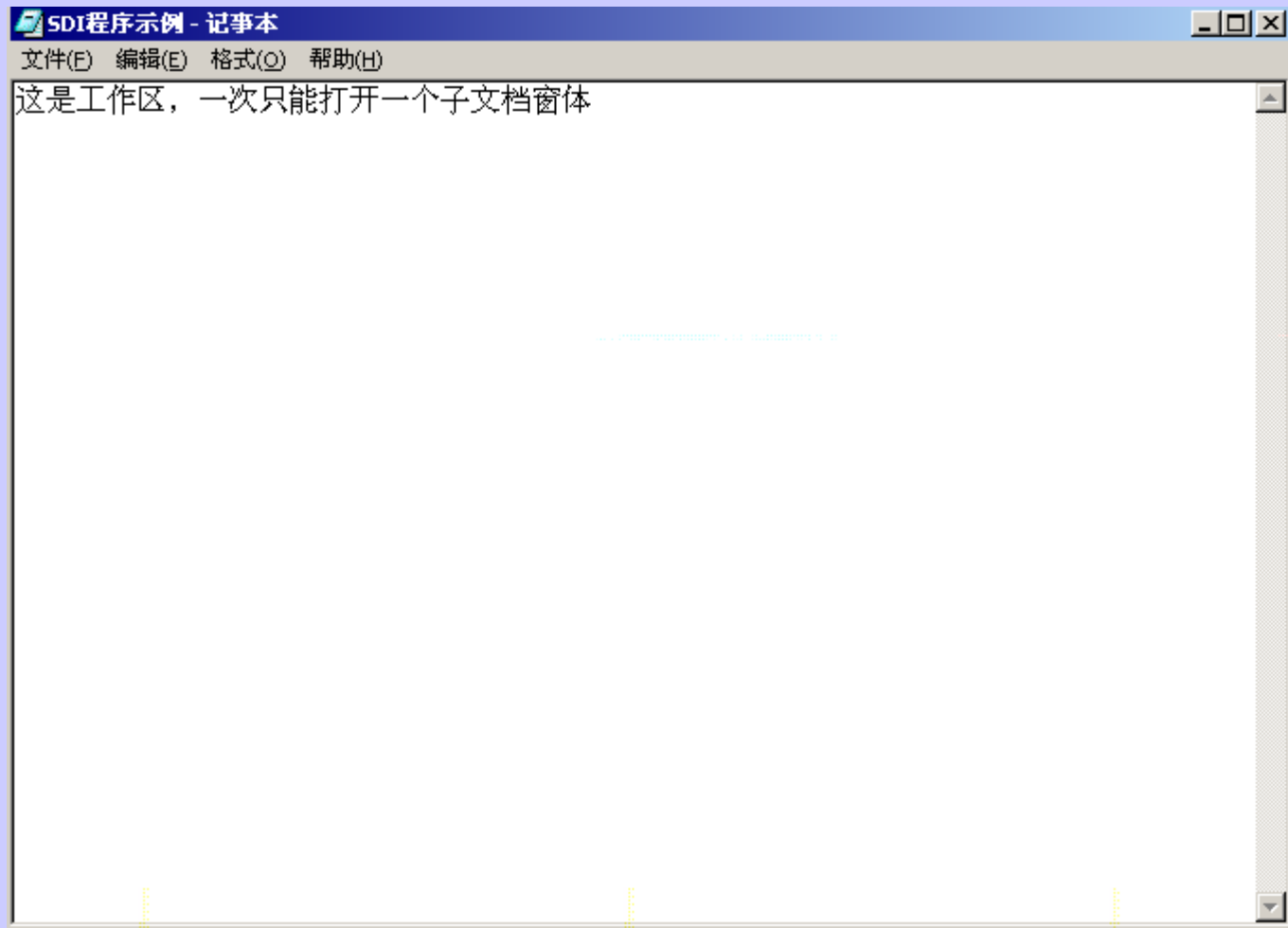


图7 SDI程序示例（Windows记事本）

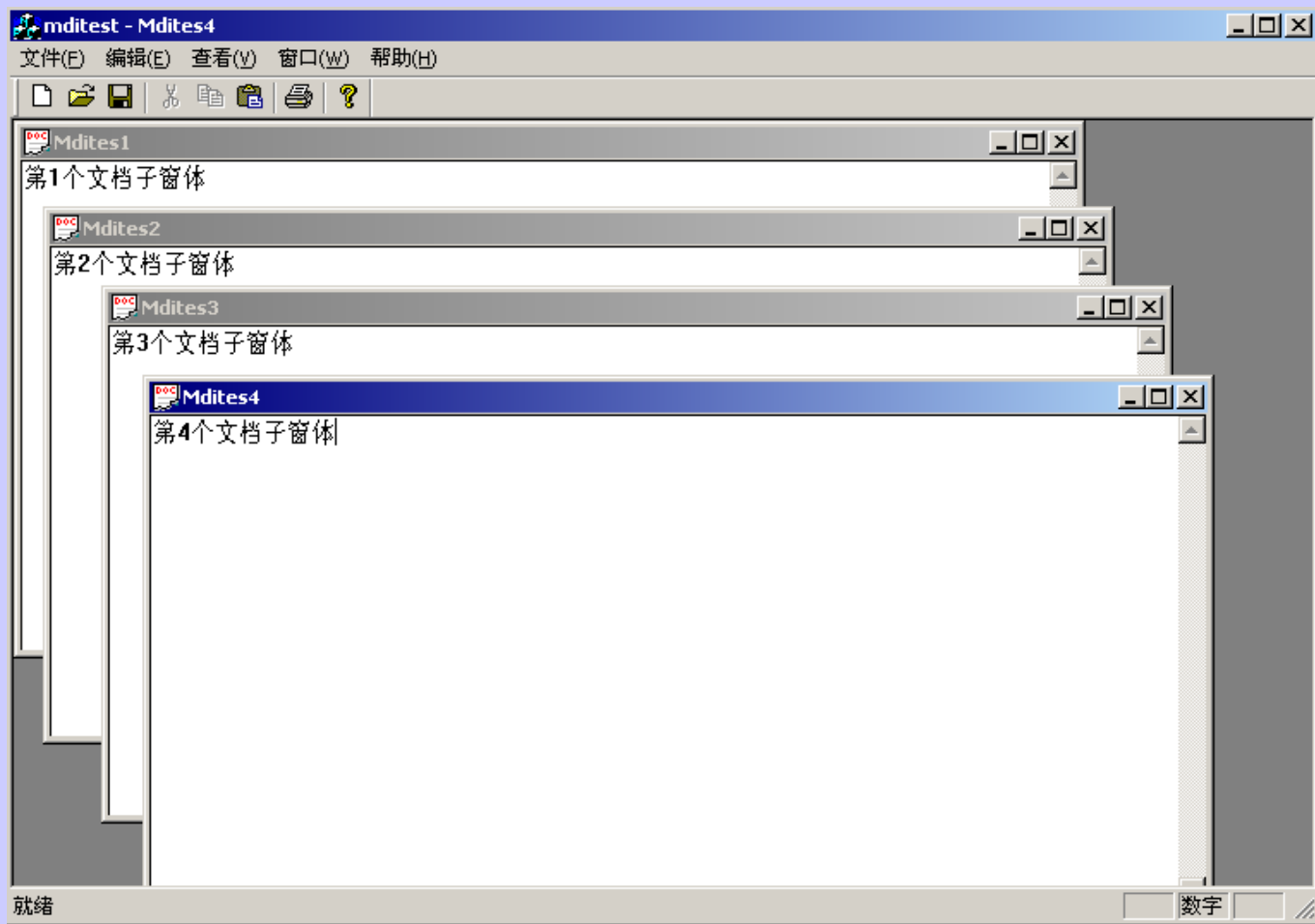


图8 MDI程序示例（又如微软的Word）

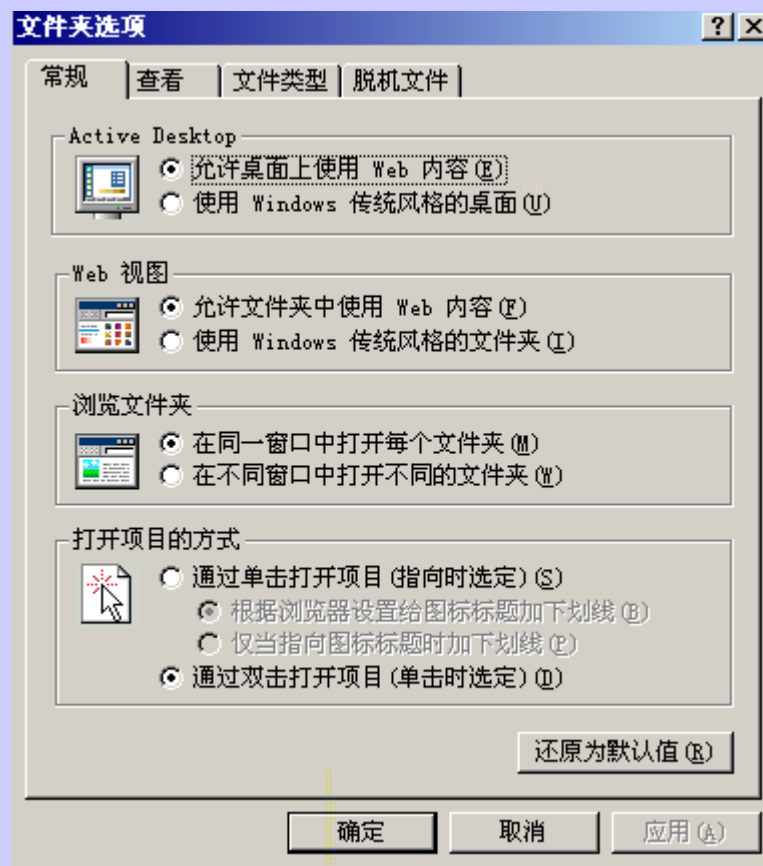


图9 Dialog-Based程序示例 (Windows我的电脑中文件夹选项设置窗口)

3. 一个简单的多文档应用程序实例

利用编程工具MFC AppWizard。不需要增添一句代码就可获得标准的多文档应用程序框架，包括打印等高级功能。

程序运行界面如图8所示。

首先新建project，选择的工程类型是：MFC AppWizard (exe)，示例项目名可自己设定，这里为mditest。

接下去的step-1的选项中，创建的程序类型选择 Multiple documents（多文档），其余选项使用缺省值。

Document/View architecture support?（文档/视图结构支持）选项缺省已经是选中状态。这一项如果不选中，则程序运行时不支持文档的打开/保存操作（需要文档/视图结构支持才有效）。

语言支持选择中文。

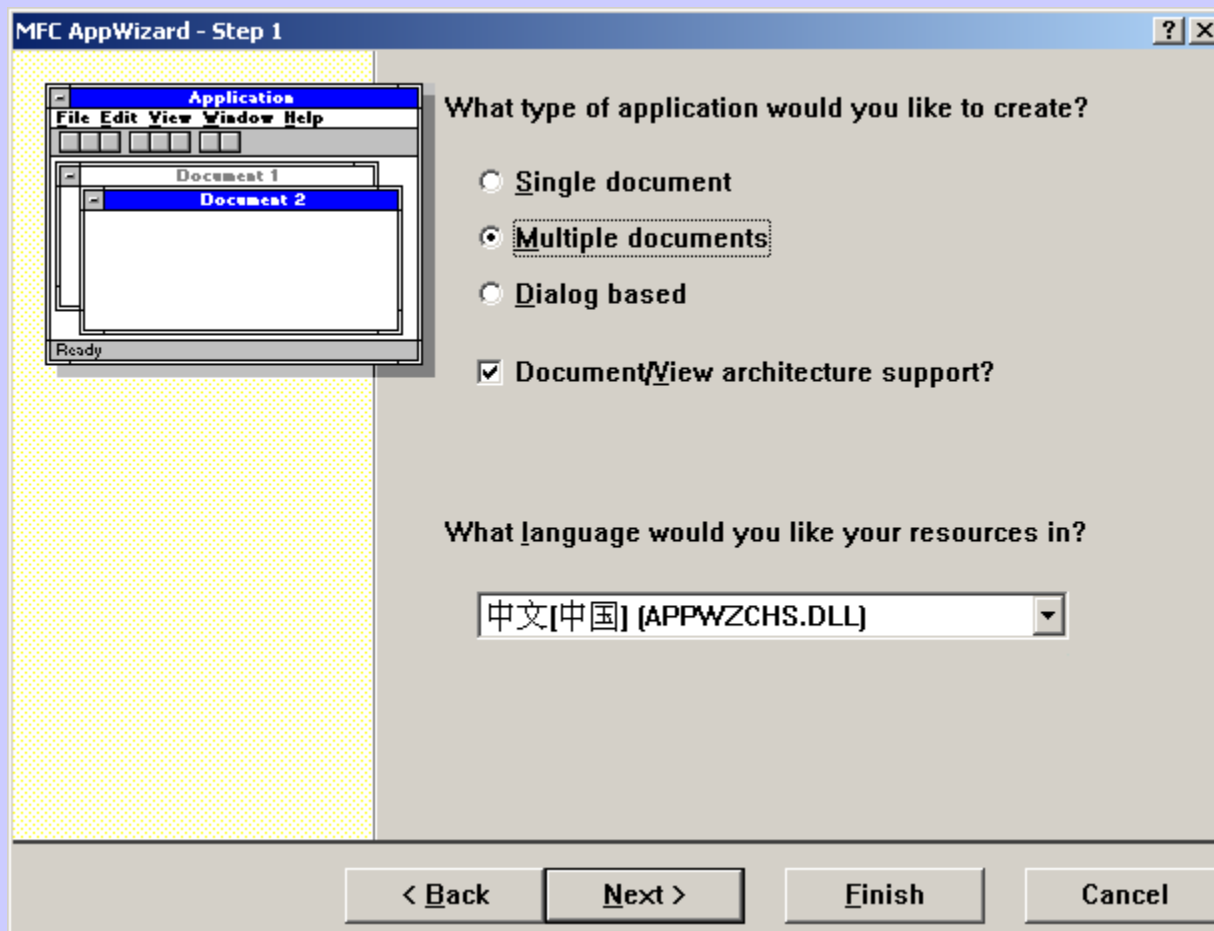


图10 MFC AppWizard-Step 1

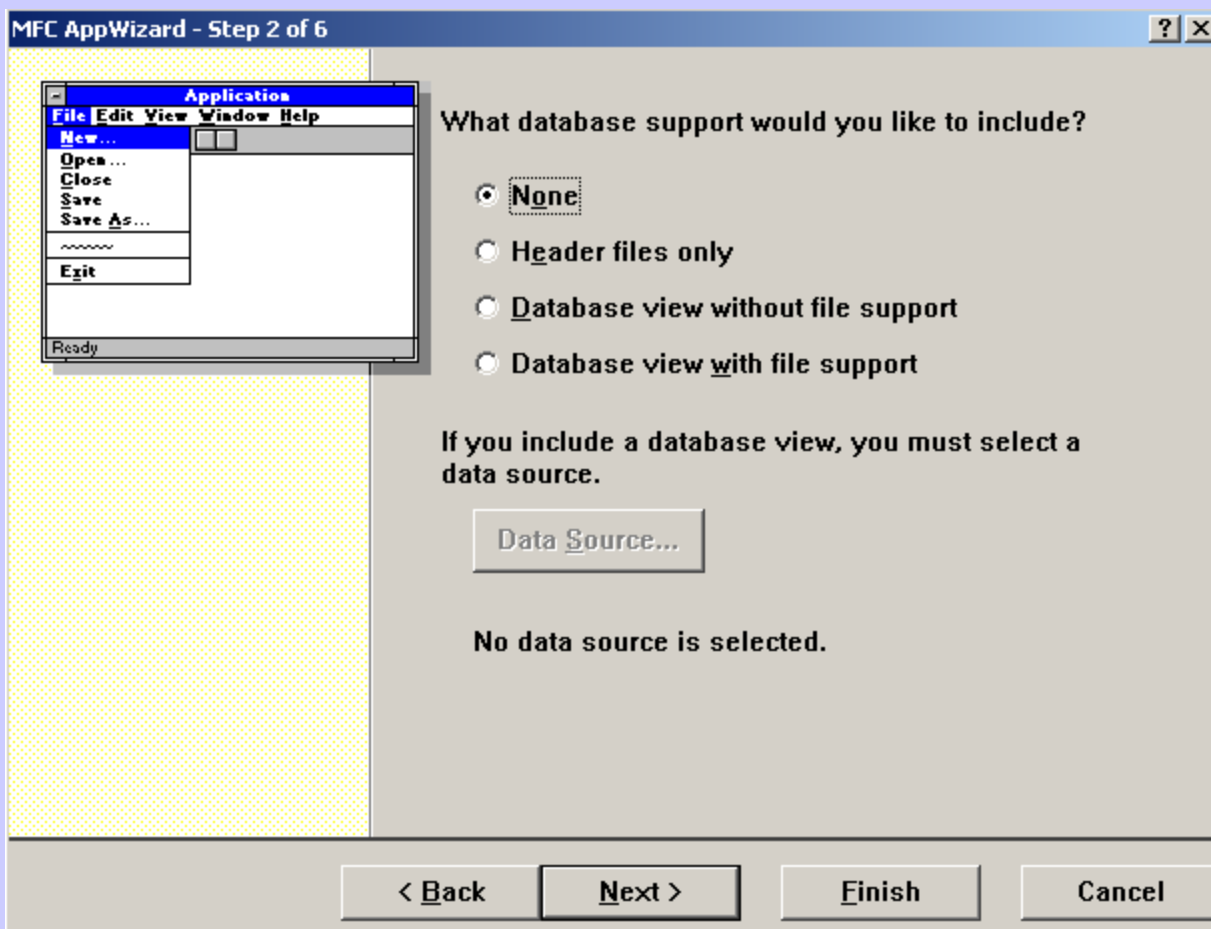


图11 MFC AppWizard-Step 2
这一步选择数据库支持，当前示例程序不需要。

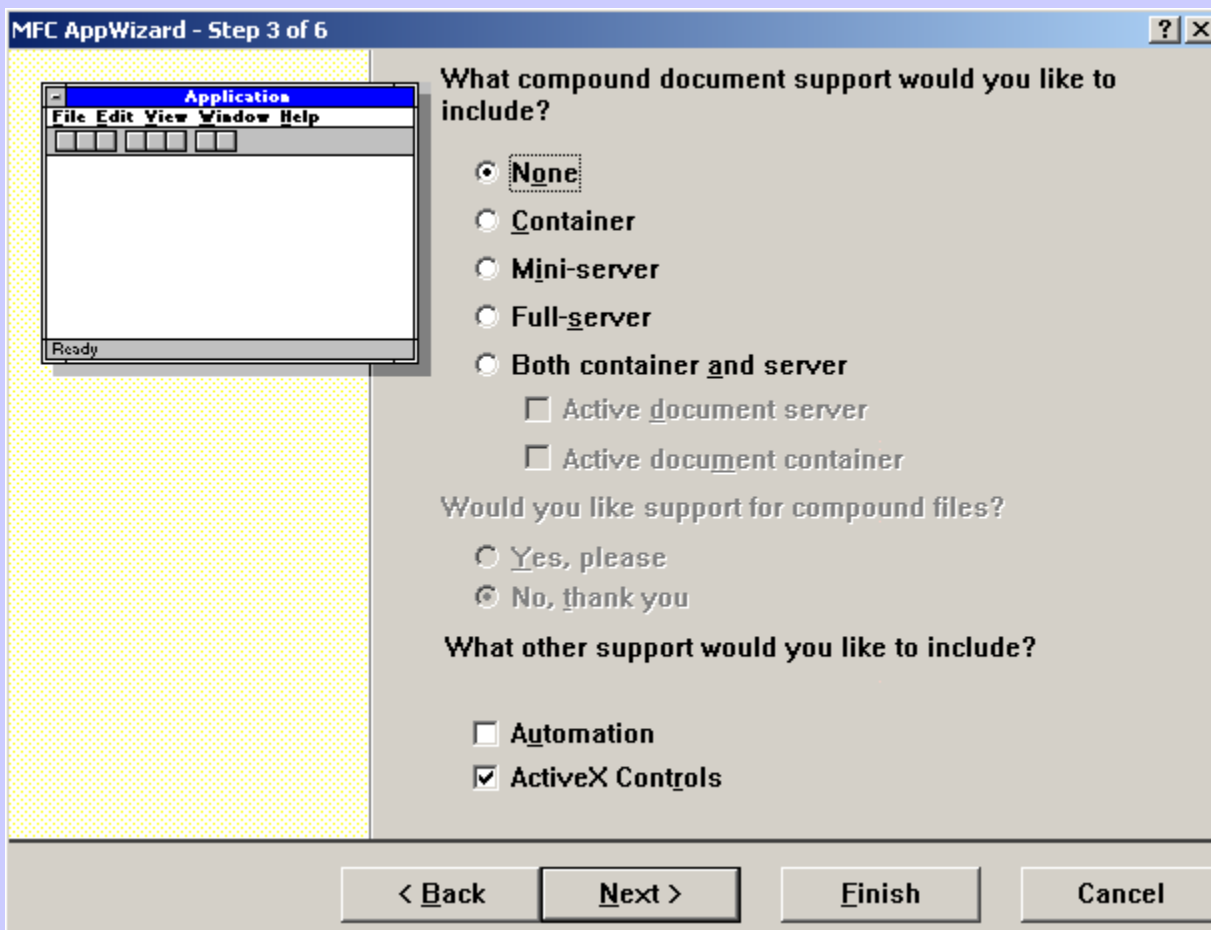


图12 MFC AppWizard-Step 3

这一步选择混合文档和ActiveX支持，当前示例程序只需要支持纯文本文档。为求简化，ActiveX Controls支持也可以不要。

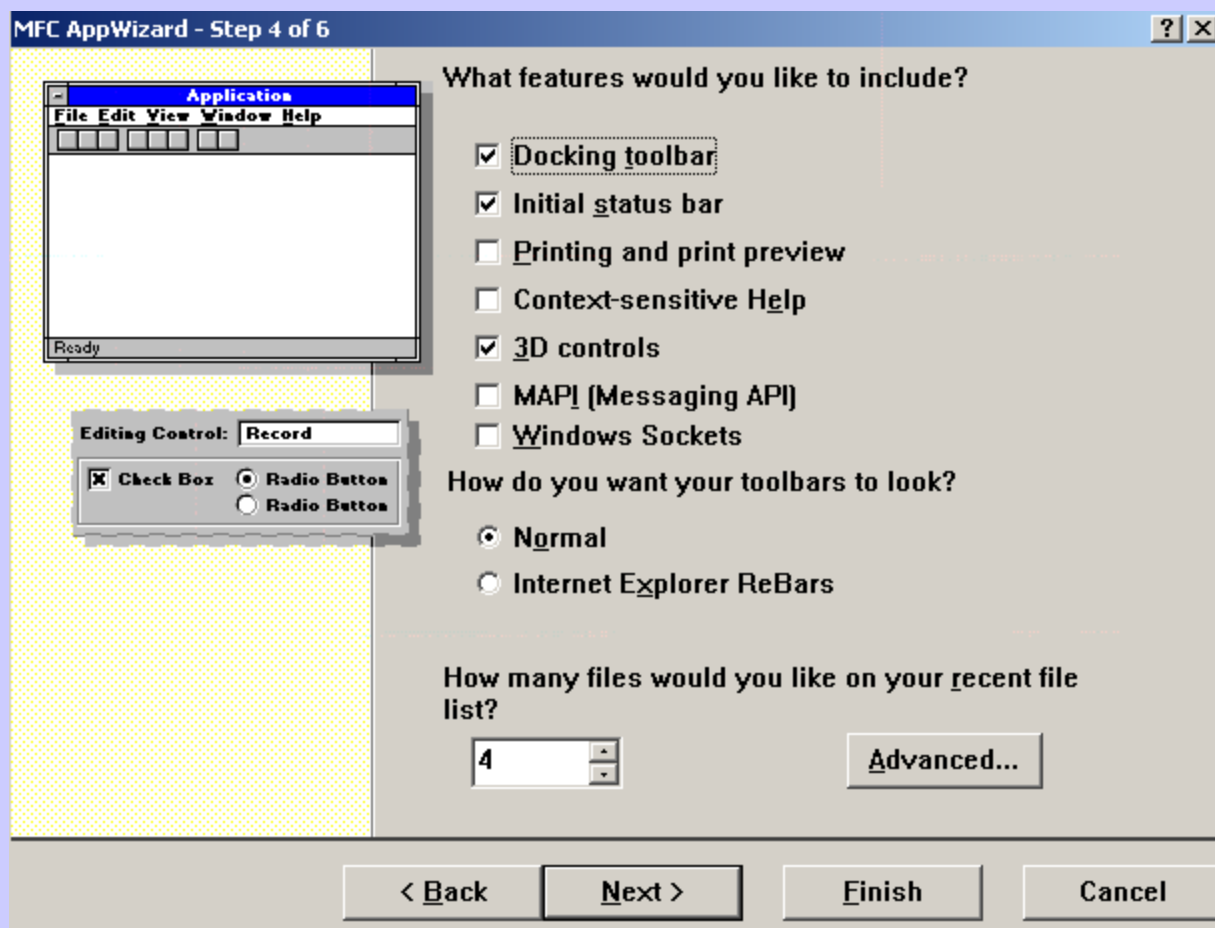


图13 MFC AppWizard-Step 4

这一步选择用户界面特性，例如标准工具栏、状态条、打印以及预览、上下文敏感帮助、3D风格等。第二部分选择工具栏风格。第三部分是选择文件菜单中的最近使用文件列表中显示的文件数量。

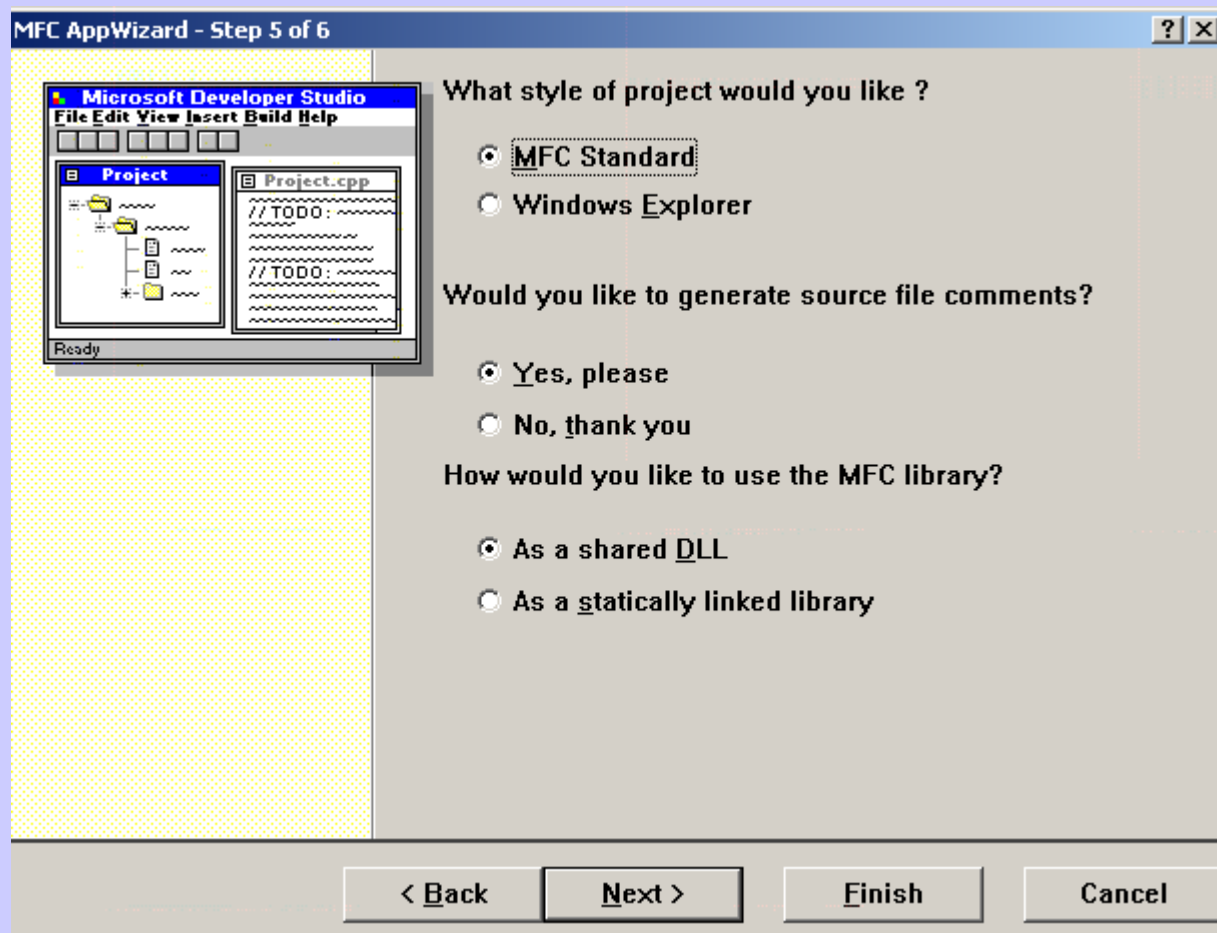


图14 MFC AppWizard-Step 5

这一步选择程序以及代码风格。第一项是程序风格，MFC标准风格和Windows浏览器风格。第二项是询问是否需要为程序代码生成一些说明文字。第三项是选择希望使用的MFC版本（动态连接版或者静态）。

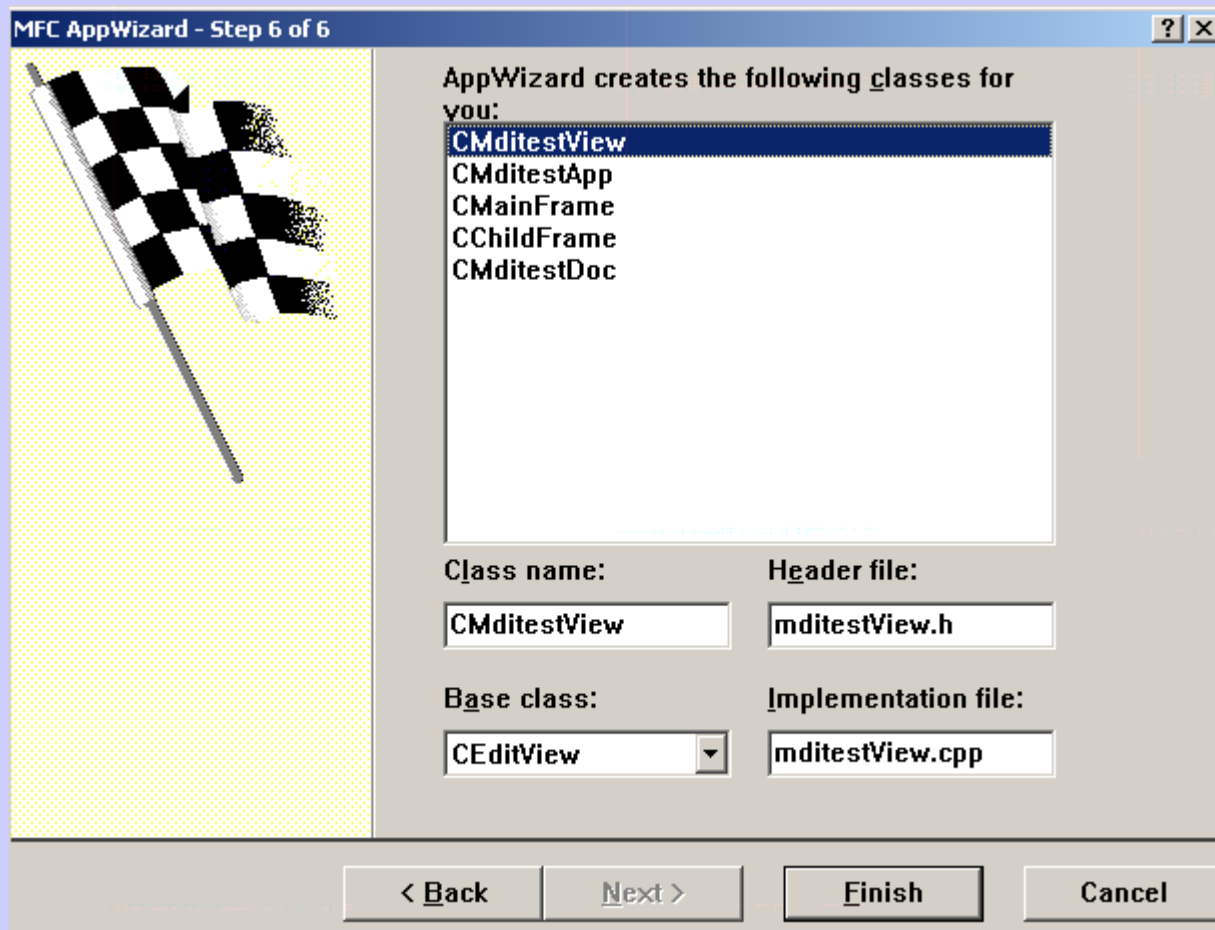


图15 MFC AppWizard-Step 6

这是最后一步，允许用户在这里更改各文件名和类名。在上面的类列表中选中一个类，下面显示的就是相应的类名、头文件将名、基类和实现文件名。这里为了使文档可编辑，将CMdittestView的基类改为CEditView。完成后点击Finish完成向导。

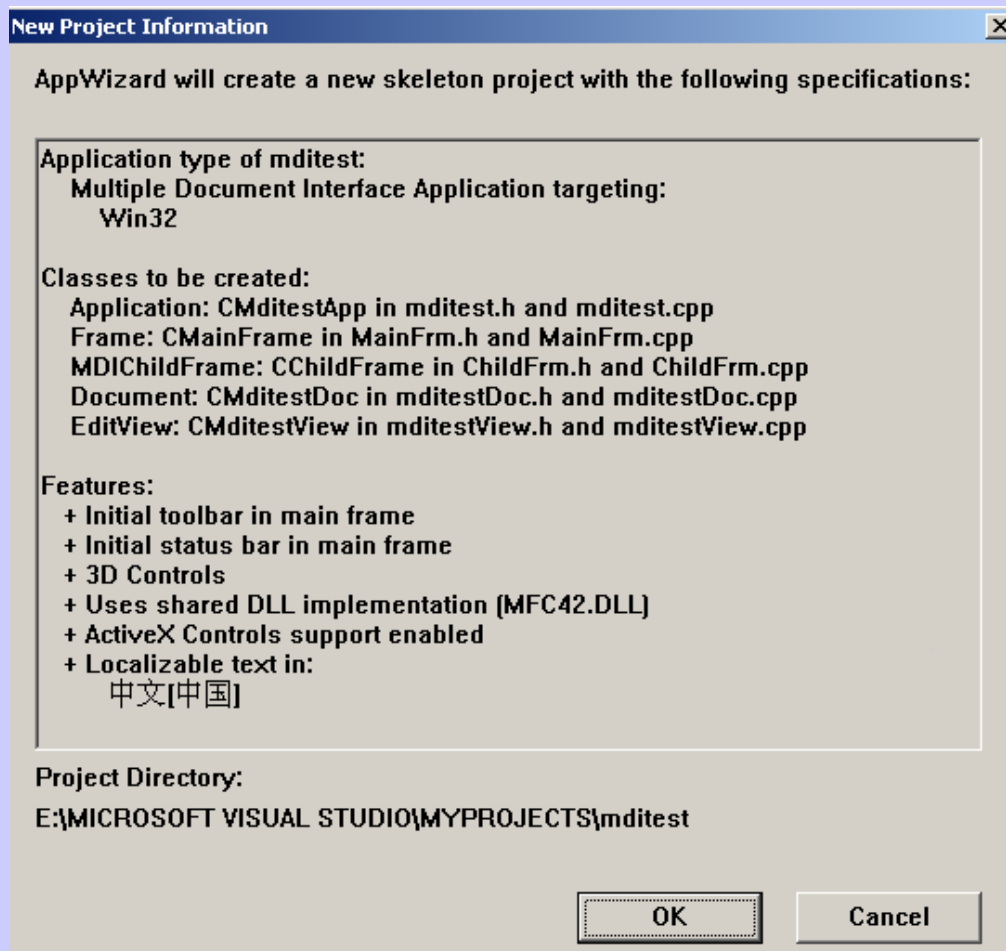


图16 项目信息

显示用户在前面各步所作选择，点击OK完成，否则Cancel重新选择。

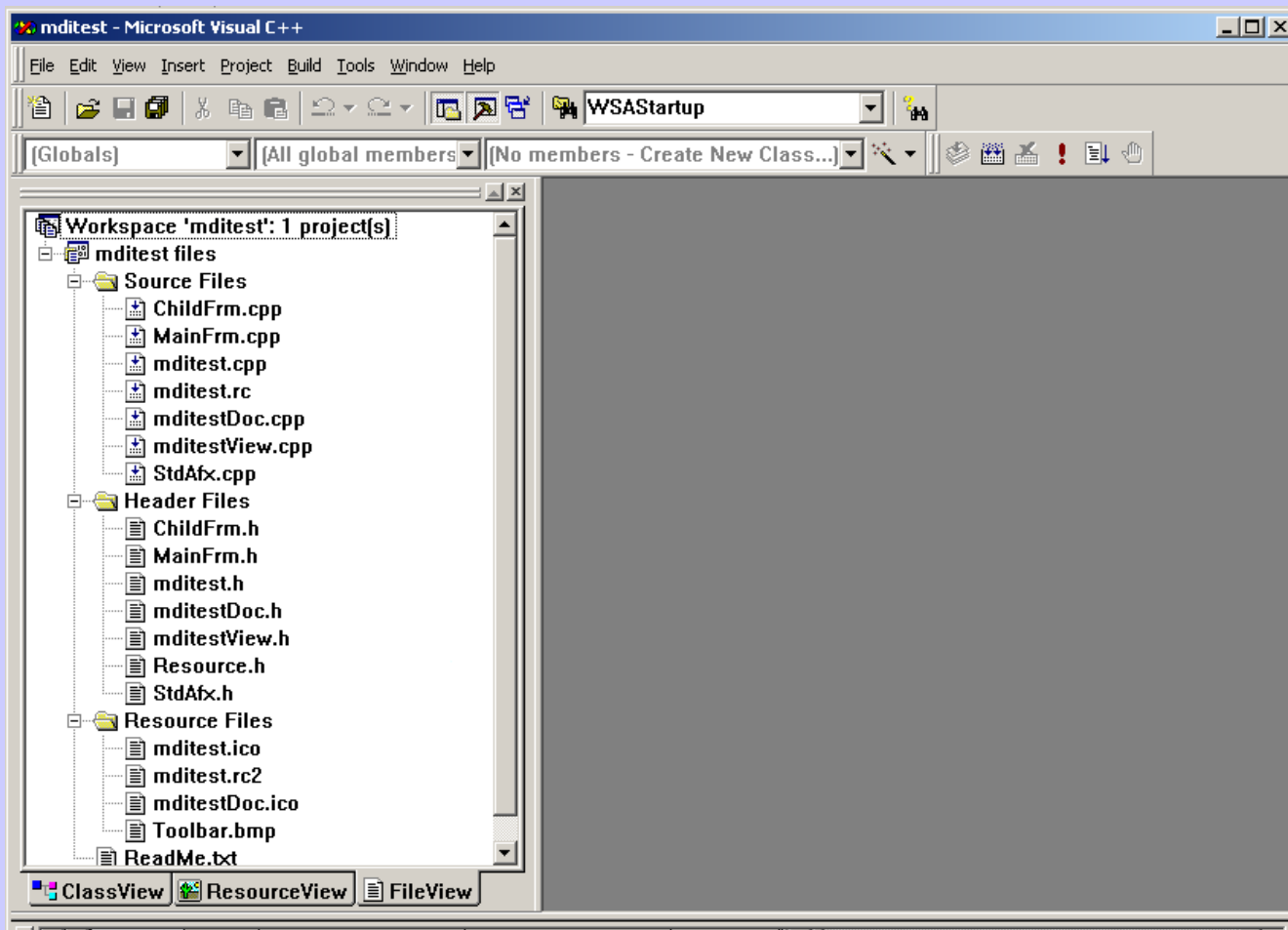


图17 项目文件列表

根据用户选择，VC已经生成了所需的全部文件。这个程序已经可以运行，运行结构就是图2所示的效果。程序已经具有标准的打开、编辑、保存以及打印等功能，而我们没有添加一行代码。

4. 自己添加代码的sayhello程序

- 这部分我们演示一个对话框程序，添加一些标准控件，并添加代码控制程序的行为。
- 新建project，名称为sayhello，利用AppWizard，只需要在指定程序类型为Dialog-Based之后，就可以点击Finish，完成程序框架。
- 程序界面如右上图。
- 点击提示按钮，弹出提示框显示文本框中字符串如右下图；点击添加按钮，文本框中字符串添加到列表中（如果不重复的话）；点击取消关闭整个窗口。

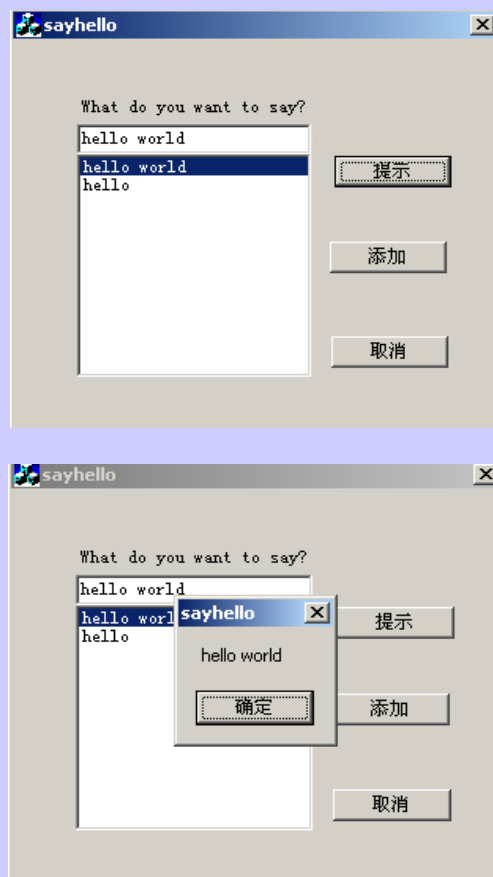


图18

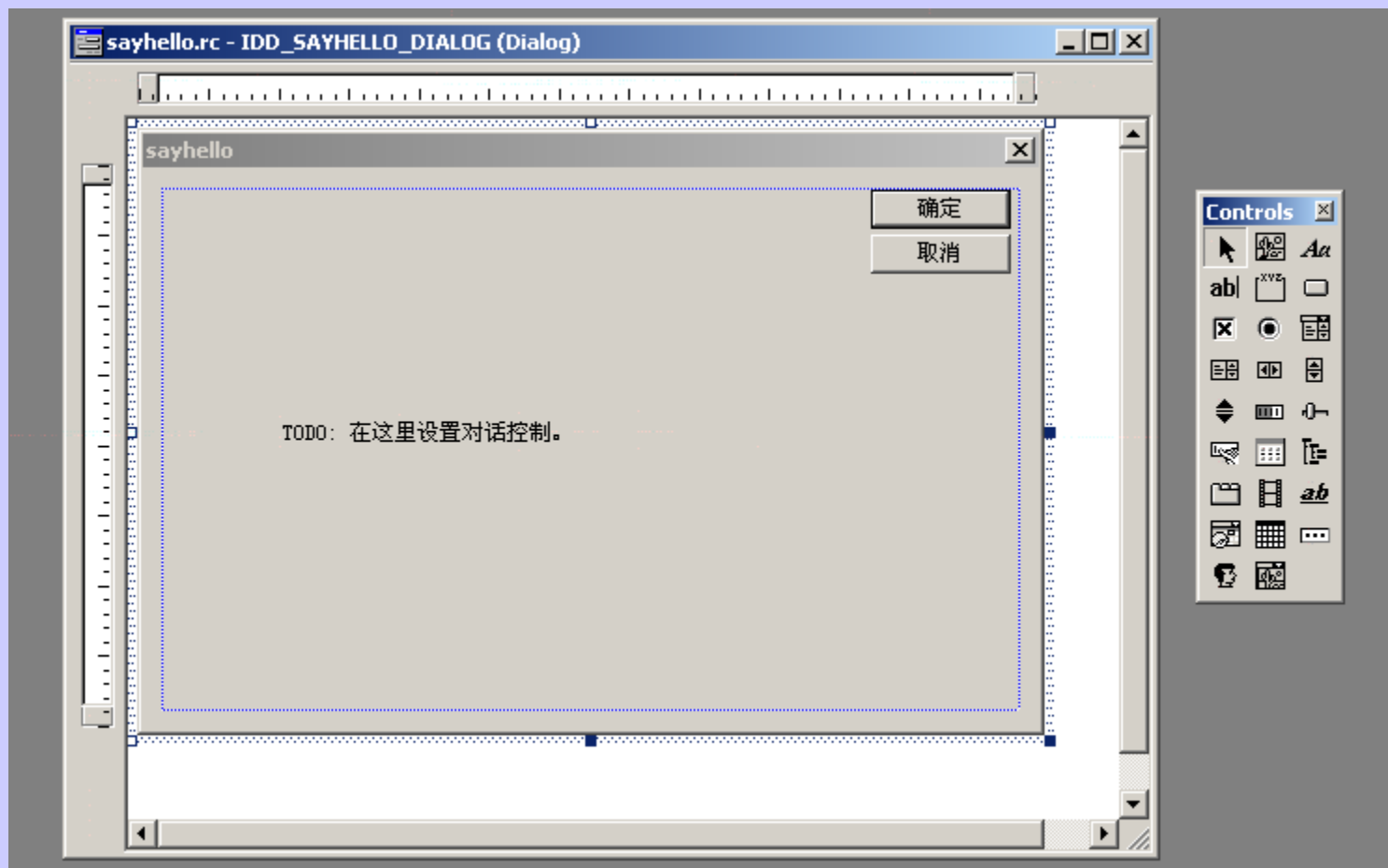


图19 框架创建后的编辑区

根据用户选择，VC已经为我们生成了所需的程序框架。这里要编辑的主要是sayhello主窗口，已经生成的有一个静态文本，两个按钮。右边的控件窗口显示一些标准窗口控件，可以通过拖放操作添加到窗口。

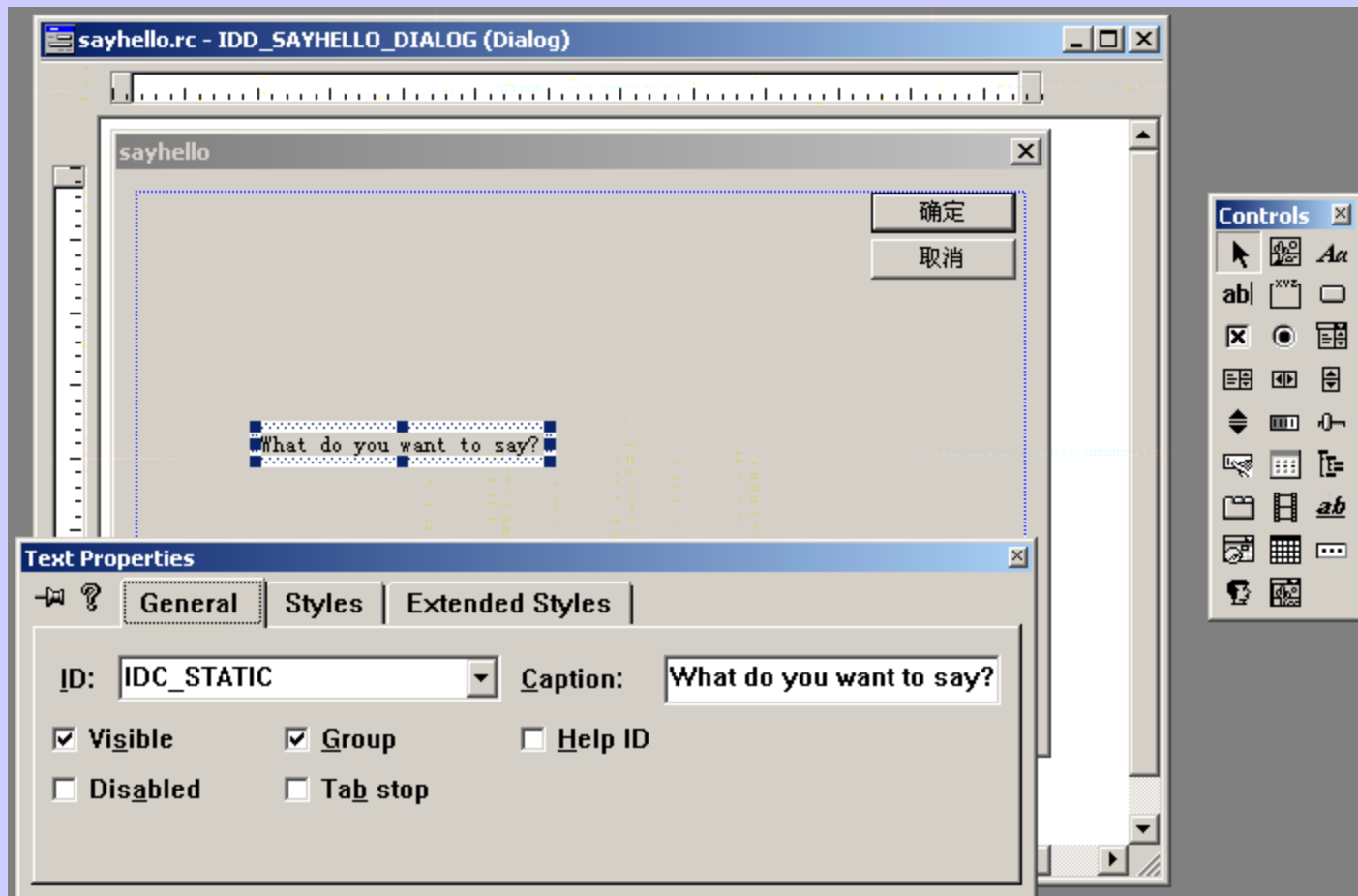


图20编辑控件属性

右键点击某控件，选择Properties，弹出该控件的属性设置窗口，如图。可以设置控件标题(caption)、可见性、对齐等属性，这些属性、风格设置分布在三个页（General、Styles、Extended Styles）上。这里我们主要需要设定的是Caption，设置为：What do you want to say？

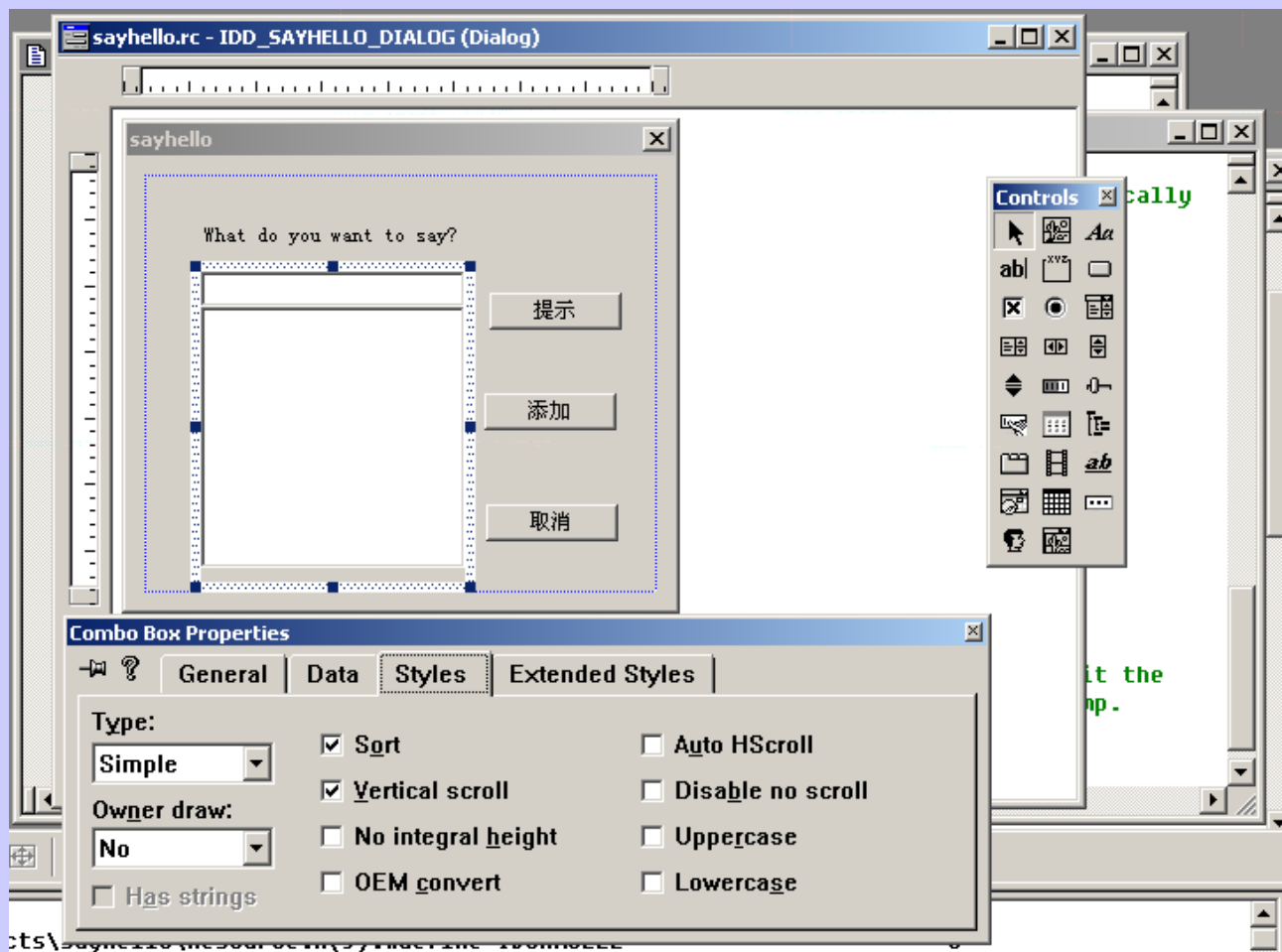


图21 完成界面设置

保留原来的取消按钮，删除确定按钮，添加两个按钮和一个 Combo Box控件，属性中Styles页设置情况如图。通过拖放、对齐操作后完成界面构造如图。其中几个控件的对齐可以通过选中多个控件后右键菜单中相关的Align选项实现。

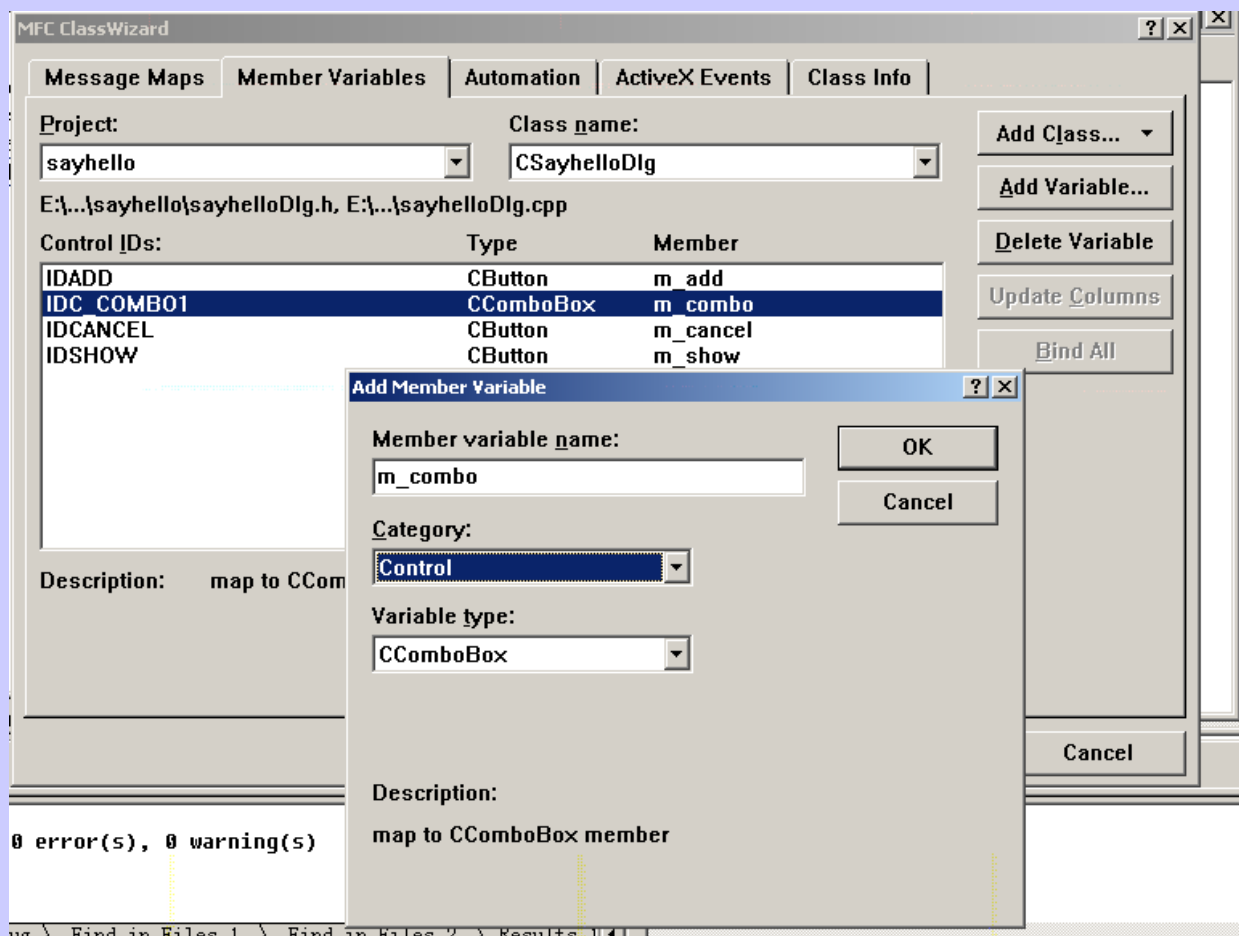


图22 设置成员变量名

点中某一控件，右键菜单中选择ClassWizard进行类设置。在Member Variables（成员变量页）选择CSayhelloDlg（代表主窗口）类，设置三个按钮和一个ComBox的变量名如图（通过双击相应条目编辑）。注意这里Category选择Control，变量类型也正确指定。这样在代码中就可以通过这些名字引用相应控件对象。

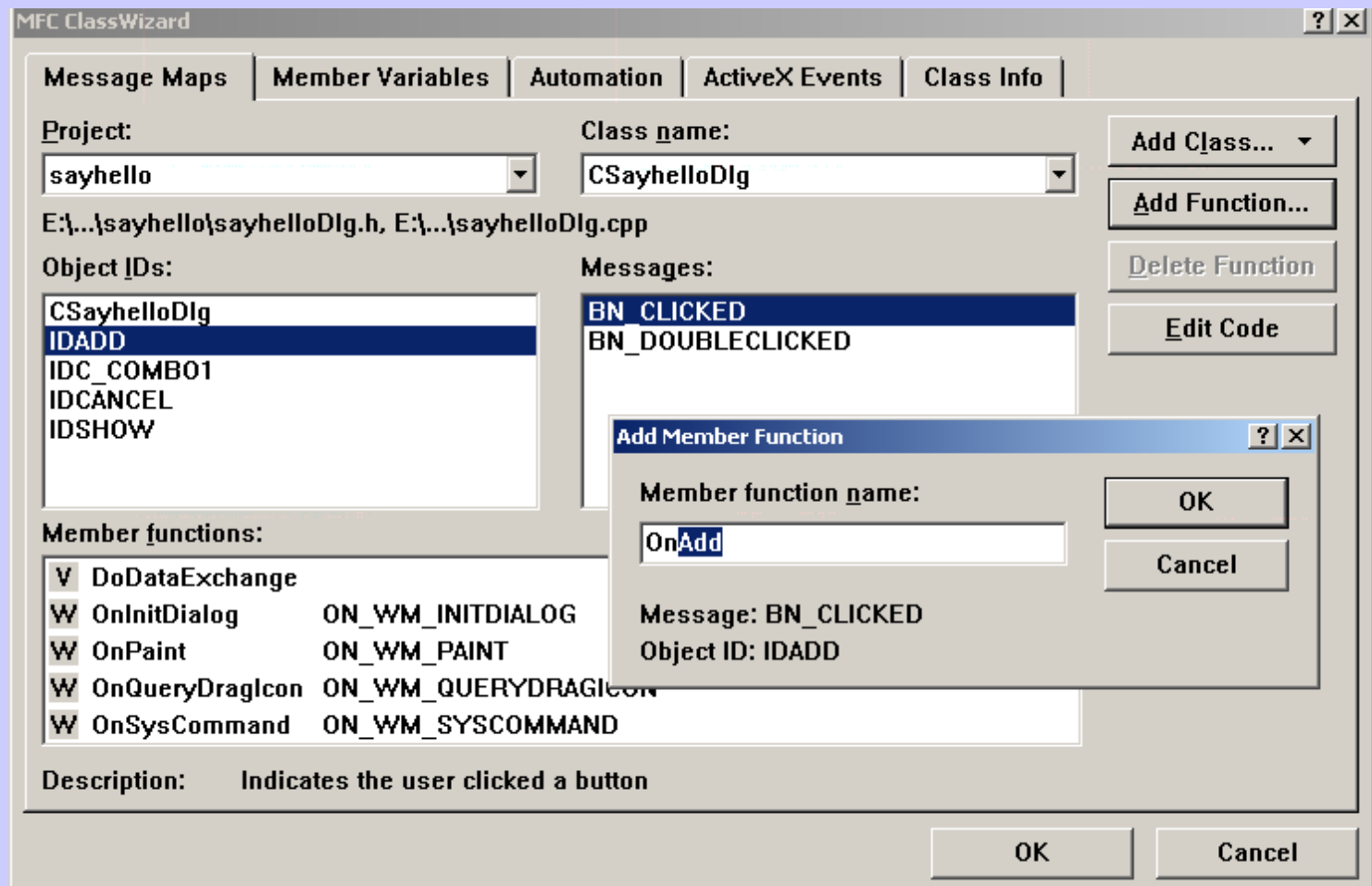
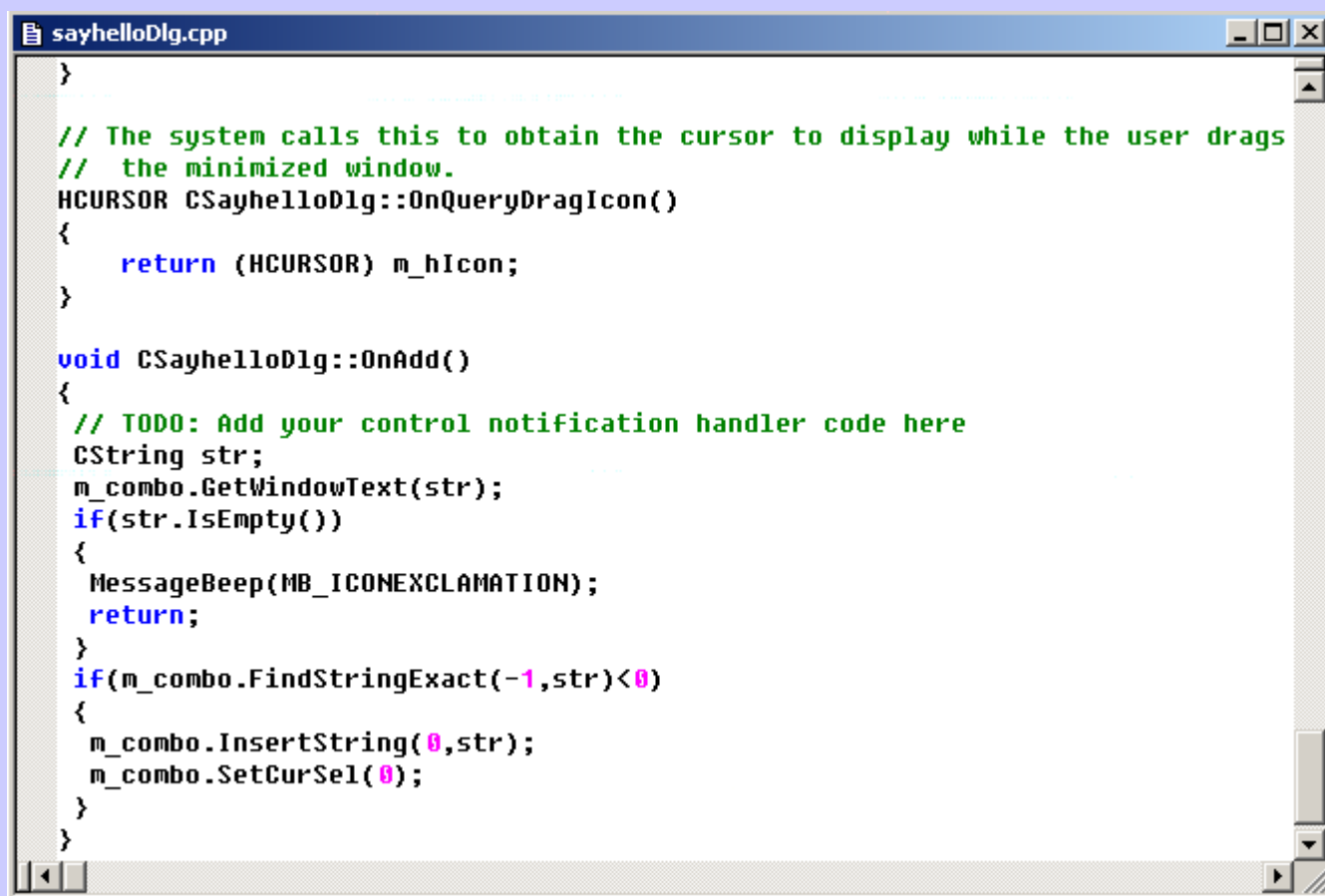


图23添加消息映射方法

仍然是在ClassWizard中，这里添加相应对象的消息处理方法。Class name中指定要处理的类，这里是主窗口类CSayhelloDlg，选择IDADD对象（“添加”按钮）和BN_CLICKED消息（单击），然后点击Add Function，指定方法名后，相应的消息处理方法框架就生成了。



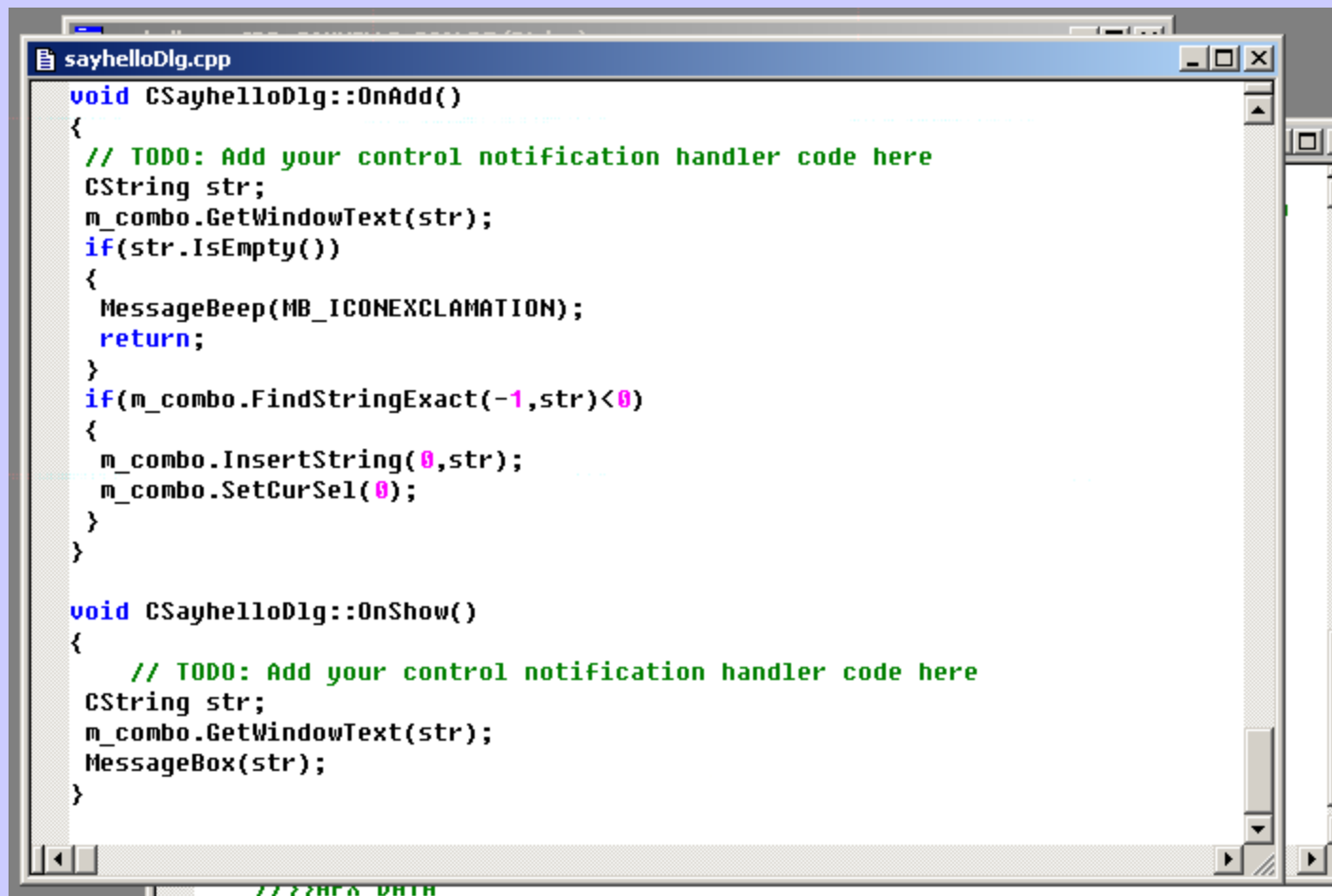
```
> sayhelloDlg.cpp
}

// The system calls this to obtain the cursor to display while the user drags
// the minimized window.
HCURSOR CSayhelloDlg::OnQueryDragIcon()
{
    return (HCURSOR) m_hIcon;
}

void CSayhelloDlg::OnAdd()
{
    // TODO: Add your control notification handler code here
    CString str;
    m_combo.GetWindowText(str);
    if(str.IsEmpty())
    {
        MessageBeep(MB_ICONEXCLAMATION);
        return;
    }
    if(m_combo.FindStringExact(-1,str)<0)
    {
        m_combo.InsertString(0,str);
        m_combo.SetCurSel(0);
    }
}
```

图24添加消息映射方法

在上一步中点击Edit Code，编辑“添加”按钮单击消息的处理方法代码。添加的代码如图，主要过程是获取ComBox输入框字符串，判断是否为空，以及ComBox中是否已存在此字符串，如果不存在则添加并将光标定位于第一项。代码中的TODO注释是生成框架时自动生成的。



```
void CSayhelloDlg::OnAdd()
{
    // TODO: Add your control notification handler code here
    CString str;
    m_combo.GetWindowText(str);
    if(str.IsEmpty())
    {
        MessageBeep(MB_ICONEXCLAMATION);
        return;
    }
    if(m_combo.FindStringExact(-1,str)<0)
    {
        m_combo.InsertString(0,str);
        m_combo.SetCurSel(0);
    }
}

void CSayhelloDlg::OnShow()
{
    // TODO: Add your control notification handler code here
    CString str;
    m_combo.GetWindowText(str);
    MessageBox(str);
}
```

图25添加“提示”按钮单击方法

与“添加”按钮类似的添加“提示”按钮的单击消息处理方法OnShow，代码如图。完成的操作是取得ComBox文本输入内容并跳出一个提示窗口显示该字符串。

5. 结束语

- 如第4部分这样一步步操作就完成了了一个窗口程序，编译链接后就可以得到可执行程序。
- 更加复杂的窗口程序可以用相似的过程得到，使用的应用程序框架也与此类似。只是可能会用到更多更复杂的MFC类，需要用户根据实际需要继承或引用各种MFC类，处理这些类之间的关系。还有可能需要在派生类中改写虚拟函数或者添加新的成员函数。
- 总之在Application Framework框架下，你可以以MFC类库为材料来源创造出符合各种需要的应用程序。
- 各种相关类信息可以通过MFC类库手册以及MSDN文档得到。

使用方法结束