转自（http://blog.lehu.shu.edu.cn/Article.aspx?name=byman&aid=333883&g=51f0642a-ea49-45a9-855e-dd0fe49d8195）

一．IDE基础配置

1. 字体

VC6中“Tools→Options→Format→Font”配置字体；VC2005中“工具→选项→环境→字体和颜色”配置字体。

编写代码一般采用等宽字体，等宽点阵（位图）字体，相对矢量字体而言具有兼容性好和显示清晰的优点。常用的编程等宽字体包括Fixedsys（VC6和notepad御用字体）、Consolas、YaHei Mono。

VC6默认可选字体只有Fixedsys和Terminal，可通过修改注册表来改字体，在HKEY\_CURRENT\_USER/Software/Microsoft/DevStudio/6.0/Format下面有几个子键，可以修改Source Window键的FontFace和FontSize来配置源代码窗口的显示字体。例如设置11号的Consolas字体：FontFace=Consolas，FontSize=11。

2. 制表符（TAB）

程序块要采用缩进风格编写，IDE碰到左括号换行的情况一般会自动缩进（Auto Indent），缩进的空格数和制表符的宽度一般设为4个半角字符。由于不同的编辑器对TAB空格数的设置可能不一样，有的是4有的是8，为避免使用不同的编辑器阅读程序时的布局不整齐问题，建议只使用空格键对齐。

VC6中“Tools→Options→Tabs”的“Insert spaces/Keep tabs”中选择“Insert spaces”；VC2005中“工具→选项→文本编辑器→所有语言→制表符”可配置制表符大小（Tab Size）/缩进大小（Indent Size）为4，对“插入空格/保留制表符”勾选“插入空格”。

3. 行代码字符数

关于一行编写多少代码换行比较合适的问题，可以在VC中输入一行”//”即可得知，一般为75或80。

4．外接插件和外部工具

外接插件能扩展Visual Studio的功能，例如VC6默认并无行号显示功能，可以使用VC6LineNumberAddin插件实现显示行号，“Tools -> Customize→Add-ins and Macro Files”选择“Browse”进VC6LineNumberAddin.dll即可加载使用。我们在VC2005的“工具→外接程序管理器”中可以看到安装的“Visual Assist X”插件，并可选择启用与否。

外部工具主要是结合IDE使用，如使用AStyle格式化当前打开的代码文件，使用gedit打开当前代码文件。外部工具的配置很简单，VC6中通过“Tools→Customize→Tools”添加，VC2005中通过“工具→外部工具”添加。

以下为AStyle的配置：

标题：AStyle

命令：C:/Program Files/Microsoft Visual Studio 8/Common7/Tools/AStyle.exe

参数：--style=ansi $(ItemFileName)$(ItemExt)

初始目录：$(ItemDir)

勾选“使用输出窗口”。

这样，“工具菜单”中新增一个“AStyle”的子项，点击“AStyle”即可按ANSI风格格式化当前文件代码。

5. IDE指定和打开多个项目

Q1：安装VS2005后，之前的VC6默认打开程序和图标都变更为VC2005的风格。如何恢复至VC6的默认风格呢？

A1：工具→文件夹选项→文件类型（Win7中是设置默认程序）

找到DSW文件，高级（可能先还原）→编辑，用于执行操作的应用程序由"C:/Program Files/Microsoft Visual Studio 8/Common7/IDE/devenv.exe" "%1"更改为"C:/Program Files/Microsoft Visual Studio/Common/MSDev98/Bin/MSDEV.EXE" "%1"，这样默认\*.dsw文件就以VC6.0打开，而不是以VC8.0打开。同样更改\*.dsp文件类型以用VC6打开，当然可以更改显示图标为VC6风格的。

Q2：双击VC项目文件（\*.sln/\*.dsw），并没有打开一个新的IDE窗口，而是替代了已经打开的项目。

A2：工具→文件夹选项→文件类型（Win7中是设置默认程序）

找到DSW/SLN文件，高级（可能先还原）→编辑，勾掉“使用DDE”。

二．预处理宏

1. 通用宏

VC6.0中：

Project Setting→C/C++→Preprocessor definitions

VC2005中：

项目属性→配置属性→C/C++→预处理器定义

（1）平台识别宏

WIN32：指明是Window 32位系统平台。

常见的平台识别宏如下：

\_DOS\_：MS-DOS 16位系统平台。

WIN64：指明是Window 64位系统平台。

\_WIN32\_WCE：Window Mobile 32位系统平台。

\_UNIX：UNIX系统平台。

\_POSIX\_：POSIX（Portable Operating System Interface of Unix）系统平台。

\_LINUX\_：LINUX系统平台。

\_APPLE\_/\_MAC\_：苹果（Macintosh）系统平台。

（2）字符集识别宏

\_MBCS/\_UNICODE：指明该工程使用的字符集（多字节字符集/UNICODE）。

在VC6.0的“Project Setting→C/C++→Project Options”处将会有/D "\_MBCS"或/D "\_UNICODE"开关。

在VC2005的“项目属性→配置属性→常规→项目默认值→字符集”中选择“使用多字节字符集”或“使用 Unicode 字符集”；在“项目属性→配置属性→C/C++→使用UNICODE响应文件”中选择“是”或“否”。在“项目属性→配置属性→C/C++→命令行” 处将会有/D "\_MBCS"或/D "\_UNICODE" /D "UNICODE"开关。

当项目中的文件具有 UNICODE 路径（例如中文路径）时，应在“项目属性→C/C++→常规”和“项目属性→链接器→常规”中开启“使用UNICODE响应文件”开关。

（3）UI识别宏

\_CONSOLE/\_WINDOWS：指明应用程序的类型是控制台程序或窗口程序（CUI/GUI）。

（4）debug/release版本识别宏

\_DEBUG/NDEBUG：指明编译版本（Debug/Release）。

2.项目类型特定宏

VC不同的项目类型，有一些特定的宏，以在编译器区分项目类型。

（1）Win32 Static Library项目

VC6.0中：

Project Setting→C/C++（Category：General）→Preprocessor definitions：\_LIB

如果选择了MFC Support则\_WINDOWS,\_AFXDLL

VC2005中：

项目属性→常规→配置类型→静态库(.lib)

项目属性→C/C++→预处理器→预处理器定义：\_LIB

（2）Win32 Dynamic-Link Library项目

VC6.0中：

Project Setting→C/C++→Preprocessor definitions：\_WINDOWS,\_USRDLL,DLL\_EXPORTS

VC2005中：

项目属性→常规→配置类型→动态库(.dll)

项目属性→C/C++→预处理器→预处理器定义：\_WINDOWS;\_USRDLL;DLL\_EXPORTS

（3）MFC DLL项目

<1>Regular DLL with MFC statically linked（带静态链接MFC的规则DLL）

VC6.0中：

Project Setting→C/C++→Preprocessor definitions：\_WINDOWS,\_WINDLL,\_USRDLL

VC2005中：

项目属性→C/C++→预处理器→预处理器定义：\_WINDOWS;\_USRDLL

<2>Regular DLL using shared MFC DLL（使用共享MFC DLL的规则DLL）

VC6.0中：

Project Setting→C/C++→Preprocessor definitions：\_WINDOWS,\_WINDLL,\_AFXDLL, \_USRDLL

VC2005中：

项目属性→C/C++→预处理器→预处理器定义：\_WINDOWS;\_USRDLL

<3>MFC Extension DLL（using shared MFC DLL）（MFC扩展DLL）

VC6.0中：

Project Setting→C/C++→Preprocessor definitions：\_WINDOWS,\_WINDLL,\_AFXDLL,\_USRDLL

VC2005中：

项目属性→C/C++→预处理器→预处理器定义：\_WINDOWS; \_AFXEXT

在DLL项目中，\_USRDLL宏表示生成Export Lib！

3.其他常用宏

\_\_FILE\_\_/\_\_LINE\_\_：

\_\_FILE\_\_为包含当前程序文件名的路径字符串，例如C:/Program Files/Microsoft Visual Studio/VC98/INCLUDE/stdio.h；\_\_LINE\_\_为当前程序代码行号，从1开始的整数。

\_\_STDC\_\_/\_\_cplusplus：

当编译标准C （ANSI C）程序时，编译器自动定义\_\_STDC\_\_ ；当编译标准C++ 程序时,编译器自动定义\_\_cplusplus。

\_MSC\_VER：Microsoft Visual C++编译器的版本号。例如VC++6.0为1200，VC++8.0为1400，以此类推。

WIN32\_LEAN\_AND\_MEAN告诉编译器不加载MFC（Not Using MFC）。

关于VC编译器预处理宏，参考：

《AFXDLL 和 \_AFXEXT》

《Predefined Macros（VC6.0）》

《C/C++ Preprocessor Reference-Predefined Macros（VC8.0）》

《What’s the difference between the WIN32 and \_WIN32 defines in C++》

三．工作目录和控制台程序运行参数

VC6.0中：

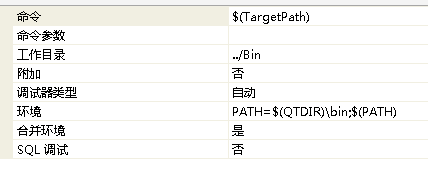
“Project Setting→Debug→Working directory”处填写exe程序的工作目录。

在“Project Setting→Debug→Program argument”处填写main运行参数。

VC2005中：

在“项目属性→配置属性→调试→工作目录”处填写exe程序的工作目录。

注意：项目文件（类似Panosee.vcproj类型）所在目录为本层目录，若工作目录为往上一层则在前面加一个“../”例如



其中Bin文件夹在项目文件夹的上一次目录下

在“项目属性→配置属性→调试→命令参数”处填写main运行参数。

默认的工作目录为ProjectDir，即\*.dsp/\*.vcproj所在目录。

通过编译器启动（Ctrl + F5）应用程序时，可以将所依赖的lib和dll库文件所在目录设为工作目录。应用程序执行期间，文件操作的默认路径为工作目录。

工作目录是指当前目录，GetCurrentDirectory()得到的路径；SetCurrentDirectory来改变。

运行目录是指exe所在路径，可通过GetModuleFileName()得到。

四．附加头文件包含

VC6.0中：

VC6.0默认include包含路径：Tools→Options→Directories→Include files。

对于特定项目的头文件包含，在“Project Setting→C/C++（Category：Preprocessor）→Additional include directories”处添加附加头文件目录，相对目录参考\*.dsp文件。

目录路径不需添加双引号，路径分隔符为’/’或’/’，不必考虑反斜杠’/’的转义。多个路径之间以逗号分隔。

相应的在“Project Setting→C/C++→Project Options”中将会出现“/I”的包含命令。

VC2005中：

VC2005默认include包含路径：工具→选项→项目和解决方案→VC++目录→包含文件。

对于特定项目的头文件包含，在“项目属性→配置属性→C/C++→常规→附加包含目录”处添加附加包含目录，相对目录参考$ProjectPath（\*.vcproj文件路径）。

..\..\inc;.\GeneratedFiles;.;"$(QTDIR)\include";".\GeneratedFiles\$(ConfigurationName)";"$(QTDIR)\include\QtCore";"$(QTDIR)\include\QtGui"

其中..\..\inc为公共头文件路径，在项目文件夹上两层，

目录路径不需添加双引号，路径分隔符为’/’或’/’，不必考虑反斜杠’/’的转义。多个路径之间以分号分隔。

相应的在“项目属性→配置属性→C/C++→命令行” 中将会出现“/I”的包含命令。

关于#include的两种包含方式

#include <std\*.h>为编译器指定的包含路径下的头文件：即在VC6.0中的“Tools→Options→Directories→Include files”（对应VC2005的“工具→选项→项目和解决方案→VC++目录→包含文件”）处指定的包含目录。默认情况下包括

（1）$(VCInstallDir)include（/Microsoft Visual Studio/VC98/INCLUDE）

（2）$(VCInstallDir)atlmfc/include（/Microsoft Visual Studio/VC98/ATL/INCLUDE、/Microsoft Visual Studio/VC98/MFC/INCLUDE）

（3）$(VCInstallDir)PlatformSDK/include（VC2005）。

对于“Tools→Options→Directories→Include files”（“工具→选项→项目和解决方案→VC++目录→包含文件”）目录下的文件的包含，一般可使用尖括号包含。系统对尖括号的包含优先到编译器设置的包含路径下定位，再到附加包含目录下进行定位。

#include "MyHeader.h"为在“Tools→Options→Directories→Include files”（“工具→选项→项目和解决方案→VC++目录→包含文件”）处人工添加的包含路径下的头文件或者附加包含目录下的头文件。

对于编译器提供的标准库函数头文件（“Tools→Options→Directories→Include files”（“工具→选项→项目和解决方案→VC++目录→包含文件”）），一般使用尖括号进行包含声明，当然使用双引号包含也没有错。

对于在“Tools→Options→Directories→Include files”（“工具→选项→项目和解决方案→VC++目录→包含文件”）处人工添加的包含路径和附加包含目录下的头文件的包含可以使用尖括号进行包含，但一般使用双引号。

原则上，对于工程或文件可见的包含目录下的头文件包含，使用以上两种包含方式中的任一种都不会出错，主要是它们在搜索目录时的顺序不一样。习惯上使用尖括号包含标准库函数头文件，使用双引号包含用户自定义头文件。

一般针对具体工程使用附加头文件包含即可，不必去动编译器的“Tools→Options→Directories→Include files”（对应VC2005的“工具→选项→项目和解决方案→VC++目录→包含文件”）。但是，对于像DirectX、wxWidgets等这样的SDK，一般宜对include/library路径作编译器级全局配置，同时指定相应source路径，以便做源码级调试（F11进入SDK内部代码）。下面的链接库包含情同此理。

External Dependencies里面显示的是你的工程要用到的头文件，不在你的工程里面，而且也不在VC的默认包含路径中。参考：《VC6.0的External Dependencies》。

五．附加链接库包含

VC6.0中：

VC6.0默认lib包含路径：Tools→Options→Directories→Library files

对于特定项目的头文件包含，可以通过以下两步添加附加链接库：

1. 在“Project Setting→Link（Category：Input）→Additional library path”处添加附加库目录，相对目录参考\*.dsp文件。

目录路径不需添加双引号，路径分隔符为’/’或’/’，不必考虑反斜杠’/’的转义。多个路径之间以逗号分隔。

相应的将会在“Project Setting→Link→Project Options”处同步“/libpath”库包含命令。

2. 在“Project Setting→Link（Category：General）→Object/Library Modules” 处添加库文件名YourLib.lib（也可在文件中使用#pragma comment(lib, "YourLib.lib")显式导入所依赖的静态库），多个库之间以空格分隔。

相应的将会在“Project Setting→Link→Project Options”处同步包含的库名称。

VC2005中：

VC2005默认lib包含路径：工具→选项→项目和解决方案→VC++目录→库文件

对于特定项目的库文件包含，可以通过以下两种方式添加附加库目录：

1.

（1）在“项目属性→配置属性→链接器→常规→附加库目录”处输入附加库目录路径，相对目录参考$ProjectPath（\*.vcproj文件路径）。

目录路径不需添加双引号，路径分隔符为’/’或’/’，不必考虑反斜杠’/’的转义。多个路径之间以分号分隔。

相应的将会在“项目属性→配置属性→链接器→命令行”处同步“/LIBPATH”库包含命令。

（2）在“项目属性→配置属性→链接器→输入→附加依赖项” 处添加库文件名YourLib.lib（也可在文件中使用#pragma comment(lib, "YourLib.lib")显式导入所依赖的静态库），多个库之间以空格分隔。

相应的将会在“项目属性→配置属性→链接器→命令行”处同步包含的库名称。

2.

在“项目属性→配置属性→链接器→命令行” 处添加库文件名YourLib.lib（必要时添加相对目录，保证YourLib的可见性）。

六．生成文件目录设置

VC6.0中：

在“Project Setting→General→Output directories”的“Output files”处填写\*.exe或\*.lib/\*.dll的生成目录，相应在“Project Setting→Link（或Library）→Output file name”处将会同步文件输出路径。相对目录参考\*.dsp文件。

在“Project Setting→General→Output directories”的“Intermediate files”处填写中间文件生成目录。相对目录参考\*.dsp文件。

VC2005中：

在“项目属性→常规→输出目录”处填写输出目录，在“项目属性→常规→中间目录”处填写中间目录，相对目录参考$ProjectPath（\*.vcproj文件路径）。

在“项目属性→链接器（或管理员）→常规→输出文件”处填写输出文件路径。

输出目录（OutDir）一般为$(SolutionDir)$(ConfigurationName)

输出文件（OutputFile）一般为$(OutDir)/$(ProjectName).exe（\*.lib，\*.dll）

1．环境变量宏说明

$(SolutionDir)为\*.sln文件所在文件夹，尾带‘/’；

$(ProjectDir)为\*.vcproj文件所在文件夹，尾带‘/’；

$(PlatformName)为平台名称，例如Win32，Windows Mobile 6 Professional SDK(ARMV4I)。

$(ConfigurationName)为\*.sln的\*.vcproj文件所在文件夹下的debug、release文件夹，参考\*.vcproj文件中的<Configurations>配置项。

具体在“项目属性→配置属性→调试→命令”下拉点击“编辑”，展开“宏(M)>>”，可以看到很多项目内置宏。

2．生成目录的配置

输出目录（OutDir）：$(ProjectDir)$(PlatformName)/$(ConfigurationName)

以上穿插$(PlatformName)主要是为了发布多平台版本考虑，对于普通的单平台开发，该路径可选。

TargetDIr为OutDir后加‘/’。

输出文件主要按项目区分如下：

（1）DLL项目：\*.pdb，\*.exp，\*.lib

（2）EXE项目：\*.pdb

中间目录（IntDir）：$(OutDir)/Intermediate

中间文件包括\*.pch，\*.obj，（\*.sbr），vc6(8)0.idb，vc6(8)0.pdb

输出文件：$(OutDir)/$(ProjectName).lib(dll、exe)

（1）DLL项目：\*.ilk，\*.dll

（2）EXE项目：\*.ilk，\*.exe

关于\*.pch，参考《预编译头文件》；注意\*.ilk和\*.pdb文件只有在Debug版本时才会生成，关于\*.ilk和\*.pdb，参考《VC项目配置基础》。

一般一个大的工程（\*.dsw，\*.sln），同时含有LIB、DLL和EXE项目，习惯将LIB输出到项目输出目录或将这些链接时用到的LIB放到专门的目录；EXE和其运行依赖的DLL都输出到BIN目录，这样便于项目的组织和最终成果的发布。

七．关于相对目录

相对目录是相对某个文件而言的，例如在VC项目属性配置附加包含目录、附加库目录则是相对\*.dsp、\*.vcproj文件而言的；在某个\*.c、\*.cpp文件中使用#include包含命令，是相对这个\*.c、\*.cpp文件而言的。

当前目录是可见的，一般直接引用文件名（夹），对于子文件夹可以添加”./”。

..：父目录

../include：父目录下的include目录

../..：祖父目录

../../include：祖父目录下的include目录

在VC2005中，“属性窗口”可查看当前文档的“文件信息”，其中可查看文件相对\*.vcproj的相对路径。

八．依赖项目联编

一个解决方案(dsw、sln)可以创建很多辅助子项目(dsp、vcproj)，如何设置使在编译主项目的时，先编译所依赖的辅助项目呢？

一般会配置辅助子项目的OutDir和主项目相同，这样生成的lib或dll对于主项目exe直接可见。

VC6.0中：

选择菜单“Project→Dependencies”可以选择需要联编的项目。

VC2005中：

在解决方案资源管理器中右击项目名称，选择“引用”或“项目依赖项”菜单可添加（选择）引用依赖项目；选择“项目生成顺序”可查看建立项目依赖后的项目生成顺序。

建立依赖关系后，主项目编译（F7，Build）时，若发现辅助依赖子项目有改动，则会先编译子项目。

九．附加DLL调试

VC6.0中：

如果你需要调试的DLL是动态装入（LoadLibrary）的，则需要在“Project Setting →Debug（Category：Additional DLLs）”处添加这个DLL文件。

Place a check in the first column to load the module’s symbols at the start of a debug session.

十．导出makefile

VC6.0中选择菜单“Project→Export Makefile”即可导出Makefile文件\*.mak。

VC6以后不再支持makefile输出了。

参考：《VC的MAKE文件的语法》，《vs6 IDE导出的makefile使用方法》，《教你在VS2005下写MakeFile》

十一．代码的编辑与阅读导航

Visual Studio本身就带有充分的代码组织、编辑、分析以及阅读能力的支持。

VC6.0的“Project Settings→C/C++→Generate Browse info”和“Project Settings→Browse info→Build browse info file”可生成向导文件，FileView和ClassView需要这些信息导航。

VC6.0和VC2005中的“Edit→Find in Files”支持指定项目、工程或文件内的关键字搜索。

VC6.0的“Tools→Source Browser”（Alt+F12），提供对“Definitions and References”、“File Outline”、“Base Classes and Members”、“Derived Classes and Members”、“Call Graph”、“Callers Graph”等导航功能支持。VC2005侧边栏“类视图”中支持显示“基类型”、“派生类型”、“继承成员”等功能，右键快捷菜单中的“调用浏览器”支持函数调用关系图和被调用关系图的查看。

编辑时，VC6右键快捷菜单支持“List Members”、“Parameter Info”、“Type Info”、“Complete Word”。对应VC2005文本编辑器工具栏中的“显示对象成员列表”、“显示参数信息”、“显示快速信息”、“显示文字自动完成”。

VC2005右键快捷菜单中支持“查找所有引用”，配合代码定义窗口，可很方便地浏览代码。VC2005标准工具栏中的“向后定位/向前定位”支持定位功能。此外，在阅读代码时，我们可以使用书签（Bookmark）来设置锚点，这样就可以实现基于书签的自定义索引导航。

下面介绍常用的编程辅助插件软件及源代码层次结构分析工具软件。

1.Visual Assist X是一款非常好的Microsoft Visual Studio辅助插件，支持C/C＋＋，C＃，ASP，Visual Basic，Java和HTML等语言，能自动识别各种关键字、系统函数、成员变量、自动给出输入提示、自动更正大小写错误、自动标示错误等，有助于提高开发过程的自动化和开发效率。例如VA Outline中右键→Refactor→Rename能够快速更新变量命名。

2．Windows下的Source Insight是一个面向项目开发的程序编辑器和代码导航浏览器，它还能显示reference trees，class inheritancediagrams和call trees。相应的Linux下阅读源代码的利器Source Navigator能很好地解决文件定位和跳转问题。

3．Rational Rose是利用UML（统一建模语言）进行分析和设计面向对象软件系统的强大的可视化工具，可以进行项目需求分析、结构规划和生成框架代码，还可以支持从现有系统逆向转出工程代码，生成Rose模型的功能。

逆向工程（Reverse Engineer）就是从现有系统的代码来生成模型的功能。通过Rational Rose分析已有的代码，可了解代码结构和数据结构，生成类层次关系图、数据模型图和组件图。

十二. VC常用快捷键

Alt+F7：Project Settings

Ctrl+J/K：寻找上一个/下一个预编译条件

Ctrl+Shift+J/K：寻找上一个/下一个预编译条件并将这一块选定

Ctrl+Tab：当前打开文档和上一次打开文档切换。

Alt+O：头文件与cpp文件的交互显示

F12：Goto definition

Shift+F12：Goto reference

Ctrl+Home：文档头

Ctrl+Shift+Home：选定从当前位置到文档头

Ctrl+End：文档尾

Ctrl+Shift+End：选定从当前位置到文档尾

Alt+F8：自动格式重排

Ctrl+F4：关闭当前文档

Ctrl+Z/Alt+Backspace：Undo

Ctrl+Shift+Z/Ctrl+Y：Redo

Ctrl+B/Alt+F9：编辑断点

Ctrl+Shift+F9：去掉所有断点

Ctrl+D：定位工具条查找输入栏

Alt+F3/Ctrl+F：当前文档查找

Ctrl+F3：查找下一个

Ctrl+Shift+F3：查找上一个

Ctrl+H：替换

Ctrl+]：括号匹配

Ctrl+↑：向上滚屏

Ctrl+↓：向下滚屏

Ctrl+←：移到前一个单词

Ctrl+→：移到后一个单词

Ctrl+Shift+←：选定当前位置到前一个单词

Ctrl+Shift+→：选定当前位置到后一个单词

Ctrl+Shift+]：括号匹配块选

Ctrl+Shift+F8：配合鼠标和方向键实现竖直块选

Ctrl+Backspace：删除当前单词的前半截（以光标为分割）

Ctrl+Del：删除当前单词的后半截（以光标为分割）

Ctrl+L：剪切当前行

Ctrl+Shift+L：删除当前行

F2：上一个书签

Shift+F2：上一个书签

Alt+F2：编辑书签

Ctrl＋F2：添加/删除一个书签

Ctrl+Shift+F2：清除所有书签

参考：

《VC的环境设置》

《VC IDE使用技巧》

《VC的工程文件说明》

《VC6.0工程设置介绍》

《Visual C++ 6.0环境配置》

《你可能还不熟悉的VC6.0 IDE》

《VC 2005 解决方案的目录结构设置和管理》

《Managing Solutions, Projects, and Files》

《File Types Created for Visual C++ Projects》