Demo Project Setup in Visual Studio 2005

By Frank Luna

This article describes how to create and build the demo projects for the book *Introduction to 3D Game Programming with DirectX 9.0c: A Shader Approach* using Visual Studio 2005 (VS05). As a working example, we will show how to create and build the *Hello Direct3D* demo of Chapter 4.

It is assumed that the reader has already successfully downloaded and installed the latest version of the DirectX 9 SDK (available at http://msdn.microsoft.com/directx/), which is needed to program DirectX applications. The installation of the SDK is straightforward, and the installation wizard will walk you through it.

1 Create a Win32 Project

First launch VS05, then go to the main menu and select **File->New->Project,** as shown in Figure 1.

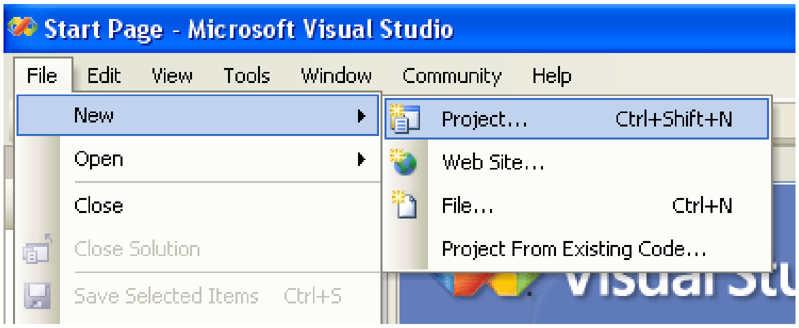
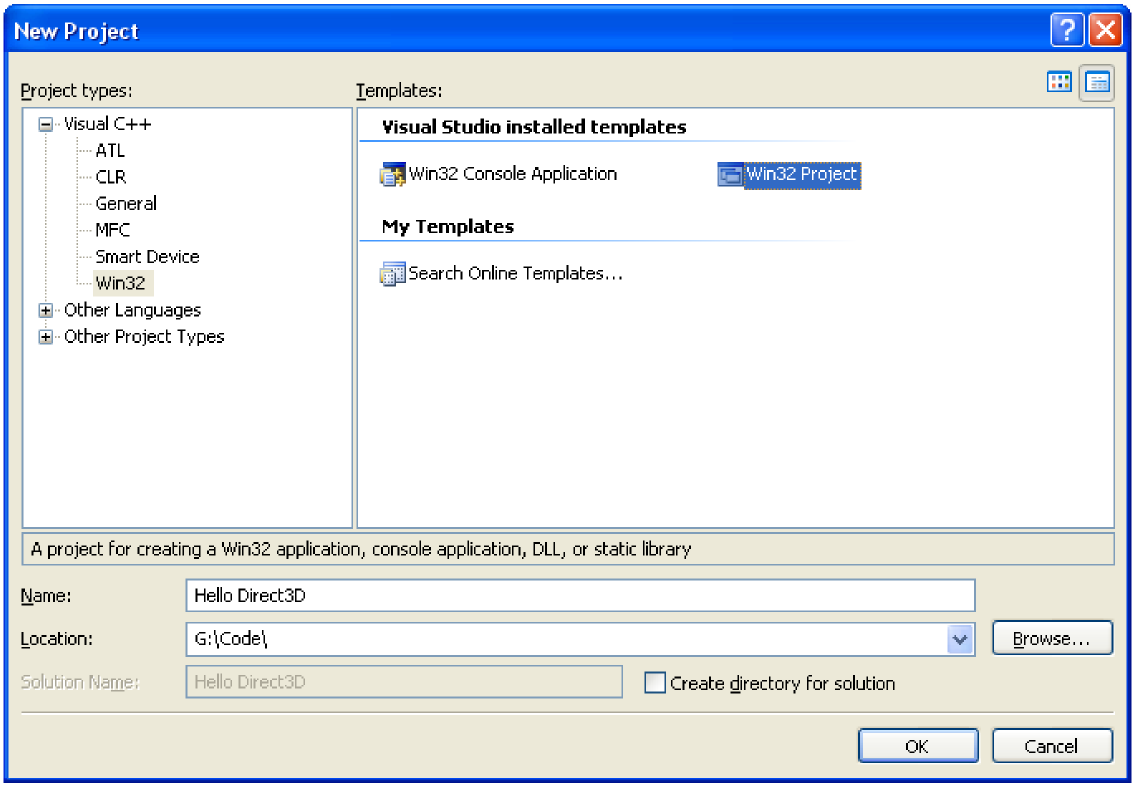


Figure 1

The New Project dialog box will appear (Figure 2). Select **Win32** from the Visual C++ Project types tree control on the left. On the right, under Visual Studio installed templates, select **Win32 Project.** Next, give the project a name and specify the location you wish to store the project folder. Also uncheck **Create directory from solution,** if it is initially checked by default. Now hit **OK.**

A new dialog box will appear. On the left, there are two options: Overview and Application Settings. Select **Application Settings,** which produces the dialog box shown in Figure 3. From here, be sure that **Windows application** is chosen, and the **Empty project** box is checked. Now press the **Finish** button. At this point, you have successfully created an empty Win32 project, but there are still some things to do before you can build a DirectX project demo.



Project types:

- Visual C++
  - ATL
  - CLR
  - General
  - MFC
  - Smart Device
  - Win32
- Other Languages
- Other Project Types

Templates: Win32 Project

A project for creating a Win32 application, console application, DLL, or static library

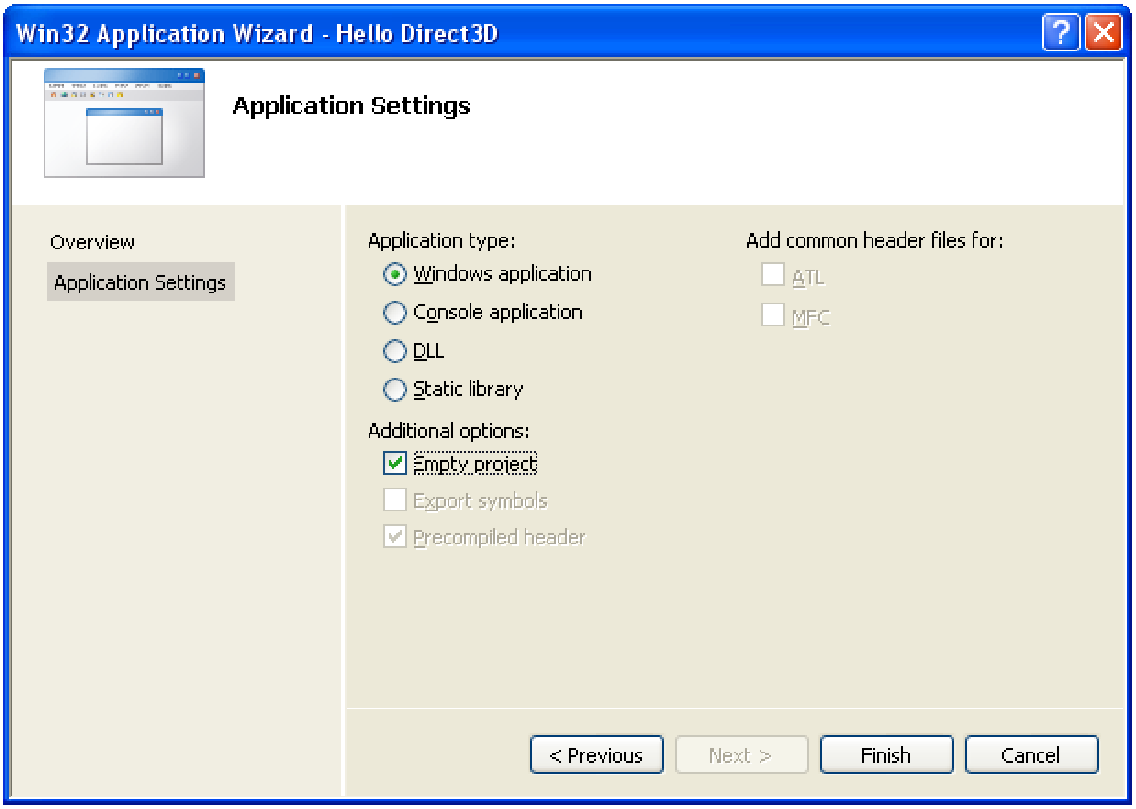
Name: Hello Direct3D

Location: G:\Code\

Solution Name: Hello Direct3D

□ Create directory for solution

Figure 2



Application type:

● Windows application
○ Console application
○ DLL
○ Static library

Additional options:

☑ Empty project
□ Export symbols
☑ Precompiled header

Add common header files for:

□ ATL
□ MFC

Figure 3

2 Setting up Libraries and Headers

We now need to link the following DirectX libraries to our project: *d3d9.lib d3dx9d.lib dxguid.lib DxErr9.lib dinput8.lib*. Note that *d3dx9d.lib* is the debug version of the D3DX library. For release projects, just use *d3dx9.lib*.

To link the library files, right click the project name under the Solution Explorer and select **Properties** from the dropdown menu (Figure 4). This launches the dialog box shown in Figure 5. From the left-hand-side tree control, select **Configuration Properties->Linker->Input.** Then on the right-hand-side, specify the library file names on the **Additional Dependencies** line. Press **Apply** and then **OK.**

We now need to make sure Visual Studio knows the directories in which to search for DirectX header and library files. If you install the DirectX SDK after you installed VS05, the DirectX installation should automatically configure VS05 to include the necessary paths for the DirectX header and library files. Nevertheless, to make sure, or to specify the directories manually, go to the VS05 main menu, and select **Tools->Options.** From the left-hand-side tree control, scroll down and select **Projects and Solutions->VC++ Directories.** Then on the right-hand-side, select **Include files** and **Library files** from the **Show directories for** combo box, and make sure the correct DirectX path is set, as in Figure 6 and 7. Note that the exact path depends on where you installed the DirectX SDK.

We have now linked the necessary DirectX library files to our project, and have configured the VS05 search paths so that it knows where to find the necessary DirectX library and header files on your system. We are almost ready to build the demo.

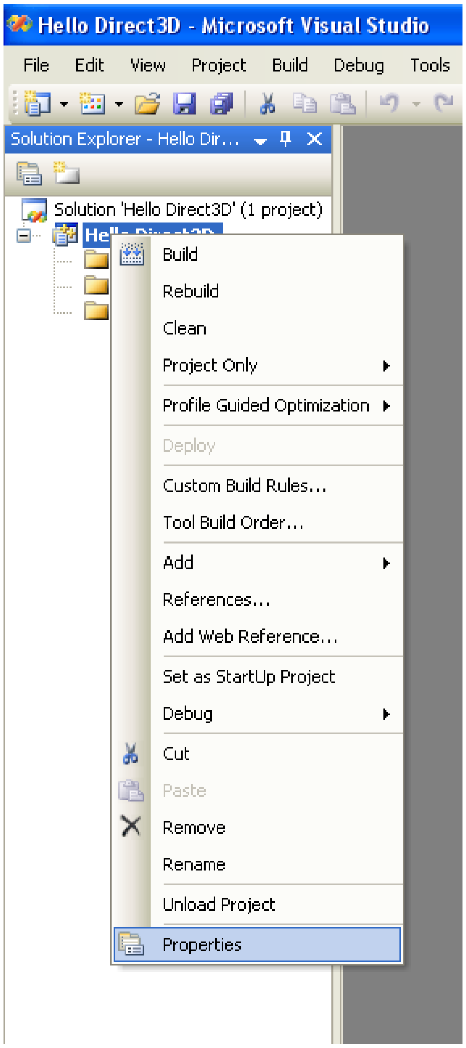
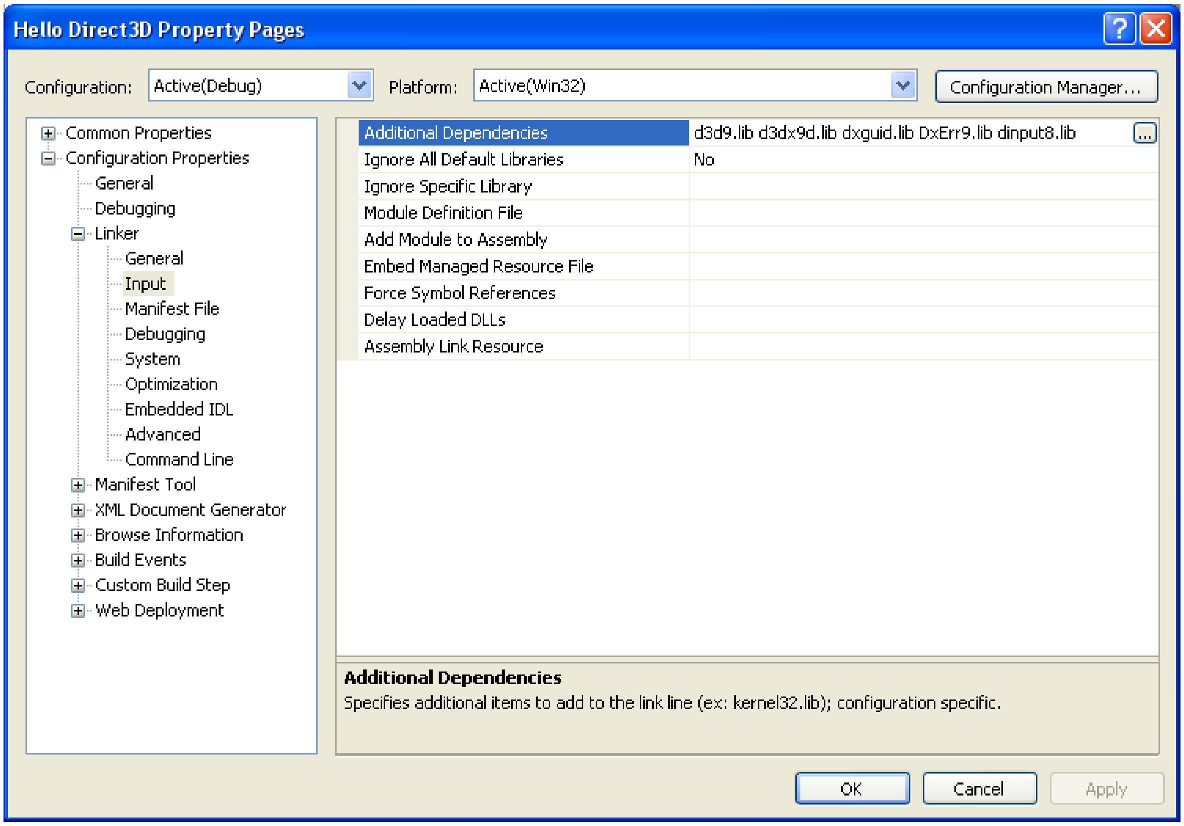


Figure 4



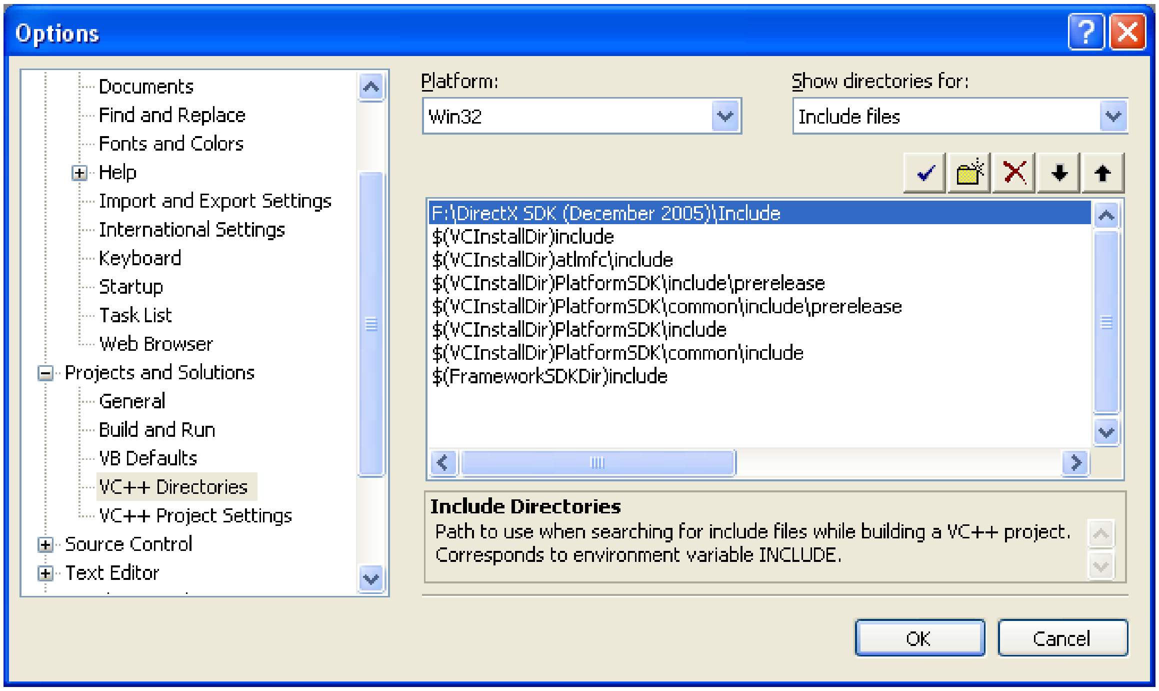
Configuration: Active(Debug)

Platform: Active(Win32)

Additional Dependencies	d3d9.lib d3dx9d.lib dxguid.lib DxErr9.lib dinput8.lib
Ignore All Default Libraries	No
Ignore Specific Library	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded DLLs	
Assembly Link Resource	

Additional Dependencies
Specifies additional items to add to the link line (ex: kernel32.lib); configuration specific.

Figure 5



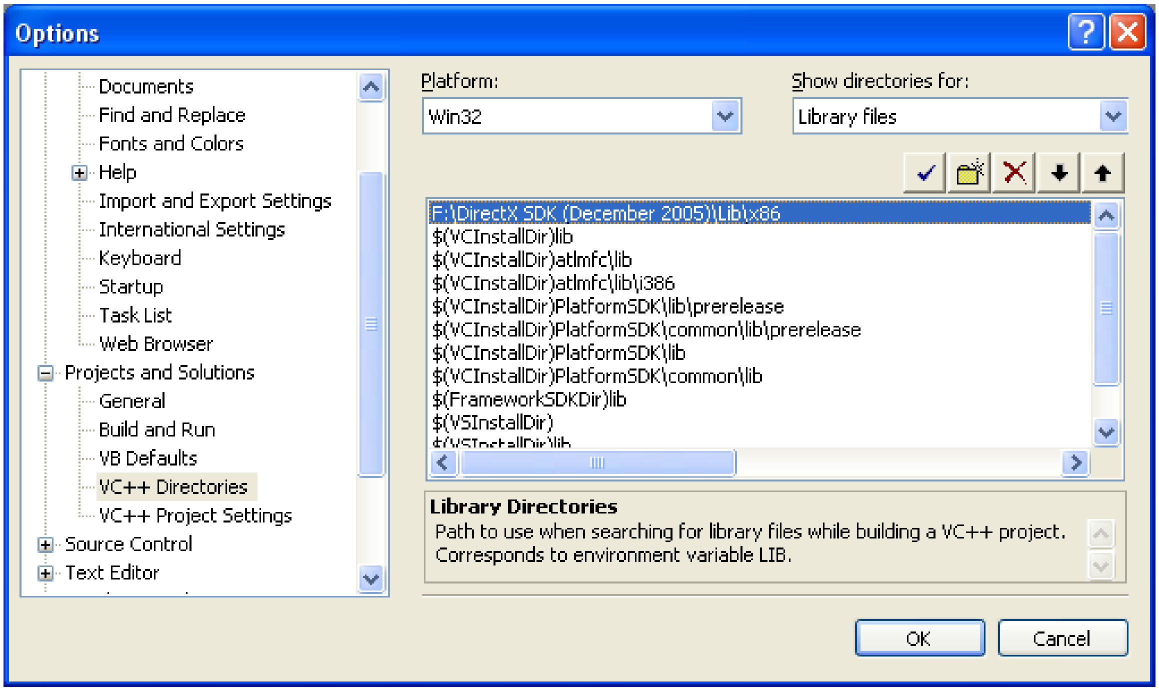
Platform: Win32

Show directories for: Include files

```
F:\DirectX SDK (December 2005)\Include
$(VCInstallDir)include
$(VCInstallDir)atlmfc\include
$(VCInstallDir)PlatformSDK\include\prerelease
$(VCInstallDir)PlatformSDK\common\include\prerelease
$(VCInstallDir)PlatformSDK\include
$(VCInstallDir)PlatformSDK\common\include
$(FrameworkSDKDir)include
```

Include Directories
Path to use when searching for include files while building a VC++ project. Corresponds to environment variable INCLUDE.

Figure 6



Platform: Win32

Show directories for: Library files

```
F:\DirectX SDK (December 2005)\Lib\x86
$(VCInstallDir)lib
$(VCInstallDir)atlmfc\lib
$(VCInstallDir)atlmfc\lib\i386
$(VCInstallDir)PlatformSDK\lib\prerelease
$(VCInstallDir)PlatformSDK\common\lib\prerelease
$(VCInstallDir)PlatformSDK\lib
$(VCInstallDir)PlatformSDK\common\lib
$(FrameworkSDKDir)lib
$(VSInstallDir)
$(VSInstallDir)lib
```

Library Directories
Path to use when searching for library files while building a VC++ project. Corresponds to environment variable LIB.

Figure 7

3 Use Multi-Byte Character Set

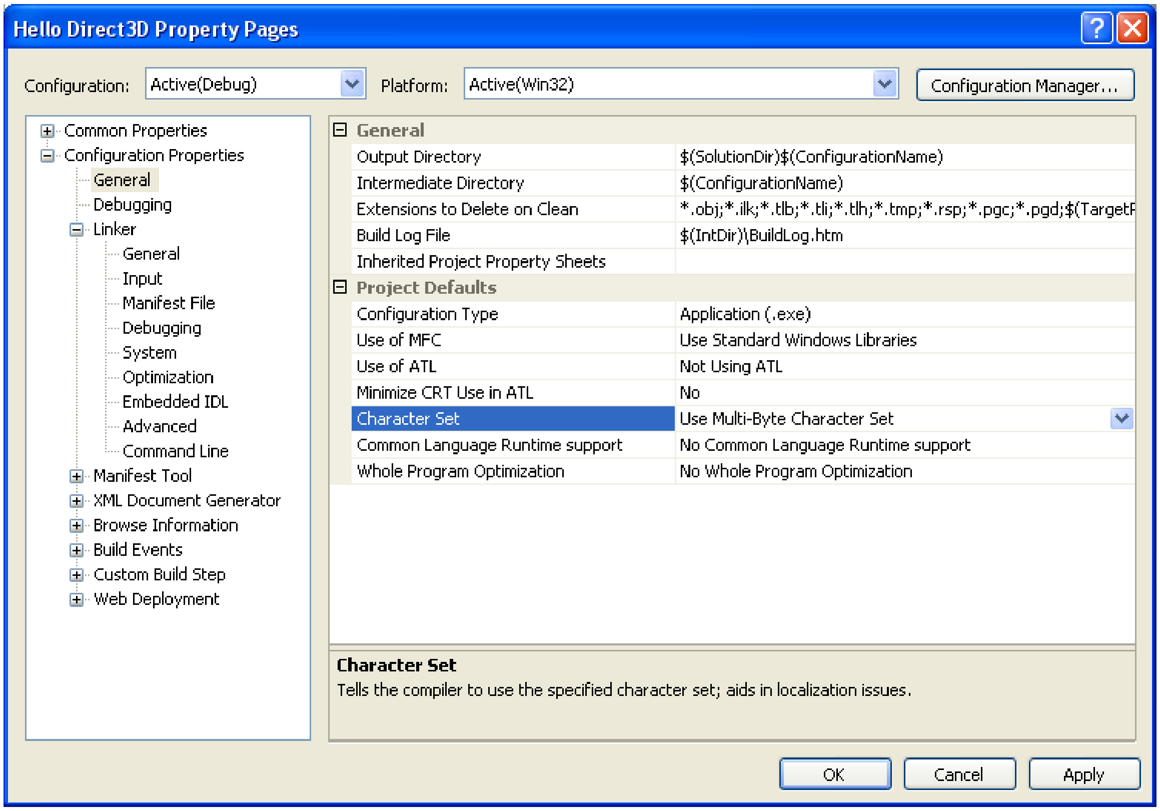
VS05 defaults to the Unicode character set, however, the demo programs were written using the multi-byte character set. Therefore, we need to switch the character set in VS05. Go to the project properties page, as Figure 4 showed how to do. Then on the left tree control, select **Configuration Properties->General.** Then on the right, find the **Character Set** line, and switch to **Use Multi-Byte Character Set,** as Figure 8 illustrates.

Note that it is relatively easy to modify the demos to support both Unicode and Multi-Byte. One way is to define:

```
#include <tchar.h>
#include <string>

typedef std::basic_string<TCHAR> tstring;
```

And then use `tstring` in place of any `std::string` type used in the demo programs. Additionally, for string literals, you need to wrap them in the `_T` macro, e.g., `_T("Hello, World!")`. The compiler will automatically switch `TCHAR` to either a Unicode character or Multi-Byte character based on the compiler settings. In this way, your project effectively supports both character sets.



Configuration: Active(Debug)   Platform: Active(Win32)

General	
Output Directory	$(SolutionDir)$(ConfigurationName)
Intermediate Directory	$(ConfigurationName)
Extensions to Delete on Clean	*.obj;*.ilk;*.tlb;*.tli;*.tlh;*.tmp;*.rsp;*.pgc;*.pgd;$(TargetP
Build Log File	$(IntDir)\BuildLog.htm
Inherited Project Property Sheets	
Project Defaults	
Configuration Type	Application (.exe)
Use of MFC	Use Standard Windows Libraries
Use of ATL	Not Using ATL
Minimize CRT Use in ATL	No
Character Set	Use Multi-Byte Character Set
Common Language Runtime support	No Common Language Runtime support
Whole Program Optimization	No Whole Program Optimization

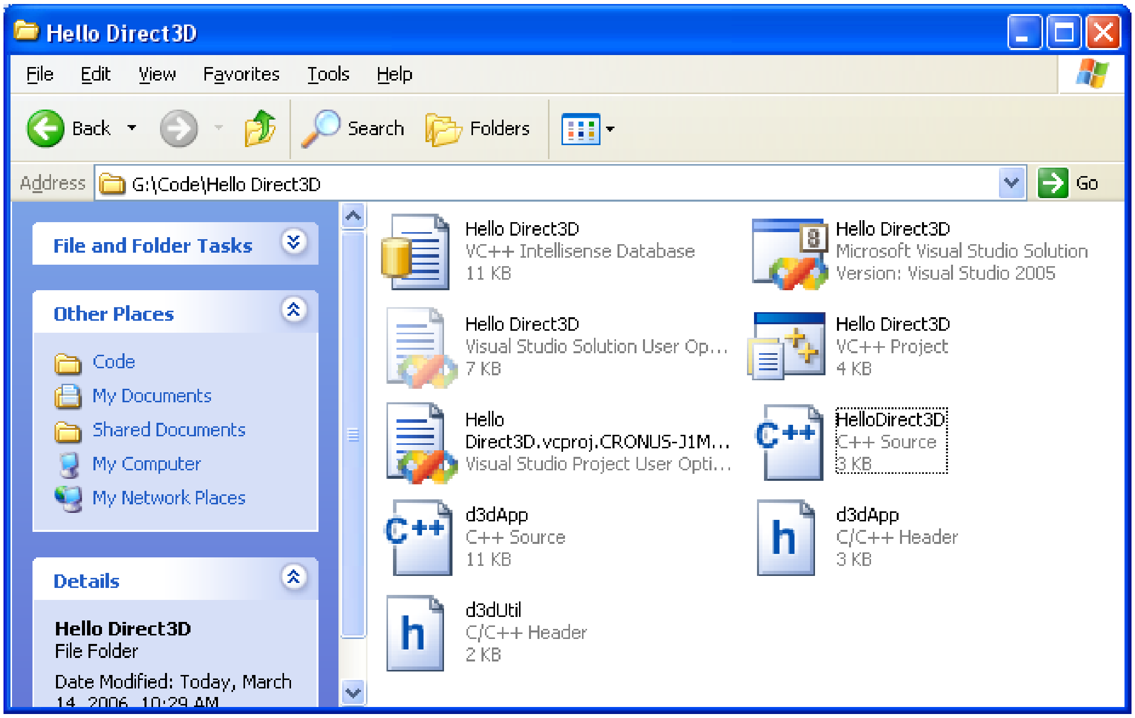
Character Set
Tells the compiler to use the specified character set; aids in localization issues.

Figure 8

4 Adding the Source Code and Building

Finally, our project setup is complete. We can now add our source code files to the project and build it. First, copy the *Hello Direct3D* source code files to your project's directory (Figure 9). Now, in VS05, go to the main menu and select **Project->Add Existing Item.** From the dialog box, Figure 10, select all four source code files and press **Add.** The source code files are now part of the project. You can now go to the main menu, and select **Debug->Start Debugging** to compile, link, and execute the demo.

This concludes the tutorial. If you have any further questions, please post in the www.moon-labs.com forums.

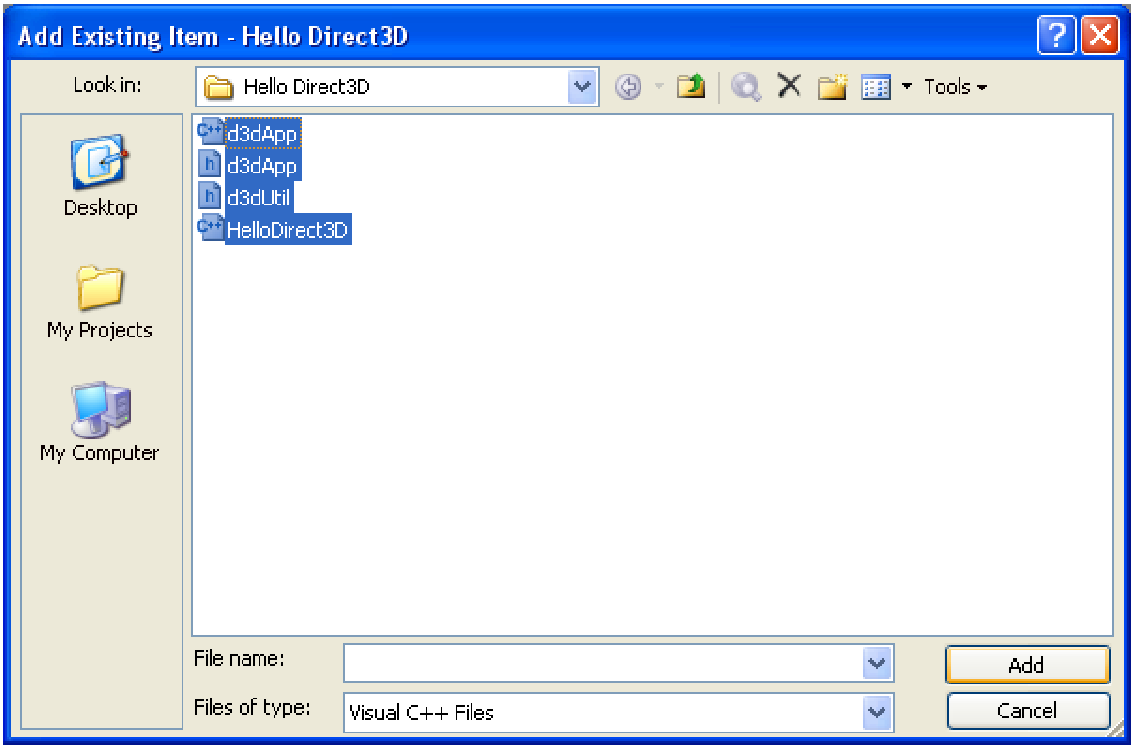


Address: G:\Code\Hello Direct3D

Hello Direct3D
File Folder
Date Modified: Today, March
14, 2006, 10:29 AM

Hello Direct3D VC++ Intellisense Database 11 KB	Hello Direct3D Microsoft Visual Studio Solution Version: Visual Studio 2005
Hello Direct3D Visual Studio Solution User Op... 7 KB	Hello Direct3D VC++ Project 4 KB
Hello Direct3D.vcproj.CRONUS-J1M... Visual Studio Project User Opti...	HelloDirect3D C++ Source 3 KB
d3dApp C++ Source 11 KB	d3dApp C/C++ Header 3 KB
d3dUtil C/C++ Header 2 KB	

Figure 9



Look in: Hello Direct3D

d3dApp
d3dApp
d3dUtil
HelloDirect3D

File name:

Files of type: Visual C++ Files

Figure 10