MICROSOFT VISUAL C++ .NET TUTORIAL

INTRODUCTION

Microsoft Visual C++ .NET allows you to create many different types of applications. This guide addresses creating and using Console Applications. A console application is a program that runs inside a DOS window. This guide is divided into the following sections:

- Starting Visual C++ .NET
- Creating and Executing a Single-File Console Application
- Importing and Executing C++ Source Code Files

STARTING VISUAL C++ .NET

To start the Microsoft Visual C++ compiler, click the **Start** button on the task bar to open the start menu. Open the **Programs** menu (**All Programs** in Windows XP) and select the **Microsoft Visual Studio .NET 2003** folder, then the program **Microsoft Visual Studio .NET 2003**. If the Microsoft Visual Studio window is not maximized, click the **maximize** button in the upper right corner of the title bar.

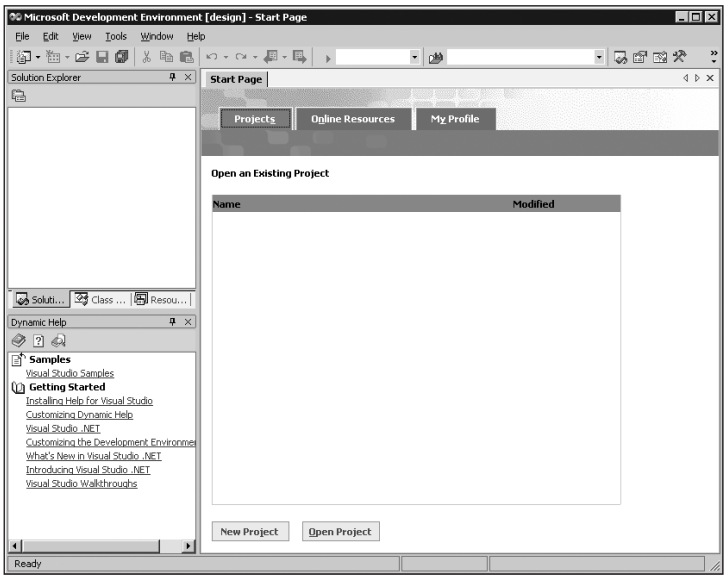

Figure 1-1 Microsoft Visual Studio

Figure 1-1 shows the initial application window. If your application window looks different, you may need to adjust the settings. You can change the settings by clicking the **Start Page** tab and selecting the **My Profile** option. There you can set various default settings. First, select **Visual C++ Developer** in the top drop-down box under the heading **Profile**. Also, set **Show Help** to External Help, as shown in Figure 1-2. A dialog box will pop up with a warning that the change to External Help will not take place until Visual Studio is restarted. Click the **OK** button and continue. Figure 1-3 shows how the profile looks after making these selections.

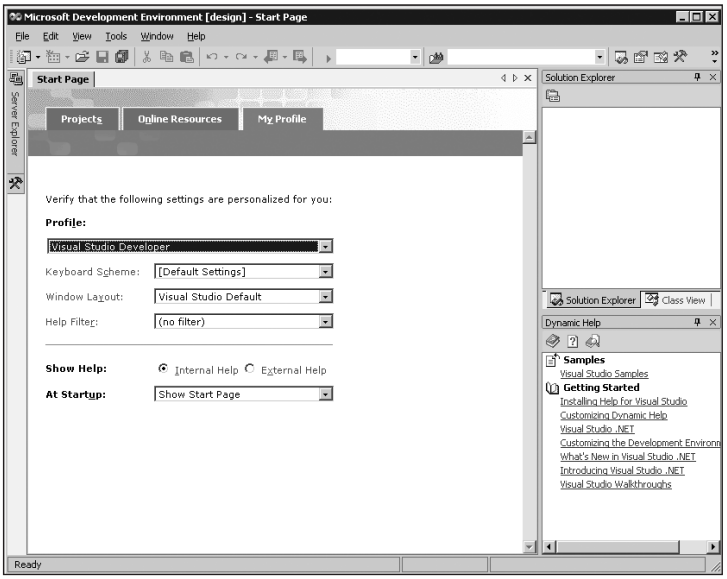

Figure 1-2 Initial profile settings

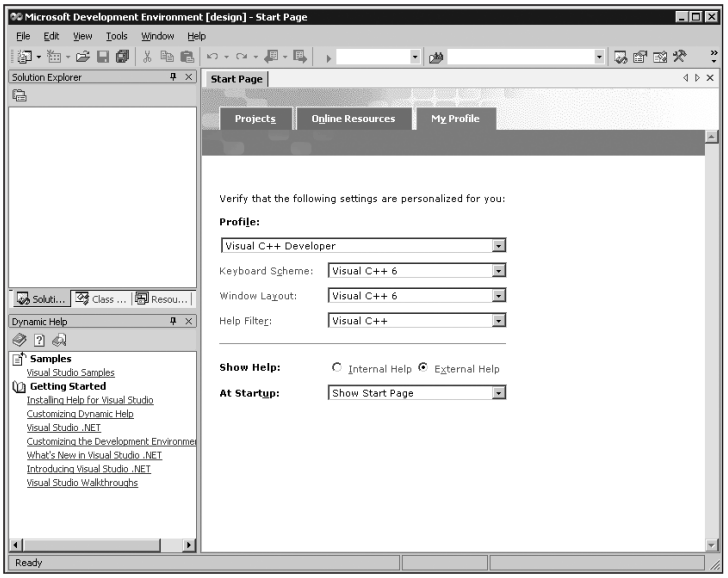

Figure 1-3 C++ developer profile settings

Creating and Executing a Single-File Console Application

One of the most common programs encountered in an introductory C++ programming class is a single source code file that contains the main function. To construct this type of application, you need to:

- Create a Win32 Console Application Project
- Add a source code file
- Write the program
- Execute the program
- Debug the program

Create a Win32 Console Application Project

Start Microsoft Visual Studio .NET 2003. On the File menu select **New,** then **Project**. (See Figure 1-4.)

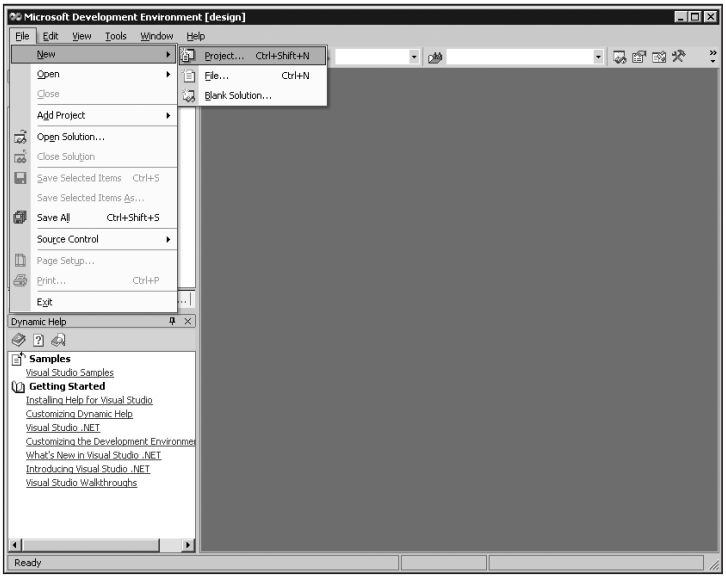

Figure 1-4 Create a new project

When the New Project dialog box opens, click on **Visual C++ Projects** in the Projects Types pane, then on **Win32 Console Project** in the Templates pane. Enter the following information (see Figure 1-5):

- Enter the project name in the Name textbox
- Select the location for the project in the Location textbox

The application provides a default location for saving projects or you can select your own location by pressing the button to the right of the location textbox to open the Choose Directory dialog box.

Press the **OK** button after entering the required information into the dialog box. When the Win32 Application Wizard appears, click on Application Settings, select **Console application** from the Application types and check **Empty project** (see Figure 1-6) and then press the **Finish** button. The Solution Explorer now contains the information about your project (see Figure 1-7.)

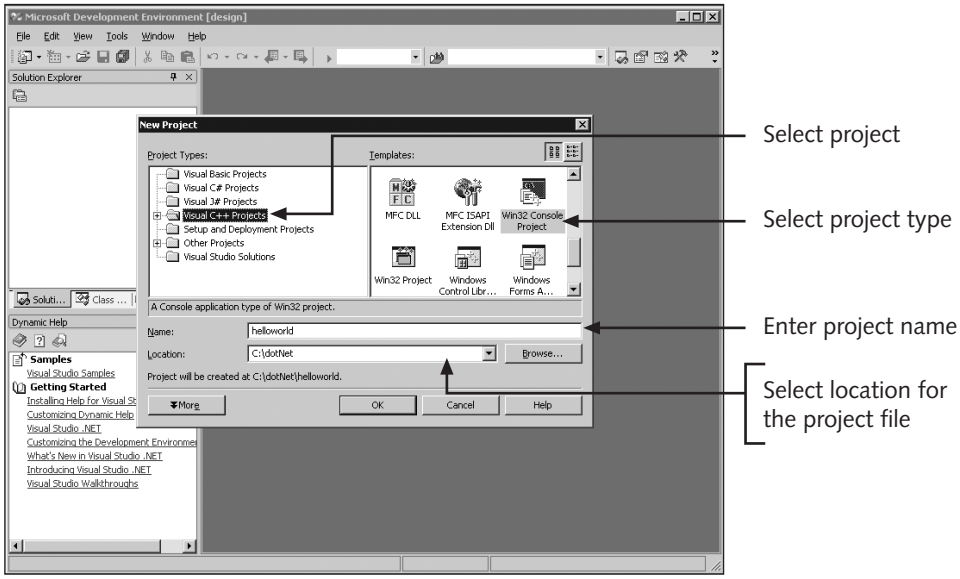

Figure 1-5 New project settings

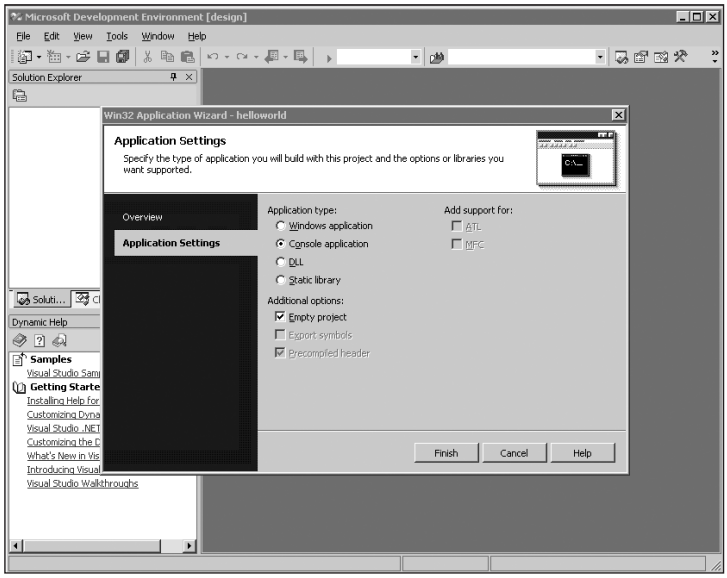

Figure 1-6 Win32 Console Application Wizard

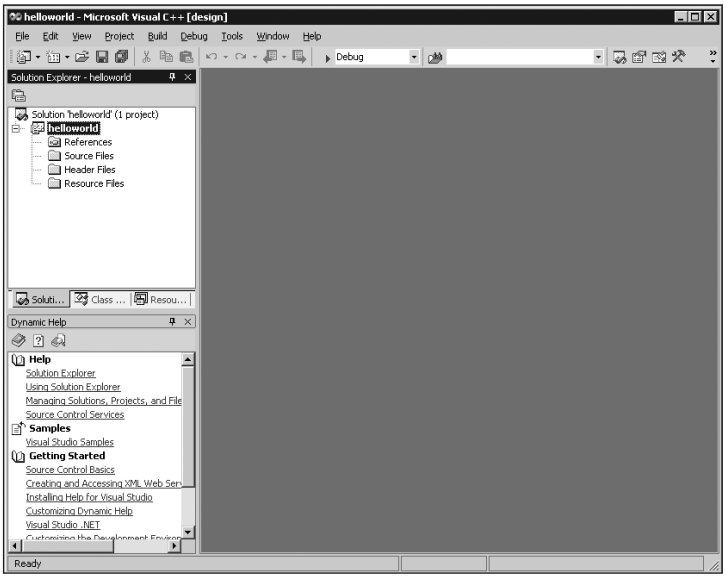

Figure 1-7 Empty Win32 Console Application

Add a Source File

Add a blank source code file to your project. On the File menu select **New,** then **File**. (See Figure 1-8.)

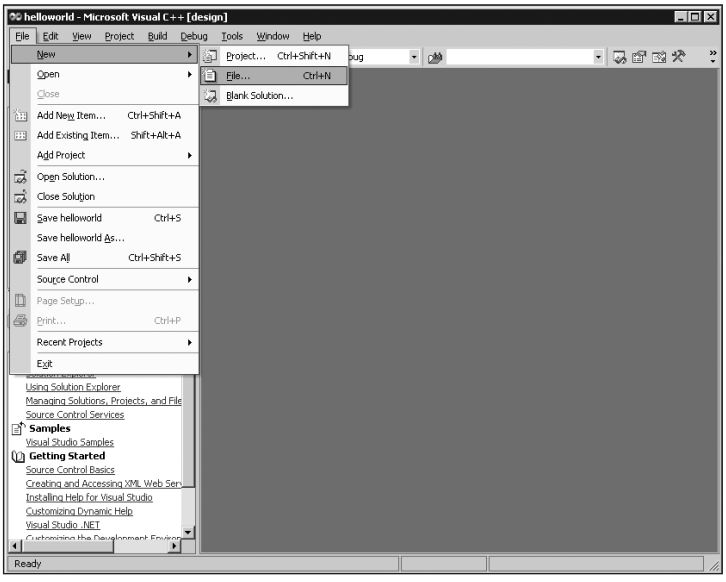

Figure 1-8 Create new file

When the New File dialog box appears, select **Visual C++** in the Categories pane and **C++ File (.cpp)** in the Templates pane. Click the Open button (see Figure 1-9).

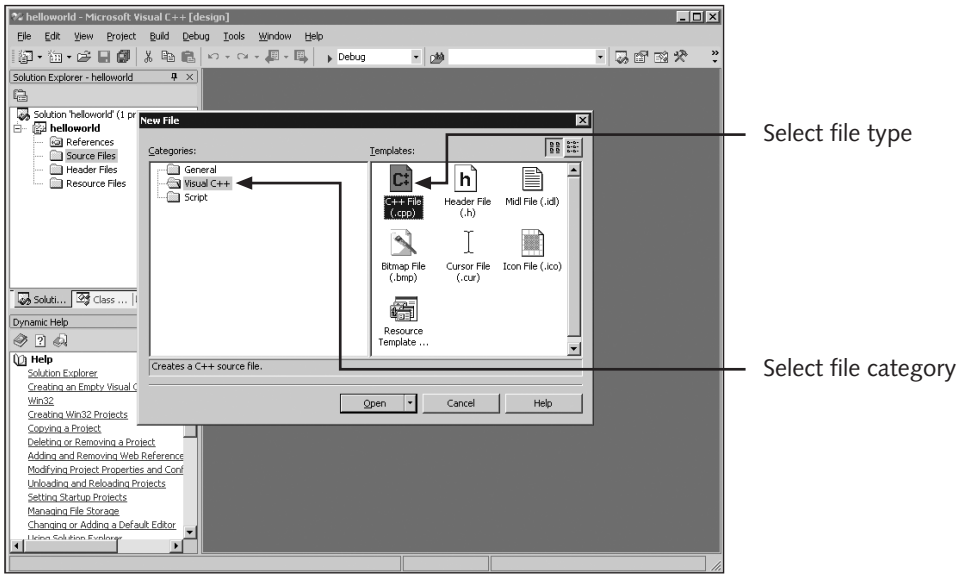

Figure 1-9 New file settings

To save the file with a specific name, on the File menu select **Save Source1 As...**. When the Save File As dialog box appears (see Figure 1-10), select the folder to hold the file (it generally should be the same as the project name) from the Save in dropdown window, then type in a name (the .cpp extension will be added automatically.) Press the **Save** button to complete the action.

Write the Program

Type the source code for your program into the blank editing pane located in the right side of the window. Figure 1-11 contains the source code for a simple C++ console program that displays the phrase, "Hello World .NET."

Note that the C++ editor automatically selects different colors for C++ reserved words and provides automatic indentation for blocks of code. Also note that as you are typing various C++ keywords, the Dynamic Help in the lower left pane will change to reflect the item you are currently working on. Clicking on any item in the Dynamic Help will bring up more information on that topic in the Help window.

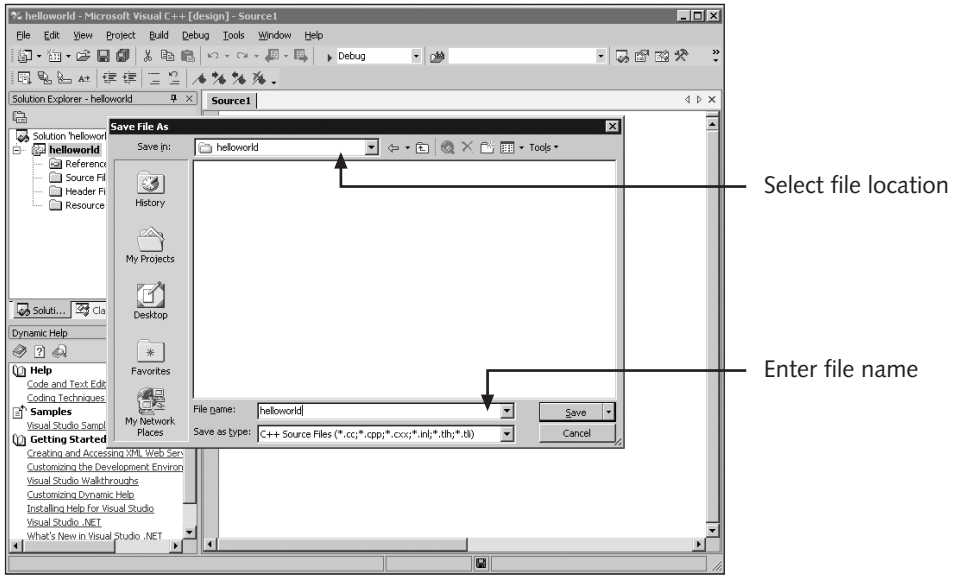

Figure 1-10 Naming project files

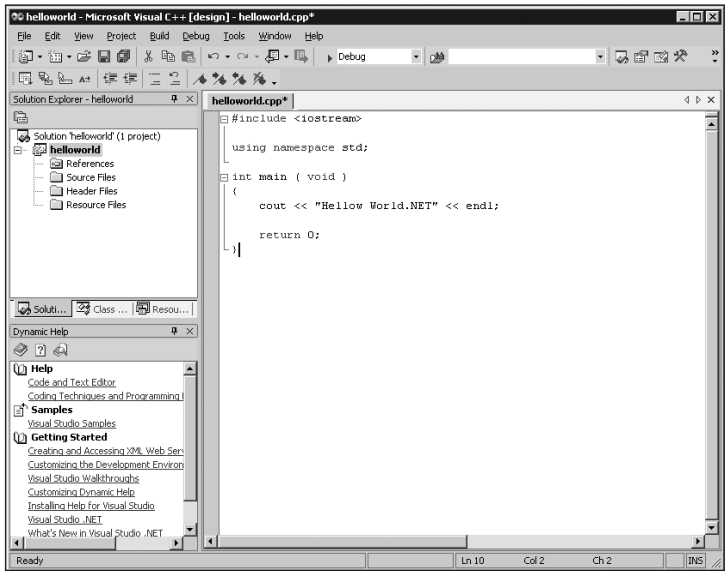

Figure 1-11 Hello World source code file

Add the Code File to the Project

Before you can compile or build the program, you must add the code file (.cpp file) to the project. Right-click anywhere in the text file window (right hand pane), select the last context menu item, **Move helloworld.cpp into Project**, then click on your project name when it appears to the side. (See Figure 1-12.) You will see that the code file is added to the Source Files folder in the Solution Explorer pane (upper left side; you may have to click the + symbol to open the folder.)

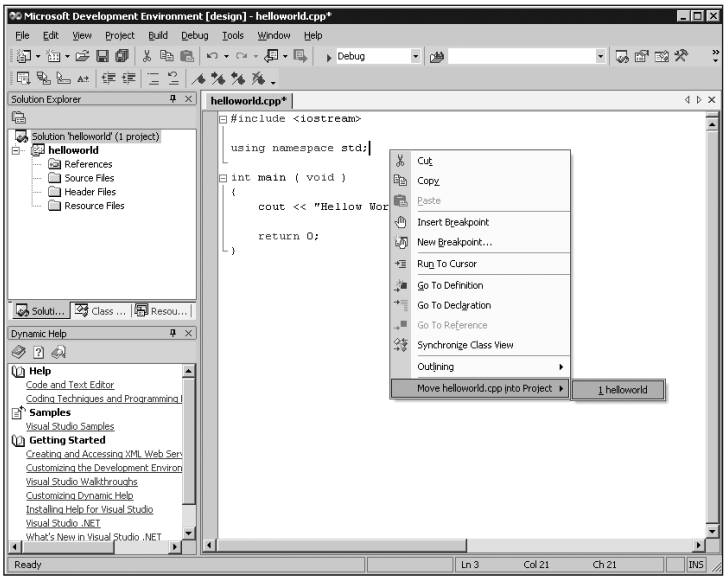

Figure 1-12 Adding file to project

Compile and Build the Program

Before you can execute the program you have written, you must save it and compile it. To save the file, either click in the edit window to make it the active object or click on the helloworld.cpp filename in the Solution Explorer, the select **Save helloworld.cpp** from the File menu.

Once the program is saved, you compile it by selecting the **Build Solution** option from the Build menu. (See Figure 1-13.) This will cause Visual Studio to both compile and link your program. Once the program compiles and links, the results will be shown in the Output window at the bottom. (See Figure 1-14.) In this example, the program compiled with no errors and no warnings.

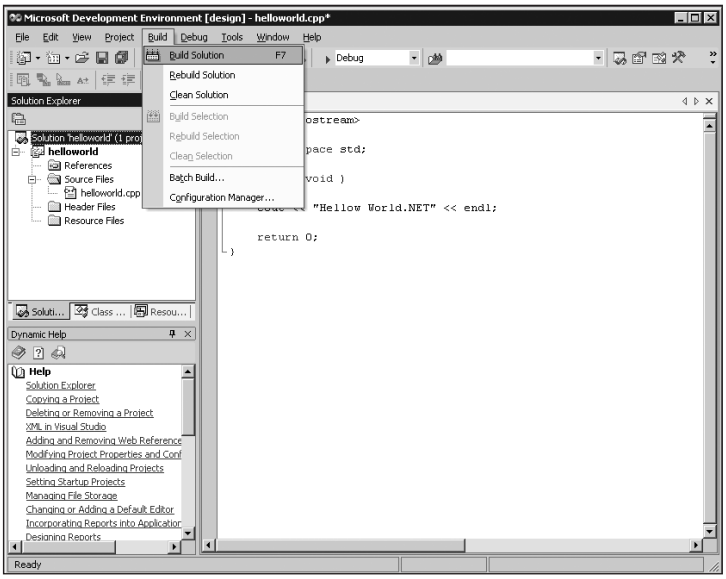

Figure 1-13 Compiling a program

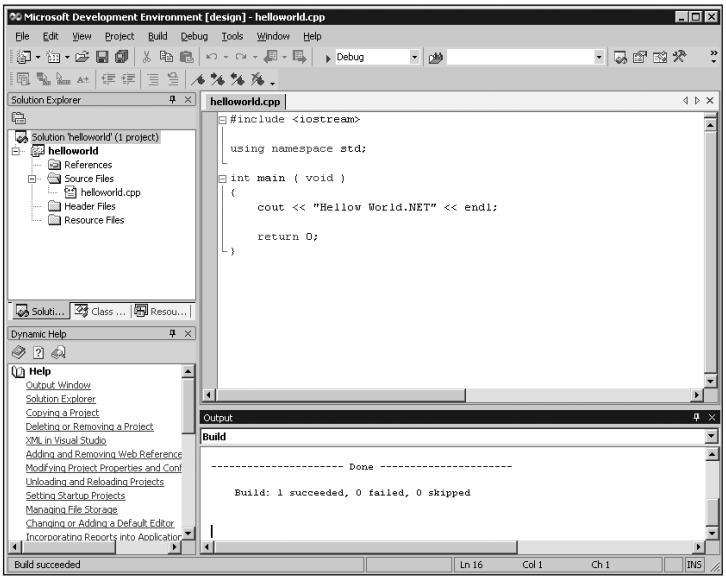

Figure 1-14 Results of compiling the program

After the build process has successfully completed, you can now execute the program by selecting **Start Without Debugging** from the Debug menu. (See Figure 1-15)

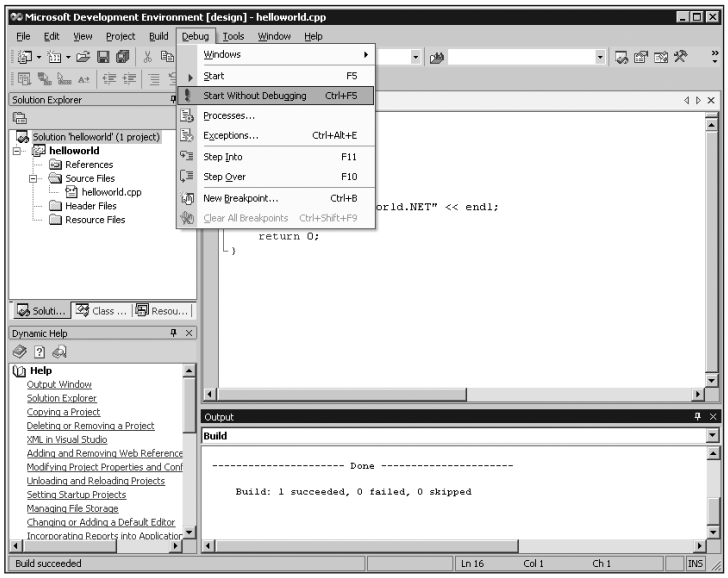

Figure 1-15 Initiating program execution

The program results will appear in a new DOS window. Notice that the phrase "Press any key to continue" has been added to the program output. This additional code was added to keep the DOS window open until you have had a chance to view the output and press a key on the keyboard. (See Figure 1-16.) Once a key on the keyboard is pressed, the program stops execution and the DOS window closes. This output phrase is not actually included in your program, as you would see if you opened a DOS window and executed the program directly.

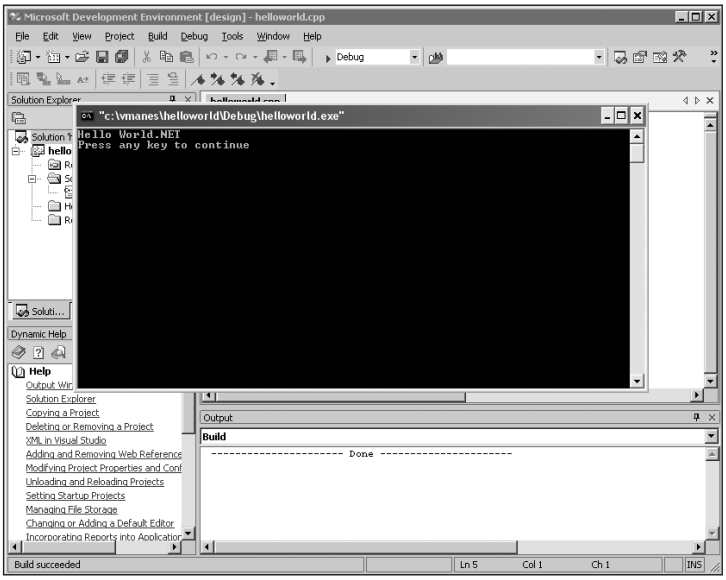

Figure 1-16 Program results

Debug the Program

Often, the program will have errors when you attempt to compile, build, and execute it. The program that was successfully executed in the previous example has been modified to include an error—the semi-colon at the end of the return 0 statement has been removed. When the modified program is compiled, an error is displayed in the output window. (See Figure 1-17.)

You can determine where the compiler found the error by scrolling up in the Output window until the specific error message is visible, then by double-clicking on the error message in the Output window. This will cause a pointer to appear in the left margin of the source file where the error was encountered. (See Figure 1-18.)

Notice in this case that the pointer is on the line after the line containing the actual error. This occurs when the error induces a compiler recognized fault on a subsequent line. While not always exact, the error pointer, in conjunction with the error description, can help locate errors in the source code. Once the error is identified and corrected, the program must be saved, rebuilt, and executed again.

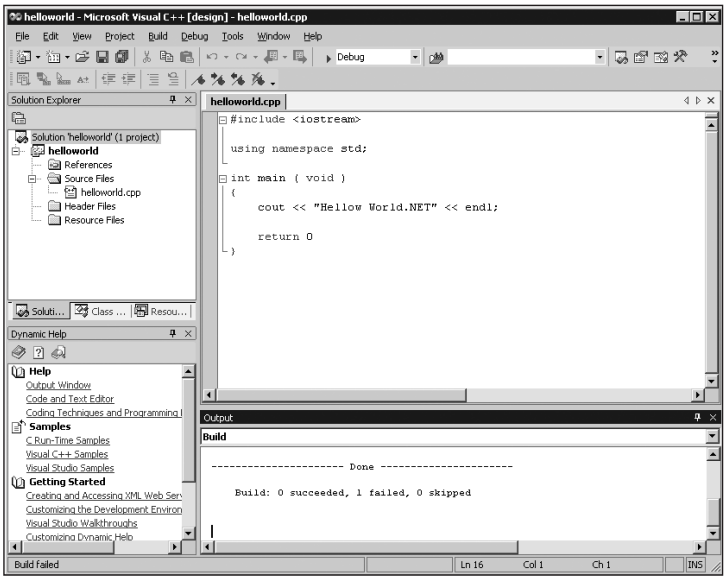

Figure 1-17 Compile error

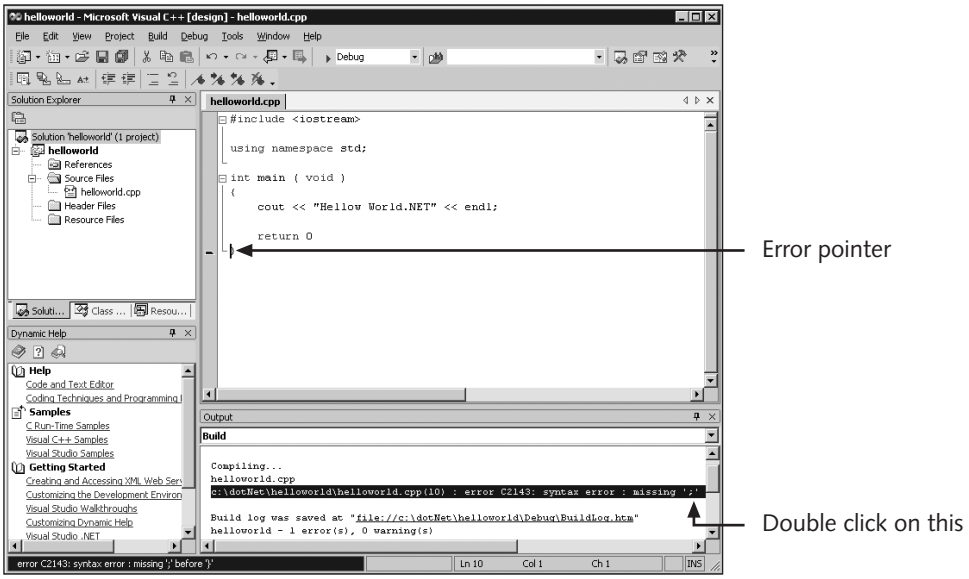

Figure 1-18 Locating error

Importing and Executing C++ Source Code Files

Often you will want to compile, build, and execute programs that you did not write, or that you wrote and stored in other locations. Consider, for example, the case where you want to execute a source file provided as part of a textbook's supplemental information. The steps are nearly identical to those described above. First you need to construct a Project file as described above. Next, instead of adding a new source file using the New command as describe above, you import the existing file into the project. This is accomplished by selecting the **Add Existing Item...** option located under the Project menu. (See Figure 1-19.)

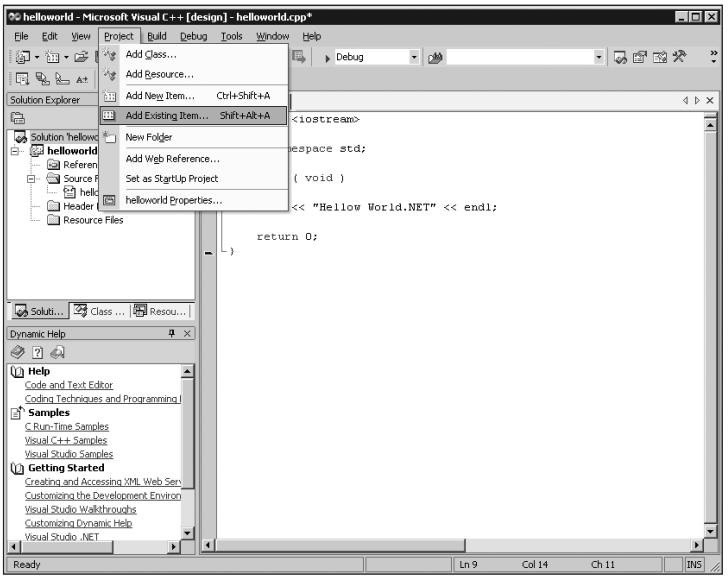

Figure 1-19 Importing a file into a project

Once the file is part of the project (it will be listed in the Source Files folder in the Solution Explorer pane) you can double-click its name to display it in the editor pane. Then you can compile, build, and execute the program as described above.

SUMMARY

Microsoft Visual C++ .NET allows you to create many different types of applications. This guide addressed creating and using single source file Console Applications, but the basic operations are the same for more complex programs.

- Always start by creating a project file of the appropriate type
- Create blank files for writing your own programs or import existing files into the project
- Compile
- Build
- Execute
- Debug, if necessary