Chapter 2

Nikon F3HP. Kodak Tri-X

Artifacts

Small Victories Creating C# Projects

Learning Objectives

- *List and describe the minimum development tools required to create C# programs*
- *State the purpose and use of operating system environment variables*
- *List and describe the steps required to create operating system environment variables*
- *State the purpose of the Path environment variable*
- *List and describe the steps required to add the .NET runtime folder path to the Path environment variable*
- *List and describe the steps required to create C# programs using Microsoft's free command-line tools*
- *List and describe the development tools found in the Microsoft Windows Software Development Kit (SDK)*
- *Describe the functions and features generally found in an IDE*
- *List and describe the steps required to create C# programs using Microsoft Visual C# Express*
- *Demonstrate your ability to create C# projects using Microsoft's free command-line tools*
- *Demonstrate your ability to create C# projects using Microsoft's free Visual C# Express*

Introduction

I call this chapter Small Victories because selecting development tools and properly configuring your development environment easily accounts for seventy-five percent of the headaches you'll get when starting down the road of C# .NET software development. Creating, compiling, and running your first project represents the biggest hurdle novice programmers face. The information in this chapter is designed to help you easily jump that hurdle and do a victory dance.

Here you will learn several critical software development skills. First, I explain exactly what you need to write, compile, and run C# programs. The good news is that you don't need a fortune to start programming with C# and the .NET Framework. Microsoft offers powerful and flexible software development tools absolutely free. Next, I will explain the purpose and use of environment variables and show you how to configure the PATH environment variable so you can compile and run C# programs from the command console. And, since you may not be familiar with the command console, I will explain its purpose and use and show you how to use several important commands.

Later, once you understand how to use Microsoft's C# .NET command-line tools, I'll show you how to create programs using Microsoft Visual C# Express. Visual C# Express is Microsoft's free integrated development environment (IDE). An IDE increases programmer productivity by providing, under a common user interface, several important software development tools including source code editing, compiling, debugging, and execution.

Creating Projects With Microsoft C#.NET Command-Line Tools

You need only two things to create professional, robust, C# .NET projects: the Microsoft .NET Framework, and a suitable text editor. Both can be obtained free of charge, although you will most likely want to buy a good text editor.

The .NET Framework supplies you with the C# command-line compiler. The C# command-line compiler is a program that is run in a console window and is used to transform programs written in C# into byte-code modules that can be executed by the .NET Common Language Runtime (a.k.a. the .NET virtual machine).

The question most students ask when I teach them how to use these tools is, "Why?". "Why, if Microsoft offers Visual C# Express (or Visual Studio), do I need to know how to create programs using command-line tools?" That's a good question with several answers, and they go something like this: Visual C# Express is a powerful program. In fact, it's so powerful that you can spend a lot of time just learning what it does and how to use it. So, in order to let students focus on learning the C# language, I recommend they learn how to use the command-line tools and postpone their involvement with Visual C# Express until they get some programming experience.

My second answer to the question has a more practical side. Nowadays, most novice programming students have little or no experience using the command console. They are familiar with the Microsoft Windows interface and pointing and clicking with a mouse, but issuing DOS commands from the command line is something completely alien to them.

I consider the ability to compile programs with the C# command-line tool, issuing DOS commands, and setting and using operating system environment variables to be fundamental skills that all programmers need to have in their tool belt. Also, mastering these fundamental skills will let you better understand what Visual C# Express is doing "under the hood". You may find it necessary one day to dive into the code generated "automagically" by Visual C# Express to make a few adjustments. The only way you'll be able to do that is to take complete control of your development environment and understand how to use the command-line tools to compile and run C# programs.

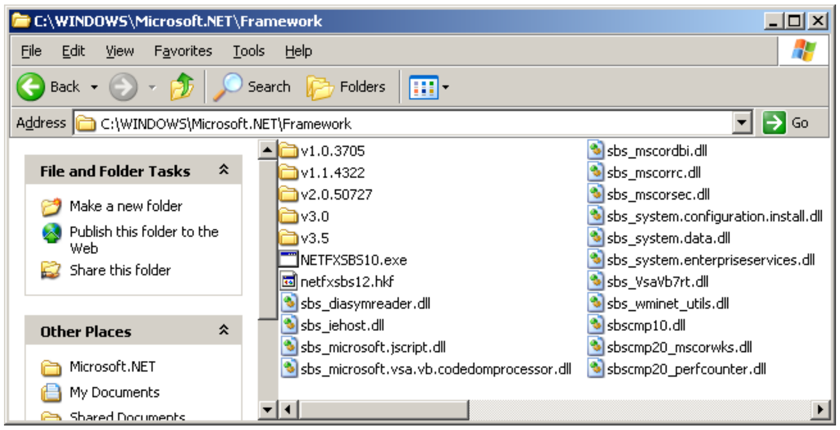
Downloading And Installing The .NET Framework

The first thing you need to do is to download and install the .NET Framework. Microsoft offers the .NET Framework Redistributable Package as a free download from their Microsoft Developer Network website (MSDN) [www.msdn.com]. You can optionally order the .NET Framework on DVD for a nominal charge.

You will find the .NET Framework 3.5 by going to MSDN and under the *Server Development* heading click on *.NET Framework*. On the *.NET Framework* page under the *Get the Framework* heading click on the *.NET Framework 3.5* link. This takes you to the Microsoft .NET Framework 3.5 download page.

At a minimum, you only need to download the .NET Framework 3.5. It provides the .NET runtime environment and the C# command-line compiler. It also includes the .NET Framework 3.0 Service Pack 1, the .NET Framework 3.0 Redistributable Package, and the .NET Framework 2.0 Service Pack 1. You can optionally download and install the Microsoft Windows Software Development Kit (SDK) for .NET Framework 3.5. The Microsoft Windows SDK provides additional development tools. However, before you can install the SDK, you must download and install the .NET Framework.

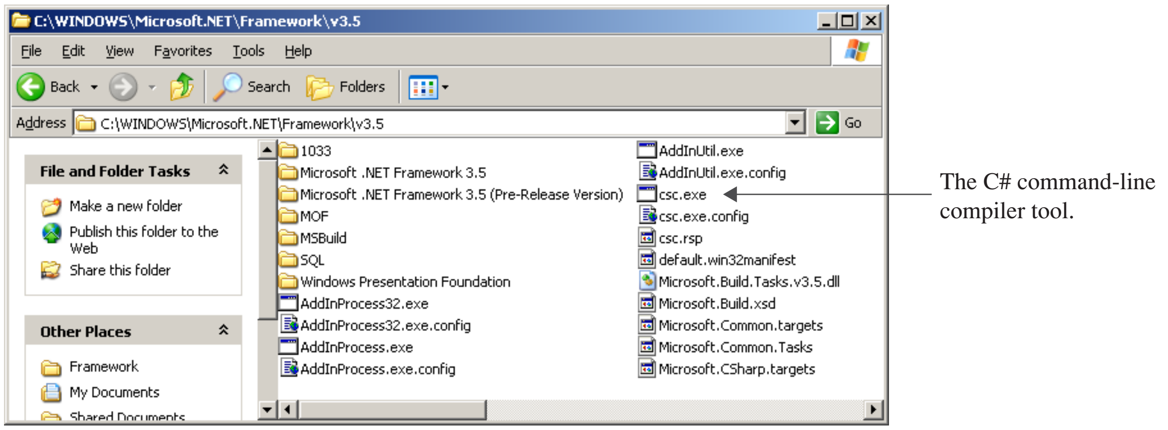
Installation of the .NET Framework is straightforward. The important thing to note during the installation process is where on your hard drive the .NET Framework is installed. The path to the .NET Framework installation directory will be [`c:\Windows\Microsoft.NET\Framework\`]. Figure 2-1 shows the .NET Framework directory structure as it appears on my computer.


C:\WINDOWS\Microsoft.NET\Framework
File Edit View Favorites Tools Help
Back Search Folders
Address C:\WINDOWS\Microsoft.NET\Framework Go
File and Folder Tasks
Make a new folder
Publish this folder to the Web
Share this folder
Other Places
Microsoft.NET
My Documents
Shared Documents

```
v1.0.3705                                   sbs_mscordbi.dll
v1.1.4322                                   sbs_mscorrc.dll
v2.0.50727                                  sbs_mscorsec.dll
v3.0                                        sbs_system.configuration.install.dll
v3.5                                        sbs_system.data.dll
NETFXSBS10.exe                              sbs_system.enterpriseservices.dll
netfxsbs12.hkf                              sbs_VsaVb7rt.dll
sbs_diasymreader.dll                        sbs_wminet_utils.dll
sbs_iehost.dll                              sbscmp10.dll
sbs_microsoft.jscript.dll                   sbscmp20_mscorwks.dll
sbs_microsoft.vsa.vb.codedomprocessor.dll   sbscmp20_perfcounter.dll
```

Figure 2-1: Microsoft.NET Framework Installation Directory

Notice in Figure 2-1 the five folders that begin with the letter ‘v’. These are the five versions of the .NET Framework installed on my computer. For the purpose of this book we’ll only be interested in the most recent version (i.e., the highest numbered folder). Figure 2-2 shows a partial directory listing of the v3.5 folder. In there you’ll find the C# compiler command-line tool.


C:\WINDOWS\Microsoft.NET\Framework\v3.5
File Edit View Favorites Tools Help
Back Search Folders
Address C:\WINDOWS\Microsoft.NET\Framework\v3.5 Go
File and Folder Tasks
Make a new folder
Publish this folder to the Web
Share this folder
Other Places
Framework
My Documents
Shared Documents
The C# command-line compiler tool.

```
1033                                                AddInUtil.exe
Microsoft .NET Framework 3.5                        AddInUtil.exe.config
Microsoft .NET Framework 3.5 (Pre-Release Version)  csc.exe
MOF                                                 csc.exe.config
MSBuild                                             csc.rsp
SQL                                                 default.win32manifest
Windows Presentation Foundation                     Microsoft.Build.Tasks.v3.5.dll
AddInProcess32.exe                                  Microsoft.Build.xsd
AddInProcess32.exe.config                           Microsoft.Common.targets
AddInProcess.exe                                    Microsoft.Common.Tasks
AddInProcess.exe.config                             Microsoft.CSharp.targets
```

Figure 2-2: Partial Directory Listing of the v3.5 Folder

Now that you’ve installed the .NET Framework you have everything you need to compile and run C# programs. What you need now is a way to create C# source files. For that you’ll need a good text editor. This is the topic of the next section.

Downloading And Installing Notepad++

If you really wanted to rough-it you could use Notepad, the text editor that ships with Microsoft Windows. Notepad is perfectly suitable for small source files and lite editing jobs, but for bigger projects, I recommend getting yourself a copy of Notepad++ from SourceForge.net [http://notepad-plus.sourceforge.net/].

Installation of Notepad++ is quick and straightforward. After you complete the installation a shortcut is automatically installed on your desktop. Although Notepad++ is free to download, install, and use, I recommend that if you like the product and use it often, and if you have the means, that you make a small donation via the website to support its further development.

Armed now with the .NET Framework and a suitable text editor, you have everything you need to create C# programs. But before you get started you'll need to properly configure your development environment.

Configuring Your Development Environment

A properly configured development environment is critical to the software creation process. In this section I will show you how to create and use operating system environment variables, how to create a project folder, how to set folder options so you can see filename suffixes, and how to set-up and configure shortcuts to the command console. The skills you learn in this section will prove time and again to be absolutely invaluable.

Environment Variables

An environment variable is a named location in memory used by Microsoft Windows to store data about the operating system environment. There are generally two types of environment variables: *system* and *user*.

System environment variables store data that pertains to and affects the operating system environment for all users. User environment variables store data that pertains to and affects the operating system environment for a particular user. Some system and user environment variables are automatically created and initialized by the operating system when it is installed and when users are created.

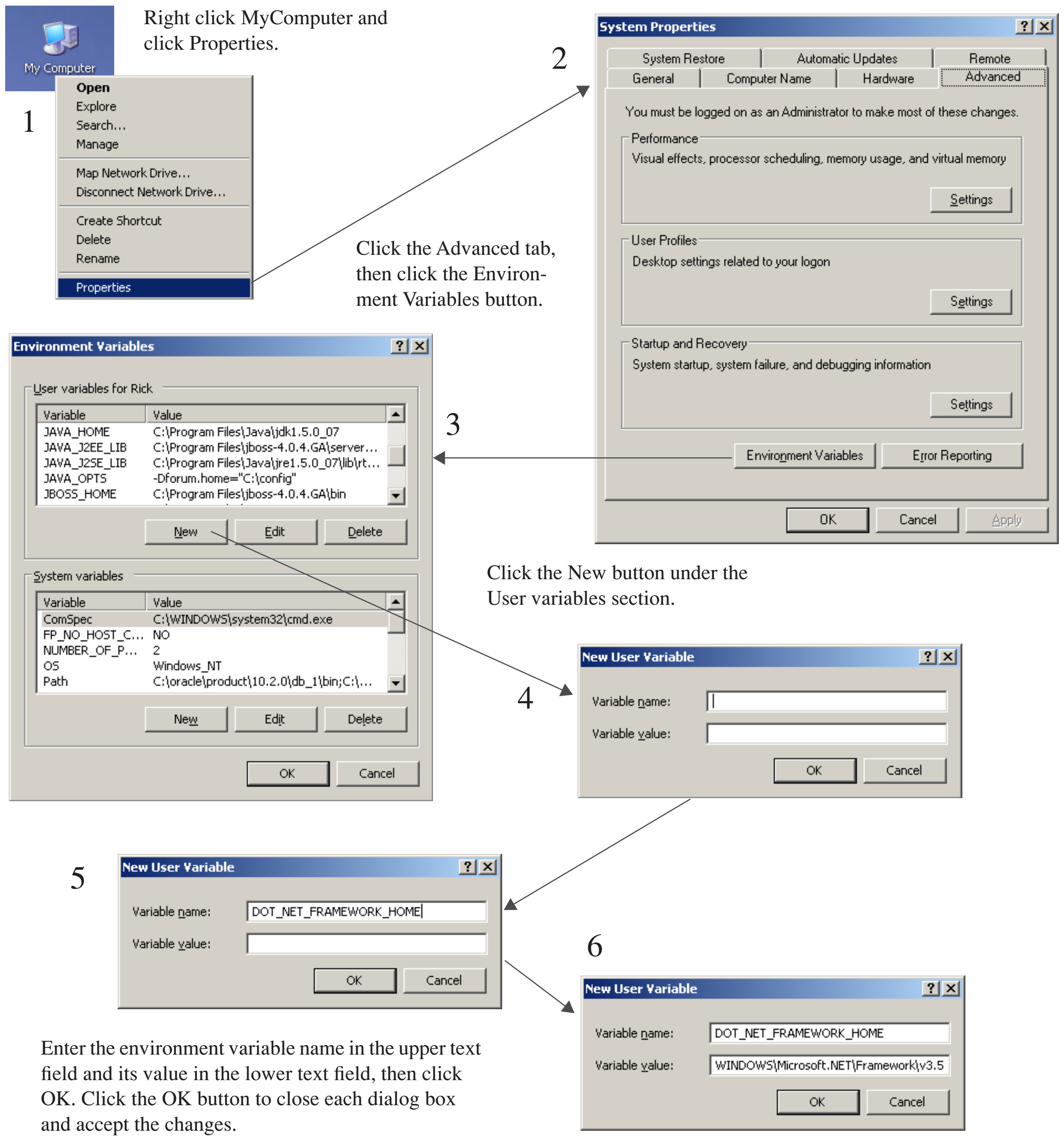
Several important environment variables must be created or edited before you can use the command-line tools to compile and run C# programs. These include: 1) a variable named DOT_NET_FRAMEWORK_HOME that contains the path to the folder location of the most current .NET Framework, and 2) the PATH variable that includes a reference to the DOT_NET_FRAMEWORK_HOME variable so the operating system will know where to find .NET-related executable files like the C# command-line compiler.

Creating Environment Variables

The first environment variable you will set will be the location of the home directory of the .NET Framework. Navigate to that folder now so that you can copy the path to the .NET Framework directory; later, you can paste this value into the environment variable value's text field. (*Doing this prevents you from making mistakes when typing long path names*) The path to this folder should be [`c:\Windows\Microsoft.NET\Framework\v3.5`]. (*Refer to Figure 2-2*) When you open the folder, select the path that shows in the Address box and copy it using CTRL-C.

Next, we'll create the user environment variable named DOT_NET_FRAMEWORK_HOME. See Figure 2-3 for an illustration of the complete environment variable creation process.

Right click the My Computer icon located on your desktop. If this icon is not located on your desktop, click the Start icon located in the lower left part of your screen along the toolbar. Find My Computer and right click it. Click Properties to open the System Properties dialog window. In the System Properties dialog click the Advanced tab, then click the Environment Variables button to open the Environment Variables dialog window. Underneath the User variables section, click the New button to create a new environment variable. This will open the New User Variable dialog window. Enter DOT_NET_FRAMEWORK_HOME into the Variable name textbox. Paste the path to the .NET Framework home directory you copied earlier into the Variable value textbox. After entering both values, your New User Variable dialog window will look similar to the completed example shown in Figure 2-3. Check your work for accuracy, then click the OK button to close the New User Variable dialog window. Click the OK button for each of the remaining open dialog windows to accept the changes. Congratulations! You just created an environment variable.



Right click MyComputer and click Properties.

Click the Advanced tab, then click the Environment Variables button.

Click the New button under the User variables section.

Enter the environment variable name in the upper text field and its value in the lower text field, then click OK. Click the OK button to close each dialog box and accept the changes.

You must be logged on as an Administrator to make most of these changes.

Performance: Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles: Desktop settings related to your logon

Startup and Recovery: System startup, system failure, and debugging information

User variables for Rick

Variable	Value
JAVA_HOME	C:\Program Files\Java\jdk1.5.0_07
JAVA_J2EE_LIB	C:\Program Files\jboss-4.0.4.GA\server...
JAVA_J2SE_LIB	C:\Program Files\Java\jre1.5.0_07\lib\rt...
JAVA_OPTS	-Dforum.home="C:\config"
JBOSS_HOME	C:\Program Files\jboss-4.0.4.GA\bin

System variables

Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	2
OS	Windows_NT
Path	C:\oracle\product\10.2.0\db_1\bin;C:\...

Variable name: DOT_NET_FRAMEWORK_HOME

Variable value: WINDOWS\Microsoft.NET\Framework\v3.5

Figure 2-3: Creating an Environment Variable

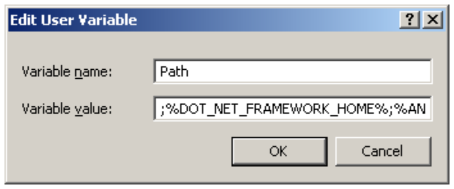
Creating or Editing the Path Environment Variable

Now that you have created the DOT_NET_FRAMEWORK_HOME environment variable, you can use it to create or edit other environment variables. The next environment variable that you must either create or edit is the Path variable. The operating system uses the Path environment variable to locate executable files. An instance of the Path variable most likely already exists in the System Environment Variables section. I recommend leaving that version alone and creating another Path environment variable in the User Environment Variables section. To create a new User Path environment variable, follow the process illustrated in Figure 2-3. Enter “Path” into the Variable name text field. Enter the following into the Variable value text field: %DOT_NET_FRAMEWORK_HOME%. Click the OK buttons to accept the changes. (**Note:** To access an environment variable's value, add a "%" to the beginning and end of the variable name.)

If a Path environment variable already exists, you'll need to select it and click the Edit button. Place your cursor in the Variable value text field and move to the far end of the value that's entered in the text field. If that value is not terminated with a semicolon, you'll need to add one before adding the following:

%DOT_NET_FRAMEWORK_HOME%;

Figure 2-4 shows the Path user environment variable being edited on my machine.



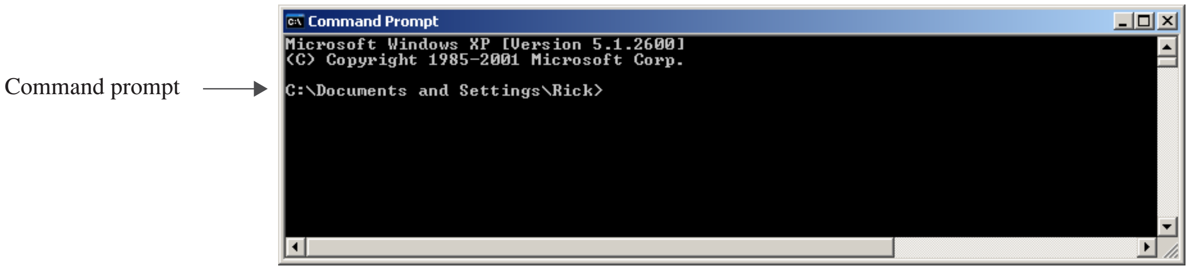
Variable name: Path
Variable value: ;%DOT_NET_FRAMEWORK_HOME%;%AN

Figure 2-4: Editing the Path User Environment Variable

Testing The Newly Created Environment Variables

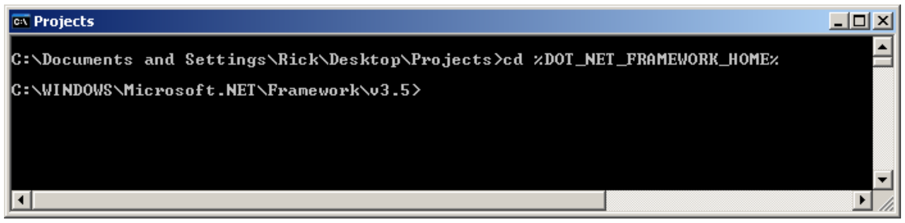
Check that you have set your environment variables correctly by running a couple of tests. The first test entails opening a command console and using the DOT_NET_FRAMEWORK_HOME variable in a command. The second test will be running the C# command-line compiler.

First, open the command console. Do this by clicking on Start->All Programs->Accessories->Command Prompt. This will open a command console window like that shown in Figure 2-5. Next, enter the following command at the command prompt: `cd %DOT_NET_FRAMEWORK_HOME%` The `cd` command stands for "Change Directory". If you have set the DOT_NET_FRAMEWORK_HOME environment variable correctly, entering this command should open the .NET Framework v3.5 directory as Figure 2-6 illustrates.



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Rick>
```

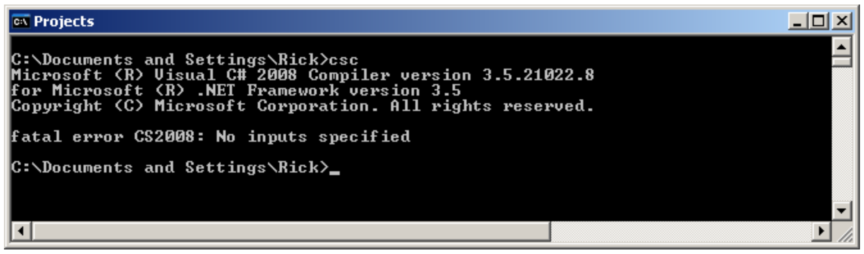
Figure 2-5: Command Console Window



```
C:\Documents and Settings\Rick\Desktop\Projects>cd %DOT_NET_FRAMEWORK_HOME%

C:\WINDOWS\Microsoft.NET\Framework\v3.5>
```

Figure 2-6: Testing the DOT_NET_FRAMEWORK_HOME Environment Variable

Now test the Path environment variable. Execute the following command in a command console window: `csc` This should run the C# compiler which will produce a result similar to that shown in Figure 2-7. If your results look like those shown in Figure 2-7, then all is set properly. Great job! If not, recheck your environment variable settings and try again until you have things set just right.



```
C:\Documents and Settings\Rick>csc
Microsoft (R) Visual C# 2008 Compiler version 3.5.21022.8
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

fatal error CS2008: No inputs specified

C:\Documents and Settings\Rick>_
```

Figure 2-7: Testing the Path Environment Variable by Running the C# Compiler

Creating A Project Folder

The next thing you must do is create a folder in which to store your C# project files. Create a folder named Projects on your C drive or some other acceptable location. This folder will serve as the root folder for any individual projects you create later. You will store each project in its own folder under the Projects folder.

A good place to create the Projects folder is right on your desktop. To do this, right-click your desktop and select New->Folder from the right-click menu as Figure 2-8 illustrates.

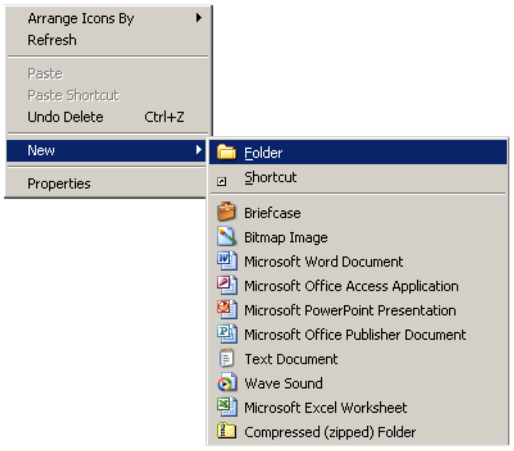

Arrange Icons By
Refresh
Paste
Paste Shortcut
Undo Delete Ctrl+Z
New
Properties
Folder
Shortcut
Briefcase
Bitmap Image
Microsoft Word Document
Microsoft Office Access Application
Microsoft PowerPoint Presentation
Microsoft Office Publisher Document
Text Document
Wave Sound
Microsoft Excel Worksheet
Compressed (zipped) Folder

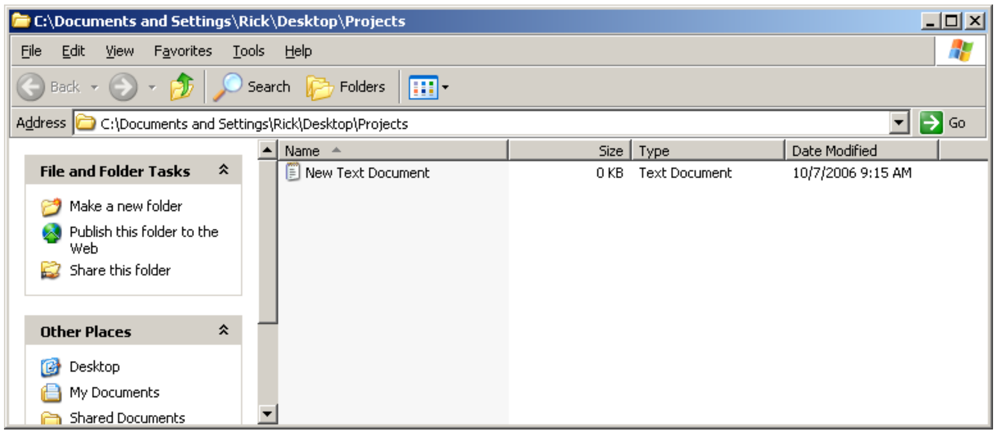
Figure 2-8: Creating a New Folder

Name the new folder “Projects”. Note that when you create a folder on your desktop, you are actually creating it in the [`C:\Documents and Settings\username\Desktop`] folder, where “username” is the username of the account you used to log on to the computer. On my machine, the full path to the Projects folder created on the desktop is this: [`C:\Documents and Settings\rick\Desktop\Projects`]

Setting Folder Options

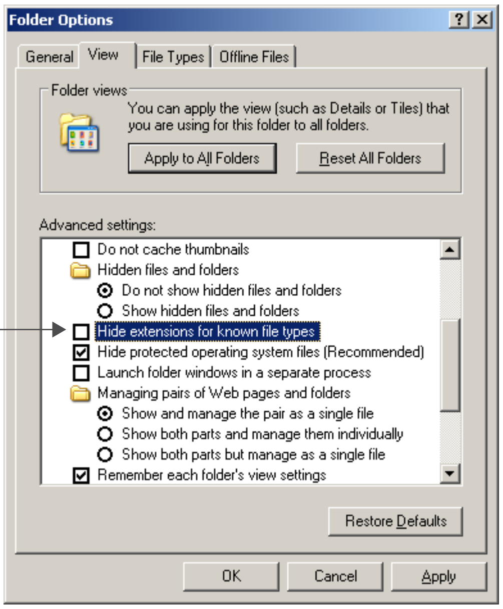
Now that you have created your Projects folder, you’ll need to change its folder options so you can see file-type extensions. Novice and experienced programmers sometimes have difficulty trying to compile a C# file because the file they thought was saved with an extension of “.cs” was in fact saved with an extension of “.cs.txt”, where the “.txt” extension was automatically added by a text editor, unbeknownst by the programmer. When this happens, the C# compiler will fail to recognize the file as a C# source file. To help prevent such headaches, it’s a great idea to change the folder options of all your folders to show file extensions.

To do this, open the Projects folder and in it create a new text file. The easiest way to create the text file is to simply right click in the open folder and select New->Text Document from the right-click menu. Save the text document with the default name provided. Your Projects folder should now look like Figure 2-9. Notice that the name of the document you just created simply shows as “New Text Document”. The “.txt” extension is hidden by default. So let’s unhide the file extensions. Click the Tools->Folder Options... menu to bring up the Folder Options dialog window. Click the View tab and scroll down until you see the check box that says, “Hide extensions for known file types”. This box is checked by default. Uncheck the box as is shown in Figure 2-10.


C:\Documents and Settings\Rick\Desktop\Projects
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Documents and Settings\Rick\Desktop\Projects Go
File and Folder Tasks
Make a new folder
Publish this folder to the Web
Share this folder
Other Places
Desktop
My Documents
Shared Documents

Name	Size	Type	Date Modified
New Text Document	0 KB	Text Document	10/7/2006 9:15 AM

Figure 2-9: Projects Folder Before Setting Folder Options



You can apply the view (such as Details or Tiles) that you are using for this folder to all folders.

Advanced settings:

- [ ] Do not cache thumbnails
- Hidden files and folders
  - (●) Do not show hidden files and folders
  - ( ) Show hidden files and folders
- [ ] Hide extensions for known file types
- [✓] Hide protected operating system files (Recommended)
- [ ] Launch folder windows in a separate process
- Managing pairs of Web pages and folders
  - (●) Show and manage the pair as a single file
  - ( ) Show both parts and manage them individually
  - ( ) Show both parts but manage as a single file
- [✓] Remember each folder's view settings

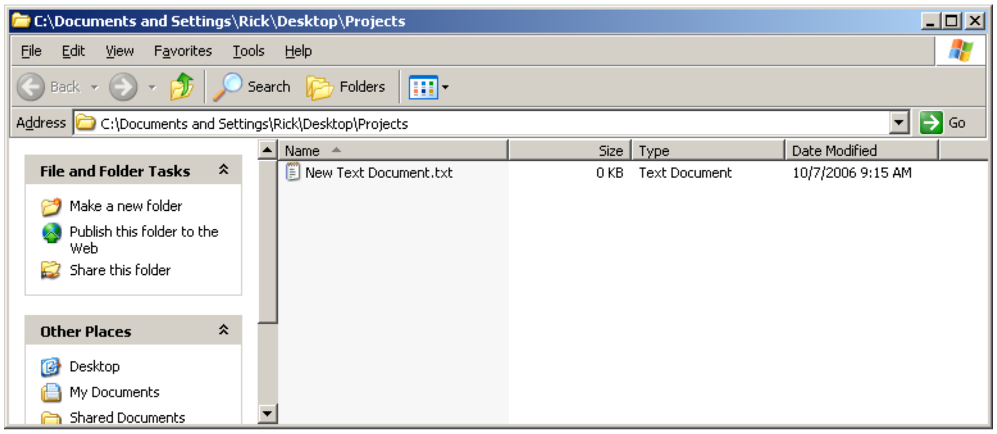
Uncheck this box to show file extensions.

Figure 2-10: Folder Options Dialog Window

Click the Apply to All Folders button in the Folder views section, then click the Apply button and lastly the OK button to dismiss the dialog. Your Projects folder should now look like Figure 2-11. Notice now you can see the ".txt" file extension.

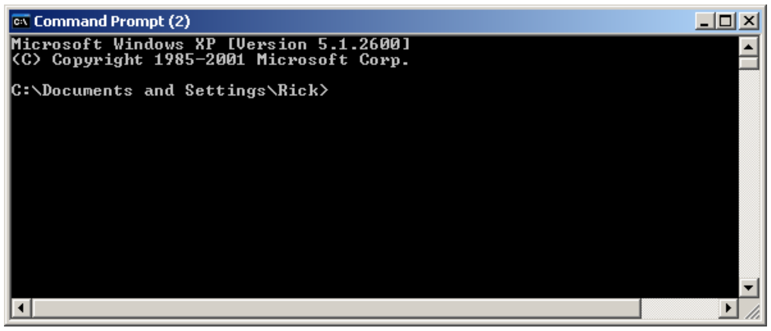
Creating A Shortcut To The Command Console And Setting Its Properties

Since you'll be using the command console to compile C# programs you'll find it convenient to place a command console shortcut on your desktop. To do this, click Start->All Programs->Accessories. Navigate to Command Prompt and right click it. Select Create Shortcut. This will create a new item in the Accessories menu named "Command Prompt (2)". Select the Command Prompt (2) icon and drag it to your desktop. Test the shortcut by double clicking it to open the command console. By default, it should open to the directory [`C:\Documents and Settings\username`], where "username" is the account you used to log on to the computer. Figure 2-12 shows how the command console window looks with its default settings on my machine.


C:\Documents and Settings\Rick\Desktop\Projects
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Documents and Settings\Rick\Desktop\Projects Go
File and Folder Tasks
Make a new folder
Publish this folder to the Web
Share this folder
Other Places
Desktop
My Documents
Shared Documents

Name	Size	Type	Date Modified
New Text Document.txt	0 KB	Text Document	10/7/2006 9:15 AM

Figure 2-11: Projects Folder After Setting Folder Options



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Rick>
```

Figure 2-12: Default Command Console Window

Change The Name Of The Command Console Shortcut

The first bit of command console shortcut customization you'll want to do is to change its name. Do this by clicking twice on the shortcut icon's name, pausing between clicks longer than a standard double-click. If you click too fast, you'll simply open the console window. If this happens simply close the window and try again. Rename the shortcut to anything you want, but I recommend changing it to "Projects", or "C# Projects".

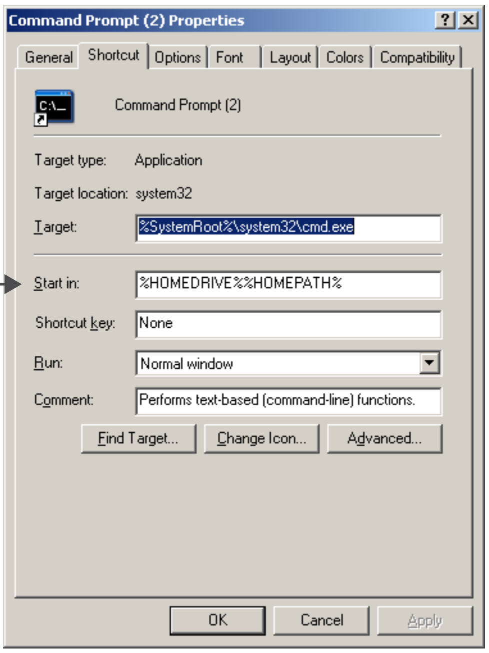
Change The Startup Folder Settings

Now that you've changed the command shortcut's name, let's make a more meaningful change. It would be nice if the console window opened automatically in the Projects directory. To make this happen, right click the command console shortcut icon located on your desktop and select Properties. This opens the properties dialog window for that shortcut. For example, if you renamed your shortcut to "Projects", the name of the properties dialog will be "Projects Properties". If you left the name of the shortcut with its default value, the name will be "Command Prompt (2) Properties" as Figure 2-13 illustrates. **Note:** The Shortcut tab is selected by default.

To make the command console automatically start in the Projects directory, change the **Start in** property by replacing its default contents with the full path to the Projects folder. This will be `[C:\Documents and Settings\username\Desktop\Projects\]`, where "username" is the account name you used to log on to the computer. Figure 2-14 shows the command console **Start in** property after I set it on my machine.

Click the OK button to accept the changes. Test the configuration by double clicking the command console shortcut icon. It should open either in the Projects folder, or the folder you designated. If not, recheck your settings and try again until everything works as expected.

You're going to change the **Start in** property.



Command Prompt (2)

Target type: Application

Target location: system32

Target: %SystemRoot%\system32\cmd.exe

Start in: %HOMEDRIVE%%HOMEPATH%

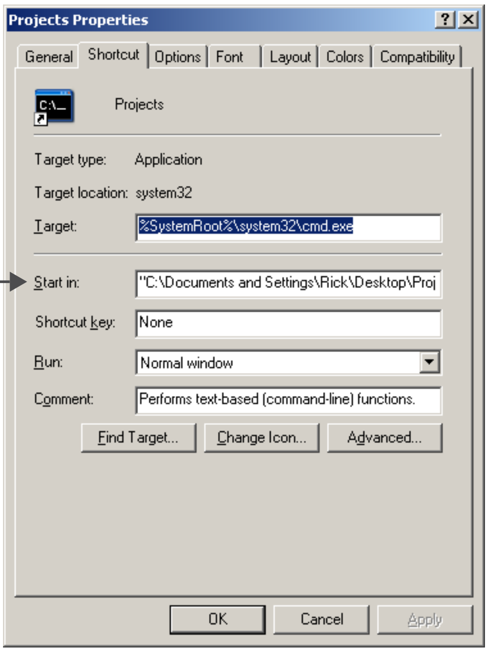
Shortcut key: None

Run: Normal window

Comment: Performs text-based (command-line) functions.

Figure 2-13: Command Console Properties Dialog

Start in property set to the path of your Projects folder. Notice the path is enclosed in quotes.



Projects

Target type: Application

Target location: system32

Target: %SystemRoot%\system32\cmd.exe

Start in: "C:\Documents and Settings\Rick\Desktop\Proj

Shortcut key: None

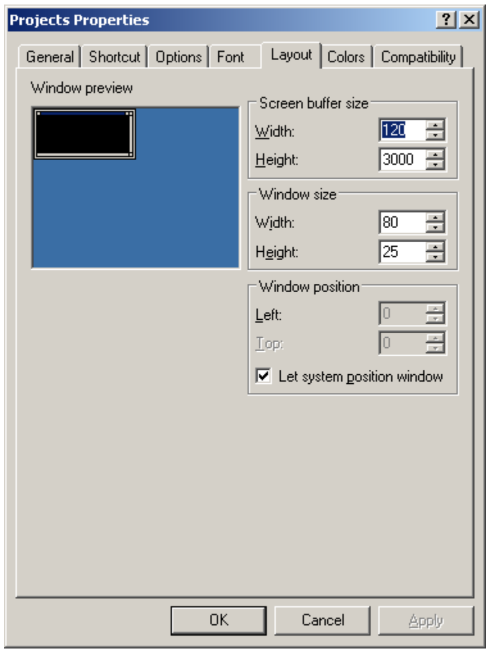
Run: Normal window

Comment: Performs text-based (command-line) functions.

Figure 2-14: Setting the **Start in** Property

Change The Layout Properties

The last adjustment left to make to the command console shortcut is to change its default screen buffer size. This will allow you to increase the length and width of the screen to see more information without the lines wrapping. Once again, right click the command console icon and click the Properties item to open the Properties dialog window. Click the Layout tab and set the screen buffer size **Width** property to 120, and change the **Height** property to 3000, as Figure 2-15 illustrates.



Screen buffer size
Width: 120
Height: 3000

Window size
Width: 80
Height: 25

Window position
Left: 0
Top: 0
☑ Let system position window

Change the Width and Height properties to see more information.

Figure 2-15: Setting Command Console Layout Properties

Click the OK button to accept the new changes. Now, double click the command console shortcut to open the console window. Change its height and width by dragging the lower right-hand corner. You'll find this to be a big help when troubleshooting and debugging your programs.

Testing The Configuration

As a final test of the configuration, let's create and run a short C# program. To do this, you'll need to create the C# source file with the text editor, compile the source file using the C# command-line compiler, and then run the program.

Creating The Source File

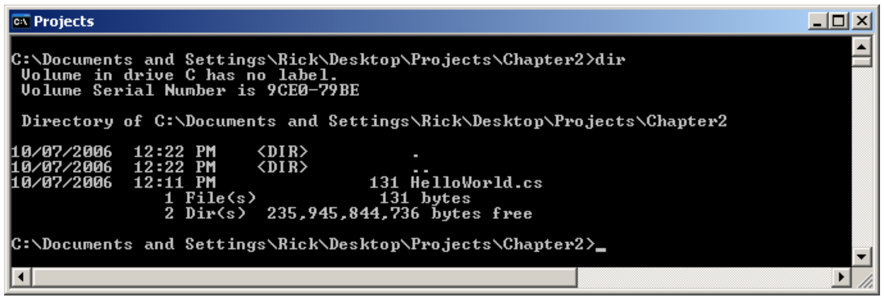
Using either Notepad++ or another text editor create a new file named "HelloWorld.cs" and save it in your Projects folder. Enter the code shown in Example 2.1 into your source file and save the file.

2.1 HelloWorld.cs

```
using System;

public class HelloWorld {

  public static void Main(){

     Console.WriteLine("Hello World!");
   }
}
```

Compiling The Source File

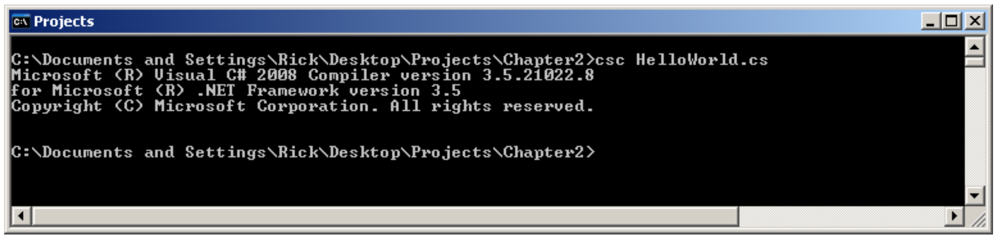
To compile the HelloWorld.cs file, open the command console and change to the directory where you saved the file. If you saved it in the Projects folder, then you're already there. If you created a sub folder then change to that directory by using the `cd` command. For example, I saved the file in a folder named "Chapter2" located in the Projects folder. To change to the Chapter2 directory from the Projects directory I entered `cd chapter2`, then pressed the Return or Enter key. Figure 2-16 shows how the console looks on my machine when I use the `dir` command to list the directory contents.



```
C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>dir
 Volume in drive C has no label.
 Volume Serial Number is 9CE0-79BE

 Directory of C:\Documents and Settings\Rick\Desktop\Projects\Chapter2

10/07/2006  12:22 PM    <DIR>          .
10/07/2006  12:22 PM    <DIR>          ..
10/07/2006  12:11 PM               131 HelloWorld.cs
               1 File(s)            131 bytes
               2 Dir(s)  235,945,844,736 bytes free

C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>_
```

Figure 2-16: Directory Listing of the Chapter2 Directory Showing the HelloWorld.cs File

To compile the HelloWorld.cs file, enter `csc` followed by the name of the source file at the command prompt. If you entered the source code correctly, you should see results similar to those shown in Figure 2-17.



```
C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>csc HelloWorld.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.21022.8
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.


C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>
```

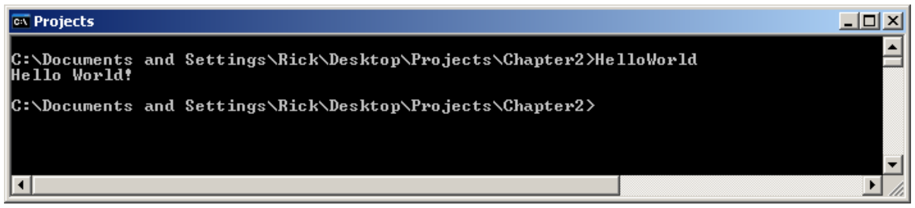
Figure 2-17: Compiling HelloWorld.cs Using the `csc` C# Compiler Command

If you execute another `dir` command to display the directory contents, you'll see a new file named "HelloWorld.exe". This is the executable program file.

Executing The Application

To run the executable file, simply enter its name at the command prompt. Figure 2-18 shows the results of running the Hello World program.

Program output:



```
C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>HelloWorld
Hello World!

C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>
```

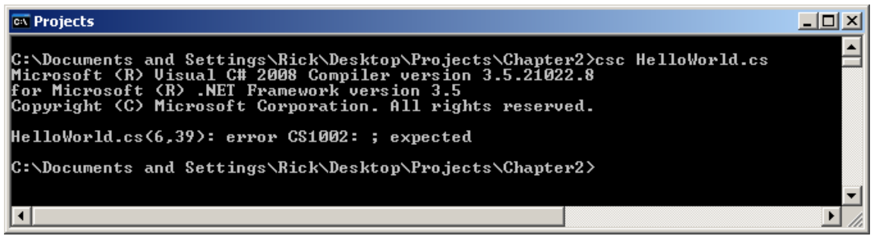
Figure 2-18: Running the HelloWorld Program

Fixing Compilation Errors

No matter how careful you try to be, you're bound to make a mistake or two (or more) when writing programs. Most of these mistakes will be simple typos like forgetting to terminate a statement with a semicolon. When you compile a program that contains a compiler error, you will see something similar to the output shown in Figure 2-19.

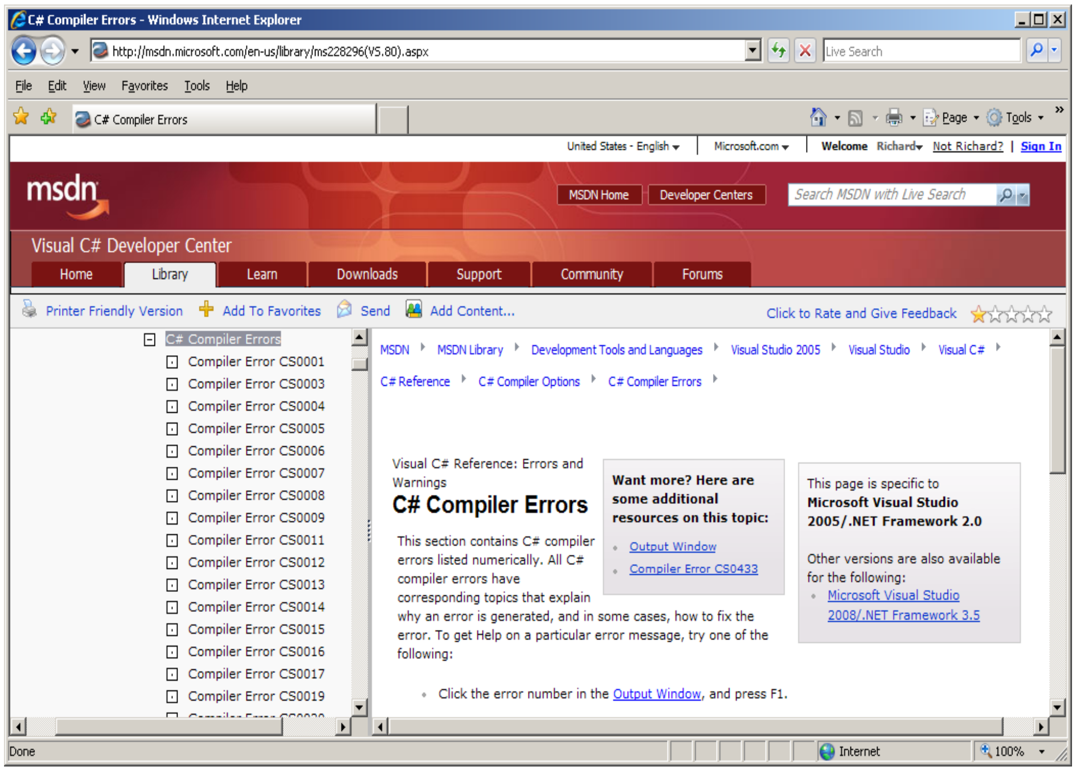
When the compiler encounters a problem it will output one or more warning or error messages. Warnings are usually non-fatal in that your program will still run if the compiler signals only a warning message. Error messages, on the other hand, must be addressed before your program will compile completely.

The error message will contain the name of the source file, the line number and character position of the problem, and the compiler error code. The C# compiler error codes can be found on the Microsoft C# language reference site, but searching for them on Microsoft's website does you little good. The best way to find detailed information about a particular C# compiler error is to enter the following search query into Google: "C# compiler error CSNNNN",



```
C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>csc HelloWorld.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.21022.8
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

HelloWorld.cs(6,39): error CS1002: ; expected

C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>
```

Figure 2-19: Compiler Output Showing Compiler Error on Line 6 at Position 39



C# Compiler Errors - Windows Internet Explorer
http://msdn.microsoft.com/en-us/library/ms228296(VS.80).aspx
Live Search
File Edit View Favorites Tools Help
C# Compiler Errors
Page Tools
United States - English Microsoft.com Welcome Richard Not Richard? | Sign In
msdn
MSDN Home Developer Centers
Search MSDN with Live Search
Visual C# Developer Center
Home Library Learn Downloads Support Community Forums
Printer Friendly Version Add To Favorites Send Add Content...
Click to Rate and Give Feedback
C# Compiler Errors
Compiler Error CS0001
Compiler Error CS0003
Compiler Error CS0004
Compiler Error CS0005
Compiler Error CS0006
Compiler Error CS0007
Compiler Error CS0008
Compiler Error CS0009
Compiler Error CS0011
Compiler Error CS0012
Compiler Error CS0013
Compiler Error CS0014
Compiler Error CS0015
Compiler Error CS0016
Compiler Error CS0017
Compiler Error CS0019
MSDN MSDN Library Development Tools and Languages Visual Studio 2005 Visual Studio Visual C#
C# Reference C# Compiler Options C# Compiler Errors
Done
Internet
100%

Visual C# Reference: Errors and Warnings

C# Compiler Errors

This section contains C# compiler errors listed numerically. All C# compiler errors have corresponding topics that explain why an error is generated, and in some cases, how to fix the error. To get Help on a particular error message, try one of the following:

- Click the error number in the Output Window, and press F1.

Want more? Here are some additional resources on this topic:

- Output Window
- Compiler Error CS0433

This page is specific to Microsoft Visual Studio 2005/.NET Framework 2.0

Other versions are also available for the following:

- Microsoft Visual Studio 2008/.NET Framework 3.5

Figure 2-20: C# Language Compiler Errors

where “NNNN” is the compiler error number. The first result from this query will usually lead straight to the Microsoft C# compiler error page for that compiler error number. As you can see from Figure 2-20, the C# compiler’s error message information is located in the Visual C# language reference area. The detailed information for compiler error number CS1002 is shown in Figure 2-21.

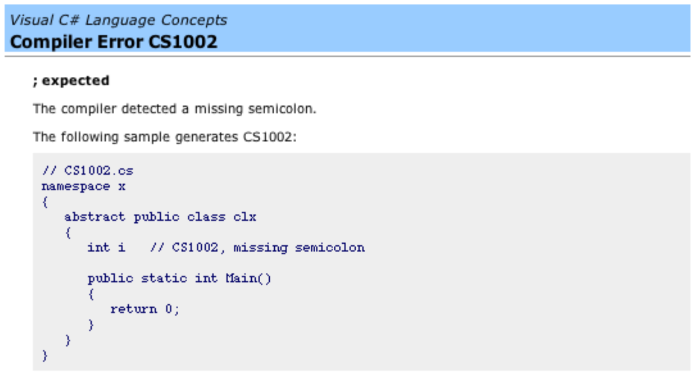
Fix The First Compiler Error First

The best advice I can offer when dealing with compiler errors is to always **fix the first compiler error first**. The reason for this is that some compiler errors trigger other errors. Fixing the first error generally eliminates many other errors on the list.

Quick Review

All you need to create robust Microsoft C# applications is a good text editor and the free Microsoft C# command-line compiler that’s included with the .NET Framework Redistributable Package.

You must configure your development environment before using the C# command-line compiler. This includes creating or editing one or more operating system environment variables. An environment variable is a named location



Visual C# Language Concepts
Compiler Error CS1002

; expected

The compiler detected a missing semicolon.

The following sample generates CS1002:

```
// CS1002.cs
namespace x
{
   abstract public class clx
   {
      int i   // CS1002, missing semicolon

      public static int Main()
      {
         return 0;
      }
   }
}
```

Figure 2-21: C# Compiler Error CS1002 "; expected"

in memory used by Microsoft Windows to store data about the operating system environment. There are generally two types of environment variables: *system* and *user*. System environment variables store data that pertains to and affects the operating system environment for all users; user environment variables store data that pertains to and affects the operating system environment for a particular user.

Environment variable values can be accessed by enclosing the variable name in '%' characters.

The operating system uses the Path environment variable to help it find executable files. You must create or edit the Path environment variable to include the full path to the C# compiler (csc.exe).

It's helpful to create a project folder and a shortcut to the command console on your desktop. Set the command console shortcut's **Start in** property so it will automatically open in your designated project folder. Increase the command console shortcut's screen buffer **height** and **width** properties to see more information in the console window.

It's also a good idea to set your folder options to display file type extensions. This will prevent headaches associated with accidently saving source files with a ".txt" extension.

To create a C# program with the command-line compiler, you must create the source file, compile the source file with the `csc` command-line compiler tool, and then execute the program by typing its name at the command prompt and pressing the Return or Enter key.

You're bound to get a few compiler errors when you start writing your own programs. Go to Microsoft's website to look up the error code. Remember to always **fix the first compiler error first**!

Creating Projects With Microsoft Visual C# Express

Microsoft Visual C# Express Edition is an IDE that combines text editing, debugging, project management, and a host of other features. Visual C# Express is a trimmed-down, lightweight version of Microsoft's flagship development environment Visual Studio. With Visual C# Express, you can create C# console and Windows forms applications. Visual C# Express also comes bundled with Microsoft SQL Server Compact Edition. This allows you to create applications that access and store data to a relational database.

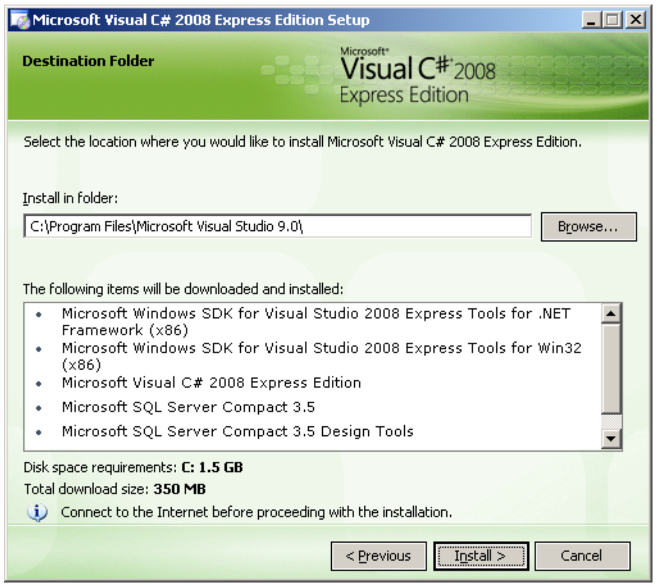
The convenience and power of Microsoft's Visual Studio products come at a price. Although conceptually they are "easy" to learn, in reality, their multitude of features do present a significant learning curve to those who are absolutely new to programming. The benefit Visual Studio brings to the programmer is its seamless integration of Microsoft's powerful arsenal of developer tools. Your productivity will exponentially increase when you do make the move from command-line tools to the Visual Studio environment — that is, after you've figured out how to properly use the tool.

Download and Install Visual C# Express

I'll make the assumption in this section that you have a high-speed internet connection. If not, I recommend you order the Visual C# Express DVD from Microsoft.

The download and installation of Visual C# Express is straightforward. Open a web browser and go to MSDN [www.msdn.com]. Under the *Developer Tools and Languages* heading click on Visual C#. This will take you to the Visual C# Developer Center home page. Under the *Get Visual Studio* heading click the Free Download: Visual C# 2008 Express link. Click the Download Now! link and select Visual C# 2008 Express Edition.

When you start the install, you'll eventually be presented with the installation screen shown in Figure 2-22.


Microsoft Visual C# 2008 Express Edition Setup
Destination Folder
Microsoft Visual C# 2008 Express Edition
Browse...
< Previous
Install >
Cancel

Select the location where you would like to install Microsoft Visual C# 2008 Express Edition.

Install in folder:

C:\Program Files\Microsoft Visual Studio 9.0\

The following items will be downloaded and installed:

- Microsoft Windows SDK for Visual Studio 2008 Express Tools for .NET Framework (x86)
- Microsoft Windows SDK for Visual Studio 2008 Express Tools for Win32 (x86)
- Microsoft Visual C# 2008 Express Edition
- Microsoft SQL Server Compact 3.5
- Microsoft SQL Server Compact 3.5 Design Tools

Disk space requirements: **C: 1.5 GB**
Total download size: **350 MB**
Connect to the Internet before proceeding with the installation.

Figure 2-22: Visual C# Express Installation Window

Note that the installation also includes Microsoft SQL Server Compact 3.5 and its associated design tools. After installation is complete you will need to restart your computer. Having done this, you should be able to start Visual C# by selecting Start->All Programs->Microsoft Visual C# 2008 Express Edition. You will get a screen that looks similar to Figure 2-23.

Quick Tour Of Visual C# Express

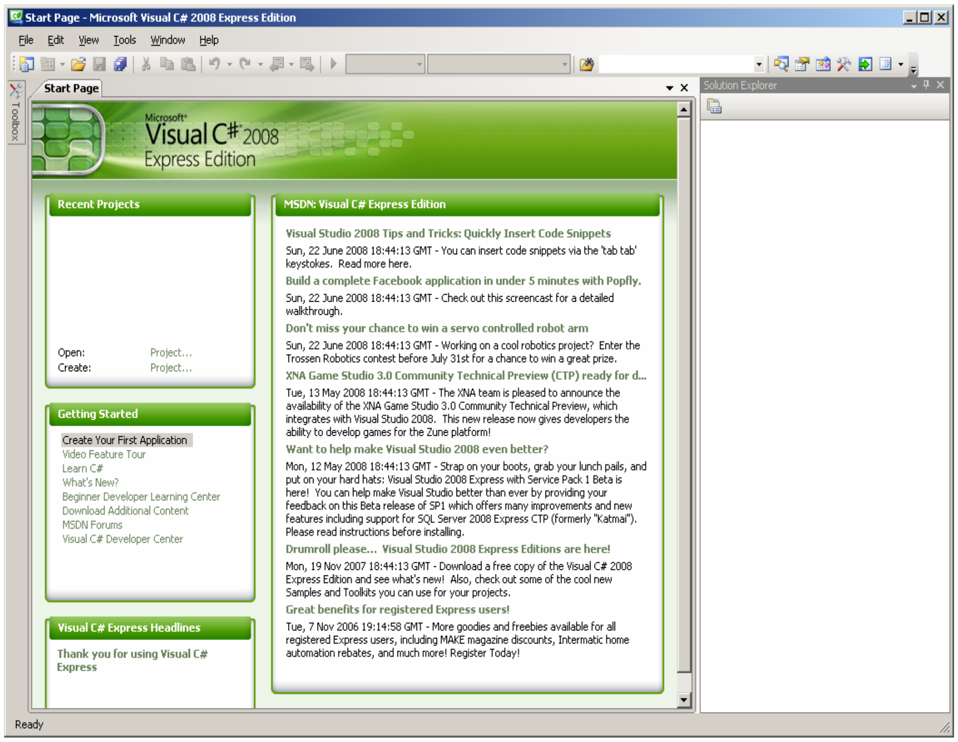
The best way to get a feel for how Visual C# Express works is to create the HelloWorld project. If you haven't already done so, launch Visual Studio Express.

Select Project Type

Start by selecting File->New Project from the main menu. This opens the New Project dialog window shown in Figure 2-24. Click the OK button to create a project. Your Visual C# Express window will now look similar to Figure 2-25. Notice in Figure 2-25 that Visual C# Express automatically generated a lot of source code when it created the new project. But, if you look closely, it added a few more lines of code than what I provided in Example 2.1, and also left one out. Additionally, the name of the class is not HelloWorld. It's "Program"! Also, it did not add the keyword `public` to the beginning of the class declaration, and it used a different version of the `Main()` method. There is also a critical piece of code missing, namely, the `Console.WriteLine("Hello World!");` statement that goes in the `Main()` method. Let's make a few changes to the automatically generated code and then run the project.

First, in the Solution Explorer -HelloWorld pane, find the Program.cs file and right-click it to rename it to "HelloWord.cs". When you do this all references to Program will be automatically updated to HelloWorld. Next, in front of the words `class` and `static` add the keyword `public`. Finally, add the statement `Console.WriteLine("Hello World!");` between the opening and closing brackets of the `Main()` method.

Notice that as you type, Visual C# Express tries to lend a hand with its IntelliSense technology, as is shown in Figure 2-26. Referring to Figure 2-26, IntelliSense is offering a list of the Console object's available public methods


Start Page - Microsoft Visual C# 2008 Express Edition
File Edit View Tools Window Help
Start Page
Solution Explorer
Toolbox
Microsoft Visual C# 2008 Express Edition
Recent Projects
Open: Project...
Create: Project...
Getting Started
Create Your First Application
Video Feature Tour
Learn C#
What's New?
Beginner Developer Learning Center
Download Additional Content
MSDN Forums
Visual C# Developer Center
Visual C# Express Headlines
Thank you for using Visual C# Express
MSDN: Visual C# Express Edition
Ready

Visual Studio 2008 Tips and Tricks: Quickly Insert Code Snippets
Sun, 22 June 2008 18:44:13 GMT - You can insert code snippets via the 'tab tab' keystokes. Read more here.

Build a complete Facebook application in under 5 minutes with Popfly.
Sun, 22 June 2008 18:44:13 GMT - Check out this screencast for a detailed walkthrough.

Don't miss your chance to win a servo controlled robot arm
Sun, 22 June 2008 18:44:13 GMT - Working on a cool robotics project? Enter the Trossen Robotics contest before July 31st for a chance to win a great prize.

XNA Game Studio 3.0 Community Technical Preview (CTP) ready for d...
Tue, 13 May 2008 18:44:13 GMT - The XNA team is pleased to announce the availability of the XNA Game Studio 3.0 Community Technical Preview, which integrates with Visual Studio 2008. This new release now gives developers the ability to develop games for the Zune platform!

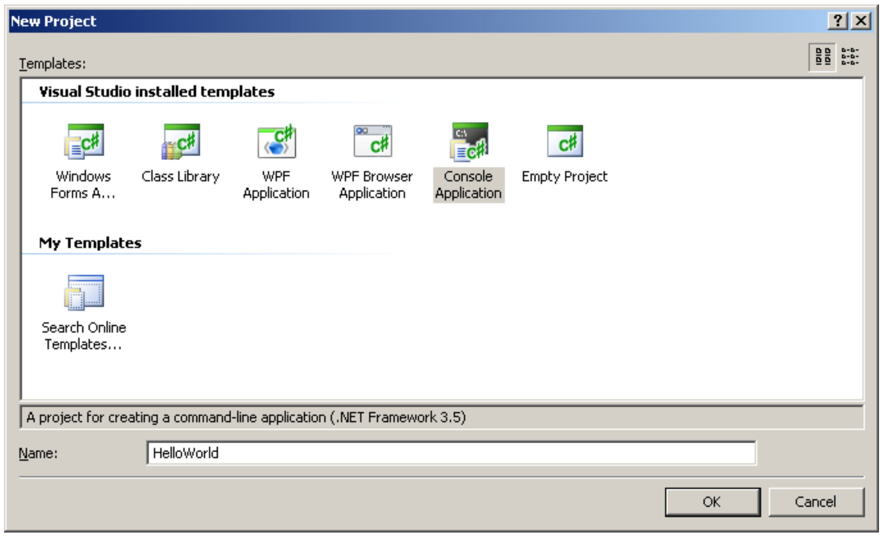
Want to help make Visual Studio 2008 even better?
Mon, 12 May 2008 18:44:13 GMT - Strap on your boots, grab your lunch pails, and put on your hard hats: Visual Studio 2008 Express with Service Pack 1 Beta is here! You can help make Visual Studio better than ever by providing your feedback on this Beta release of SP1 which offers many improvements and new features including support for SQL Server 2008 Express CTP (formerly "Katmai"). Please read instructions before installing.

Drumroll please... Visual Studio 2008 Express Editions are here!
Mon, 19 Nov 2007 18:44:13 GMT - Download a free copy of the Visual C# 2008 Express Edition and see what's new! Also, check out some of the cool new Samples and Toolkits you can use for your projects.

Great benefits for registered Express users!
Tue, 7 Nov 2006 19:14:58 GMT - More goodies and freebies available for all registered Express users, including MAKE magazine discounts, Intermatic home automation rebates, and much more! Register Today!

Figure 2-23: Visual C# Express Initial Start-Up Screen

Name the project HelloWorld.



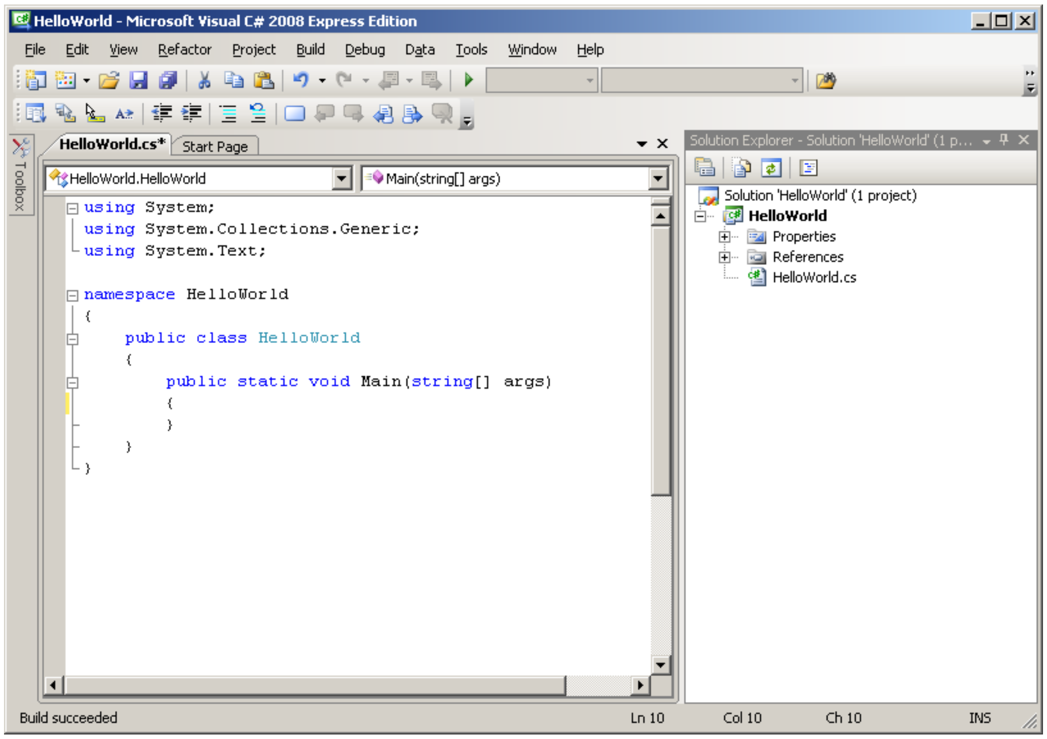
A project for creating a command-line application (.NET Framework 3.5)

Name: HelloWorld

Figure 2-24: New Project Dialog Showing Console Application Selected

and properties. IntelliSense can potentially save you a lot of time looking up .NET Framework class information, but it won't do all the work for you.

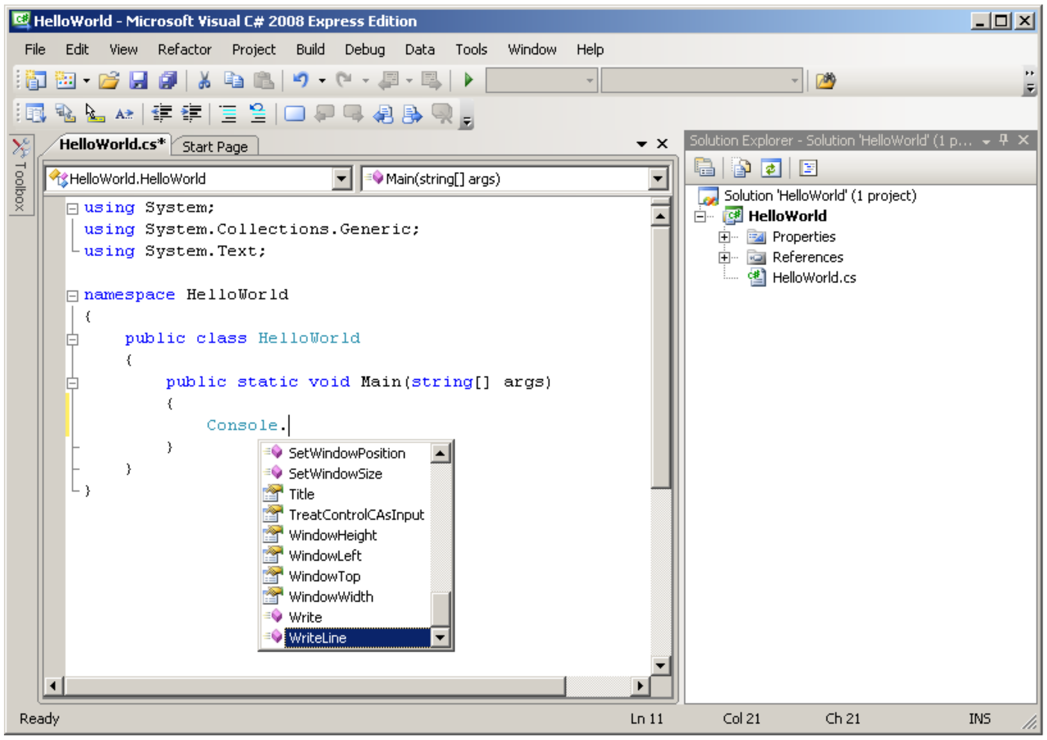
When you have finished making the changes to the automatically generated code, the project should look similar to Figure 2-27.


HelloWorld - Microsoft Visual C# 2008 Express Edition
File Edit View Refactor Project Build Debug Data Tools Window Help
Toolbox
HelloWorld.cs*
Start Page
HelloWorld.HelloWorld
Main(string[] args)
Solution Explorer - Solution 'HelloWorld' (1 p...
Solution 'HelloWorld' (1 project)
HelloWorld
Properties
References
HelloWorld.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace HelloWorld
{
    public class HelloWorld
    {
        public static void Main(string[] args)
        {
        }
    }
}
```

```
Build succeeded                Ln 10     Col 10     Ch 10     INS
```

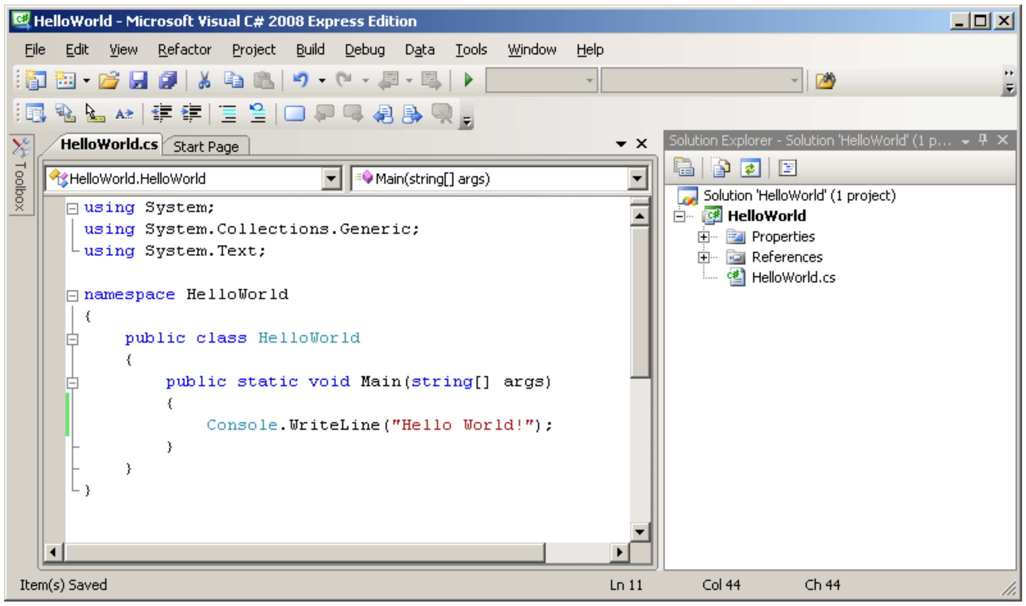
Figure 2-25: HelloWorld Project View



```
using System;
using System.Collections.Generic;
using System.Text;

namespace HelloWorld
{
    public class HelloWorld
    {
        public static void Main(string[] args)
        {
            Console.
        }
    }
}
```

```
Ready                Ln 11     Col 21     Ch 21     INS
```

Figure 2-26: IntelliSense Pop-Up Window Showing Available Console Object Methods and Properties

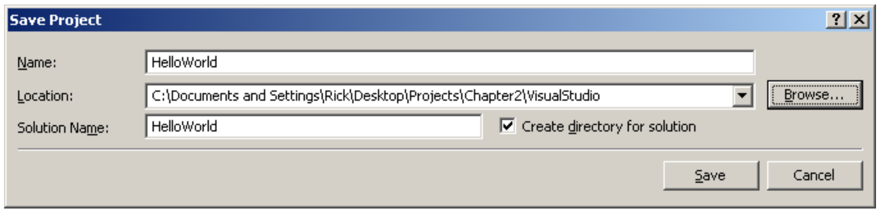

HelloWorld - Microsoft Visual C# 2008 Express Edition
File Edit View Refactor Project Build Debug Data Tools Window Help
HelloWorld.cs
Start Page
HelloWorld.HelloWorld
Main(string[] args)
Solution Explorer - Solution 'HelloWorld' (1 p...
Solution 'HelloWorld' (1 project)
HelloWorld
Properties
References
HelloWorld.cs
Toolbox
Item(s) Saved
Ln 11
Col 44
Ch 44

```
using System;
using System.Collections.Generic;
using System.Text;

namespace HelloWorld
{
    public class HelloWorld
    {
        public static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

Figure 2-27: Updated HelloWorld Visual C# Project

Saving The Project

Next, save the HelloWorld project by selecting File->Save All from the main menu to open the Save Project dialog. You can create a new folder in your Projects folder and save it there.



Name: HelloWorld
Location: C:\Documents and Settings\Rick\Desktop\Projects\Chapter2\VisualStudio
Solution Name: HelloWorld
[✓] Create directory for solution

Figure 2-28: Saving the HelloWorld Project

Build The Project

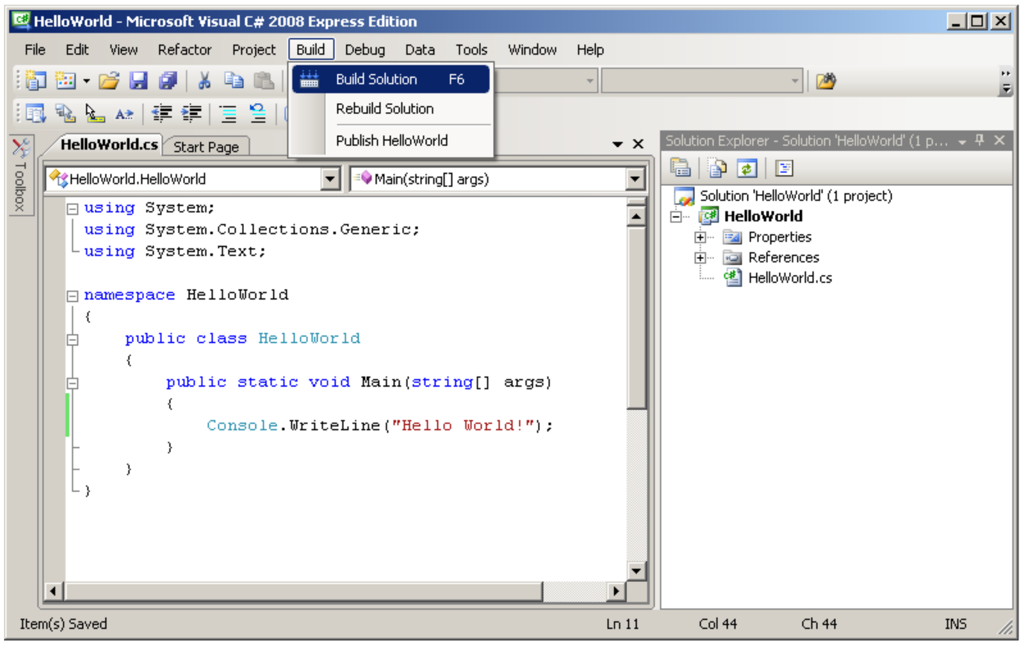
Before you can run the project you must build it. Building the project compiles the source files and generates an executable file (i.e., a .exe file). Select Build->Build Solution from the main menu or press the F6 key as is shown in Figure 2-29.

Locating The Project Executable File

To run the HelloWorld project, navigate to the folder in which Visual C# saved the executable file. You could run the project directly from Visual C# Express. But, because HelloWorld simply prints a short message to the screen and then immediately exits, you'll have to be quick to catch the program's output. This type of short console application is best run from the command line.

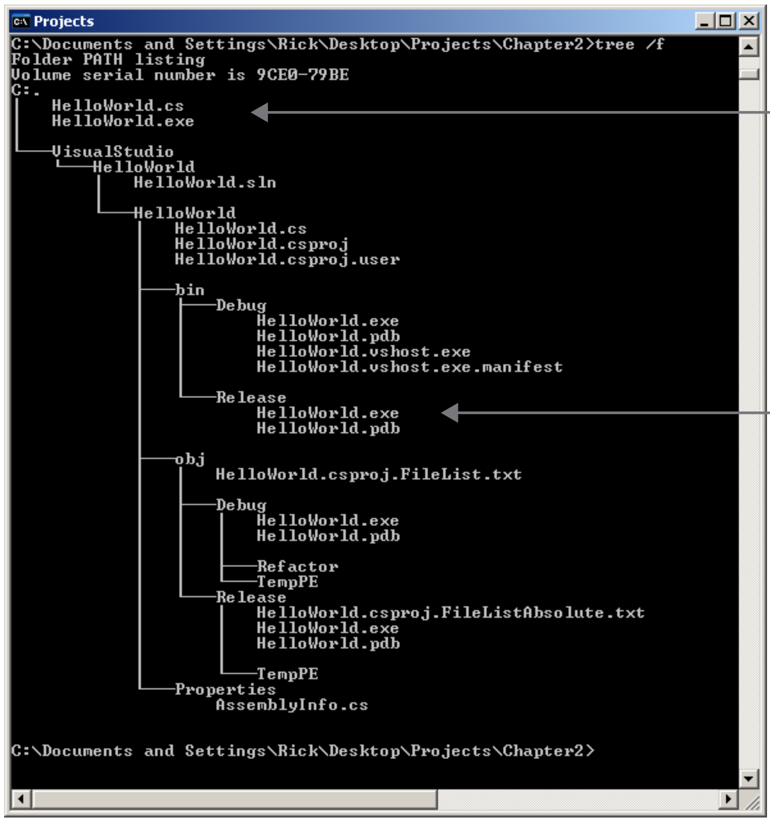
Find the HelloWorld.exe file by opening a command console window, changing to the directory in which you saved the project, and then executing the `tree /f` command at the command prompt. This will give you a directory and file listing similar to that shown in Figure 2-30.

Notice that Visual C# Express automatically creates many subdirectories and files in the project's directory. Some of this data is used to maintain project state information, like the number and types of files it contains, along with debugging information.


HelloWorld - Microsoft Visual C# 2008 Express Edition
File Edit View Refactor Project Build Debug Data Tools Window Help
Build Solution F6
Rebuild Solution
Publish HelloWorld
HelloWorld.cs
Start Page
HelloWorld.HelloWorld
Main(string[] args)
Solution Explorer - Solution 'HelloWorld' (1 p...
Solution 'HelloWorld' (1 project)
HelloWorld
Properties
References
HelloWorld.cs
Item(s) Saved
Ln 11
Col 44
Ch 44
INS

```
using System;
using System.Collections.Generic;
using System.Text;

namespace HelloWorld
{
    public class HelloWorld
    {
        public static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

Figure 2-29: Building HelloWorld Project


Projects
These are the original files created in the command-line tools section.
Use the executable file located in the ..\bin\Release directory.

```
C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>tree /f
Folder PATH listing
Volume serial number is 9CE0-79BE
C:.
│   HelloWorld.cs
│   HelloWorld.exe
│
└───VisualStudio
    └───HelloWorld
        │   HelloWorld.sln
        │
        └───HelloWorld
            │   HelloWorld.cs
            │   HelloWorld.csproj
            │   HelloWorld.csproj.user
            │
            ├───bin
            │   ├───Debug
            │   │       HelloWorld.exe
            │   │       HelloWorld.pdb
            │   │       HelloWorld.vshost.exe
            │   │       HelloWorld.vshost.exe.manifest
            │   │
            │   └───Release
            │           HelloWorld.exe
            │           HelloWorld.pdb
            │
            ├───obj
            │   │   HelloWorld.csproj.FileList.txt
            │   │
            │   ├───Debug
            │   │   │   HelloWorld.exe
            │   │   │   HelloWorld.pdb
            │   │   │
            │   │   ├───Refactor
            │   │   └───TempPE
            │   └───Release
            │       │   HelloWorld.csproj.FileListAbsolute.txt
            │       │   HelloWorld.exe
            │       │   HelloWorld.pdb
            │       │
            │       └───TempPE
            └───Properties
                    AssemblyInfo.cs


C:\Documents and Settings\Rick\Desktop\Projects\Chapter2>
```

Figure 2-30: Results of Running the `tree /f` Command from the Command Prompt

As you can see from figures 2-28 and 2-30, I have saved my version of the Visual C# Express HelloWorld project in a directory named “VisualStudio” located in the Chapter2 directory of the Projects directory.

Note to Students: One good reason you need to know where to find the executable file is so you can turn in executables from programming assignments to your instructor!

Execute The Project

Execute the HelloWorld program generated by Visual C# Express by opening the command console and navigating to the `..bin\Release` directory using the `cd` command. Enter “HelloWorld” at the command prompt and press Return or Enter. If you want to avoid typing long path names at the command prompt, open the Projects folder and navigate to the `..\Chapter2\VisualStudio\HelloWorld\HelloWorld\bin\Release` folder using Windows Explorer. When you get there, copy the path from the Address bar and paste it into the console window after the `cd` command by right-clicking the console window and clicking paste. (*In this case, using the keyboard shortcut ctrl-v key combination will only paste in the characters “^V”, which is not going to help you much.*)

Note: Programs, such as a console application that displays a text menu or a Windows Forms GUI application, that do not exit immediately can be executed successfully from the Visual C# environment.

Where To Go For More Information About Visual C# Express

For more information about Visual C# Express visit the MSDN website or refer to the documentation that came with the application.

Quick Review

Visual C# Express Edition is a lightweight version of Microsoft’s flagship IDE Visual Studio. Visual C# Express comes bundled with SQL Server Compact Edition, which allows you to create relational database applications.

Visual C# Express will automatically generate a lot of source code for your project. However, it will not generate all the code your application needs. It’s imperative you know how to modify the code that Visual C# generates in order to take control of your projects.

When you save your Visual C# Express project, it generates many different files and subdirectories. You can find the project’s executable file in the `..\bin\Release` directory.

Summary

All you need to create robust Microsoft C# applications is a good text editor and the free Microsoft C# command-line compiler that’s included with the .NET Framework Redistributable Package.

Before using the C# command-line compiler you must configure your development environment. This includes creating or editing one or more operating system environment variables. An environment variable is a named location in memory used by Microsoft Windows to store data about the operating system environment. There are generally two types of environment variables: system and user. System environment variables store data that pertains to and affects the operating system environment for all users; user environment variables store data that pertains to and affects the operating system environment for a particular user.

Environment variable values can be accessed by enclosing the variable name in ‘%’ characters.

The Path environment variable is used by the operating system to help it locate executable files. You must create or edit the Path environment variable to include the full path to the C# compiler. (csc.exe)

It’s helpful to create a project folder and a shortcut to the command console on your desktop. Set the command console shortcut’s **Start in** property so it will automatically open in your designated project folder. Increase the command console shortcut’s screen buffer **height** and **width** properties to see more information in the console window.

It’s also a good idea to set your folder options to display file type extensions. This will prevent headaches associated with accidently saving source files with a “.txt” extension.

To create a C# program with the command-line compiler you must create the source file, compile the source file with the `csc` command-line compiler tool, and then execute the program by typing its name at the command prompt and pressing the Return or Enter key.

You're bound to get a few compiler errors when you start writing your own programs. Go to Microsoft's website to look up the error code and always remember to **fix the first compiler error first**!

Visual C# Express Edition is a lightweight version of Microsoft's flagship integrated development environment (IDE) Visual Studio. Visual C# Express comes bundled with SQL Server Compact Edition which allows you to create relational database applications.

Visual C# Express automatically generates a lot of source code for your project. However, it will not generate all the code your application needs. It's imperative you know how to modify the code that Visual C# generates in order to take control of your projects.

When you save your Visual C# Express project it generates many different files and sub directories. You can find the project's executable file in the `..\bin\Release` directory.

Skill-Building Exercises

1. **Creating and Using Environment Variables:** Create an environment variable named "PROJECTS_HOME". For its value use the path to your projects folder.

2. **Setting Up Your Development Environment**: Set up your development environment following the steps outlined in this chapter. Test your development environment by compiling and running the program given in Example 2.1.

3. **Web Research**: Visit the MSDN website and familiarize yourself with the information it contains. Locate the C# compiler errors page and bookmark the page in your web browser.

4. **.NET Software Development Kit (SDK)**: Download and install the .NET SDK. List and provide a brief description of the purpose of each component.

Suggested Projects

1. **Alternative .NET Development Environments** - If you are interested in doing C#.NET development on alternative computing platforms, the Mono development environment may help. Visit the Mono Project website [www.mono-project.com] to learn more.

Self-Test Questions

1. What two things, at minimum, do you need to do C#.NET development?

2. What is an operating system environment variable?

3. What is the difference between a user vs. a system environment variable?

4. How do you create an environment variable in Microsoft Windows XP?

5. What characters must you use before and after an environment variable to get its value?

6. What is the purpose of the Path environment variable?

7. What should you do if you get more than one compiler error?

8. What's the advantage of using an IDE like Visual C# Express?

9. What are the general steps required to create, compile, and execute a C# program?

10. What's the recommended way to run a C# program that runs briefly and exits immediately after executing?

References

Microsoft Developer's Network website, [www.msdn.com]

Mono Project website, [http://www.mono-project.com/Main_Page]

Notes