Chapter 2

Configuring Your Development Environment

Learning Objectives

- *List the software required for ASP.NET development*
- *Be aware of suggested hardware configurations that facilitate the development process*
- *State the advantages of a dual-monitor configuration*
- *Arrange your Windows Desktop to better facilitate access to software development tools*
- *Understand how to structure a project folder*
- *List the steps required to set operating system environment variables*
- *Locate the installation folder for the Microsoft.NET framework*
- *Use your engineer's notebook to record software installation details*
- *Set the properties on a command prompt icon to increase window and buffer size*
- *Install and configure required Microsoft development tools*
- *Install Microsoft Enterprise Library 5.0*
- *Install NUnit*
- *Install Log4Net*
- *Install NAnt*

© 2014 Rick Miller — All Rights Reserved

Introduction

You can significantly enhance your productivity as a software developer by giving some thought regarding how to configure your development environment. This holds especially true when developing ASP.NET applications. Before you write one line of code you must install, at the very least, some version of Microsoft Visual Studio. But if you stop there you will be short-changing yourself. There are many efficiencies to be gained from a properly configured development environment.

In this chapter I will give you a fairly complete list of required and optional development tools. While some tools or third-party libraries such as NUnit and Log4Net strictly fall under the optional category, I consider them indispensable and soon you will too.

During the course of your professional software development career you will find it necessary to use the command prompt window, and although you may not use it much now, soon its use will become second nature. The command prompt will better serve your needs if you make a few minor changes to its properties such as its physical size and the size of its buffers. I'll show you how to make these and other changes to the command prompt so it better serves your needs as a software engineer.

A normal installation of Microsoft Windows lacks services you'll need for software development like web hosting and distributed transactions. I'll show you how to turn these on in Microsoft Windows 7. Later in the book when I talk about deploying applications to Windows Server 2008 R2 I'll show you how to enable required services in that operating system.

I'll also discuss a possible virtual machine configuration that will enable software development, deployment and testing in a more realistic environment.

ASP.NET Development Software

To develop ASP.NET applications you'll need a variety of software development tools, applications, libraries, and one or more suitable operating systems. Let's talk first about operating systems.

Microsoft Operating Systems

At the time of this writing I strongly recommend using Microsoft Windows 7 Professional or Windows 7 Ultimate as your development environment operating system. Windows XP Professional is still a good, solid-performing operating system but is nearing the end of its life cycle. Windows 7 is perhaps the most stable and reliable of all Microsoft operating systems with perhaps the exception of their Microsoft Windows Server 2003 and 2008 editions. I'm not a big fan of Windows 8 no matter how hard Microsoft tries to sell it. Just my two cents.

I am using Microsoft Windows 7 Ultimate running in a Parallels virtual machine image under Apple's OSX Snow Leopard. I talk more about virtual machine configurations in the Virtual Machines section.

I recommend at least 4 Gigs of RAM when running Windows 7 Professional or Ultimate and 8 Gigs of RAM just to be on the safe side. On my instance of Windows 7 all required Windows services loaded and running consume less than 1 Gig of memory. This includes IIS 7 web services. This leaves plenty of room for running Visual Studio 2010 and an SQL Server 2008 R2 database. If you like to listen to music and stream video while coding then you might want more memory.

If you purchase a laptop or desktop with a Home edition of a Microsoft operating system I strongly recommend you upgrade to Windows 7 Professional or Windows 7 Ultimate. The reason is because the Home edition, consumer-oriented operating system versions completely omit important capabilities required for testing certain security configurations. A specific example of this is Windows Authentication for IIS 7. Also, to get SQL Server 2008 R2 or IIS 7 to run on a Home edition requires loading several non-standard updates to the operating system.

 © 2014 Rick Miller — All Rights Reserved

To realistically deploy and test an ASP.NET application I recommend Windows Server 2008 R2. There are several important configuration setting differences between IIS 7 running under Windows 7 vs. Windows Server 2008. Also, to properly test distributed transactions you'll want to deploy your application on one instance of Windows Server 2008 and deploy the database on a separate instance. I talk more about how to accomplish all this on one physical machine in the Virtual Machines section.

Microsoft Tools, Applications, and Libraries

Visual Studio is Microsoft's flagship Integrated Development Environment (IDE) with 2010 being the edition I use in this book. You could use a later edition with no problem. I recommend Visual Studio Professional. You can use Visual Studio Express Edition but it lacks features and most maddening its menu choices don't necessarily correspond with Visual Studio Professional's. For example, creating a complex project is straightforward in Visual Studio Professional because there's a Solution template, but not in the Express edition. Creating a complex project in the Express edition can be done but it requires some finesse.

In addition to Visual Studio you'll need Microsoft SQL Server 2008 R2. This is Microsoft's flagship database application software. It too comes in an Express edition and for development purposes the Express edition is fine. The differences between the two products that will affect you most are connection strings and minimum database file sizes. For realistic testing you will eventually want to use the standard edition of MS SQL Server 2008 R2 deployed on an instance of Microsoft Windows Server 2008 R2.

You'll need to download and install the Microsoft Enterprise Library 5.0 to write enterprise database applications. Database connections are specified in configuration files and the Enterprise Library Application Block provides a database connection factory.

Third-Party Tools

This section presents some important third-party tools I'll be using in this book as part of my development tool repertoire.

Log4Net

Complex web applications rely heavily on logging. Everything can potentially be logged from user access attempts to exceptions thrown by the code. During the development process you can log debug messages to help you troubleshoot problems, and trust me, there will be problems, especially when you start taking your first tentative steps towards database programming. For application logging I'll be using Log4Net. It's a robust logging framework that's easy to integrate into your ASP.NET applications.

NUnit

Mature software engineers understand the importance of unit testing. To the novice, unit testing at first seems like a lot of extra work. But the "testing dividend" pays off handsomely in the end. With careful thought and implementation of unit testing you can significantly increase the quality of the code you produce and ultimately decrease your development time. I'll be using a tool called NUnit for unit testing.

Notepad++

Even though Visual Studio offers an excellent built-in text editor, you'll want a separate text editor that's more robust than Notepad or WordPad supplied by the Windows operating system. Notepad++ is my text editor of choice.

Not much else to say here except that while you may be tempted to edit configuration, source code, SQL, or batch files with a word processor like Microsoft Word, be careful. Chances are you'll screw up the files by introducing hidden formatting codes. Be smart and go Notepad++ early.

 © 2014 Rick Miller — All Rights Reserved

Subversion (SVN)

Even when working alone it's nice to keep your project under version control. I like to use Subversion (SVN) as my source code version control system of choice. There are other popular version control systems: GIT, CVS, and even Microsoft's Visual Source Safe just to name a few. I've used Subversion for several years both personally and professionally so I'll stick with that in this book. You can download Subversion from the Apache Subversion project site http://subversion.apache.org or from CollabNet http://www.open.collab.net/downloads/subversion.

TortoiseSVN

TortoiseSVN is a nice GUI-based Subversion client. It integrates perfectly with Microsoft Windows and makes using Subversion a breeze. You can get TortoiseSVN from http://tortoisesvn.net.

NAnt

NAnt is an open-source build tool that is similar to the Java build tool Ant. Although you'll normally set-up and build solutions and projects using Microsoft Visual Studio, you can use NAnt to build your projects from the command line. This is helpful if you'd rather use another text editor to edit project files but still have the capability to build complex projects. NAnt can also be used to perform continuous integration in conjunction with Cruise Control. You can download NAnt from http://nant.sourceforge.net/.

Development Tools Summary Table

Table 2-1 provides a summary listing of the operating systems, development tools, and applications discussed in this section.

OS/Tool/Application/Library	Description/Availability
Microsoft Windows 7 Professional Microsoft Windows 7 Ultimate	Recommended operating system for general software development environment. If you're a student you can buy from your campus software store or get free from Microsoft DreamSpark https://www.dreamspark.com
Windows Server 2008 R2	Recommended operating system for realistic deployment testing. If you're a student you can buy from your campus software store or get free from Microsoft DreamSpark.
Microsoft Visual Studio 2010 or later edition Professional or Express Editions	Microsoft's flagship Integrated Development Environment (IDE). Express edition can be downloaded free from MSDN.com. You can download Visual Studio Professional free from Microsoft DreamSpark.
Microsoft SQL Server 2008 R2 Standard and Express Editions	Microsoft's flagship relational database server. The Express edition can be downloaded free from MSDN.com. You can download the Standard edition from Microsoft DreamSpark.
Microsoft Enterprise Library 5.0	Provides supporting libraries that enable enterprise application programming. Download free from MSDN.com.
Notepad++	A great text editor. Download from http://notepad-plus-plus.org
NUnit	Testing framework available from http://www.nunit.org
Log4Net	Logging framework available from http://logging.apache.org/log4net

Table 2-1: Development Tool Summary

© 2014 Rick Miller — All Rights Reserved

OS/Tool/Application/Library	Description/Availability
Subversion	Source code version control system available from two sites: Apache Subversion Project http://subversion.apache.org CollabNet http://www.open.collab.net/downloads/subversion
TortoiseSVN	A GUI-based Subversion client available from http://tortoisesvn.net
NAnt	A command-line build tool available from http://nant.sourceforge.net/

Table 2-1: Development Tool Summary

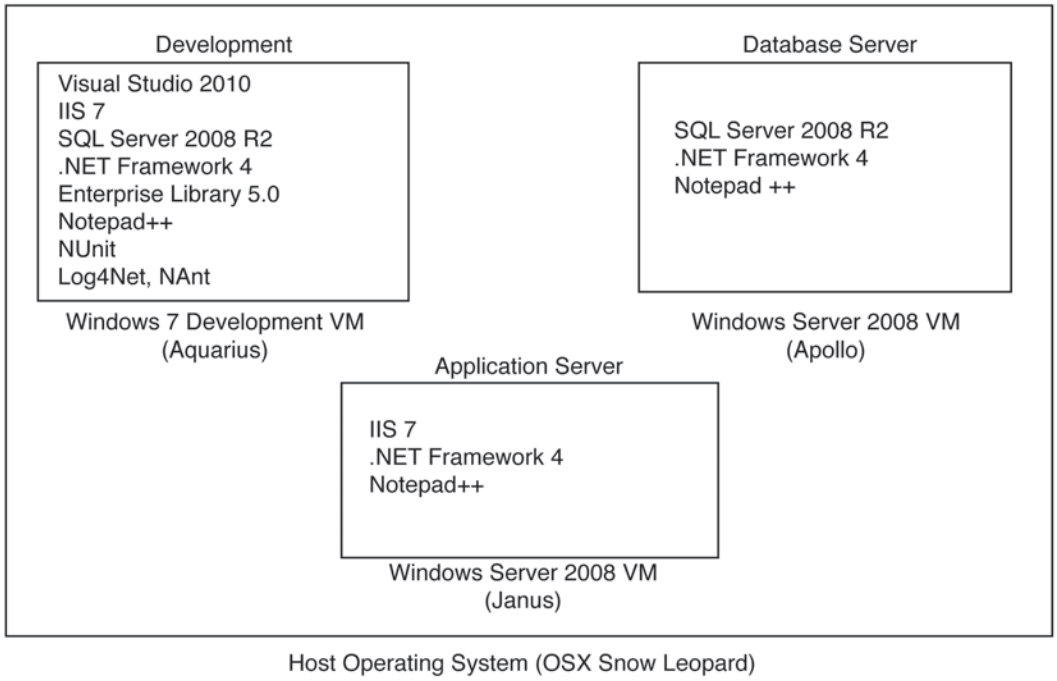
In the following section I'd like to talk about how you can deploy the operating systems, development tools and libraries discussed above on a single physical machine using virtual machines.

Virtual Machines

You can, of course, develop ASP.NET applications with a desktop or laptop machine with all your development tools loaded onto the physical machine itself, but virtual machines make it easier to test distributed deployment scenarios where the application is deployed on one machine and the database is deployed on another.

My VM Environment

As I said in chapter 1, I'll be using virtual machines to present the material in this book. Figure 2-1 shows the virtual machine configuration I'll be using.



My Virtual Machine Configuration

Figure 2-1: My Virtual Machine Configuration

 © 2014 Rick Miller — All Rights Reserved

Referring to figure 2-1 — I'll be running three different virtual machines, each configured for a specific purpose. My physical computer is an Apple Mac Pro with dual 4-core Xeon processors, 32 Gigs of RAM and 5 Terabytes of RAID 5 hard disk space. Needless to say I keep this machine plugged into an uninterruptible power supply (UPS) in case of power fluctuations or failures. On this machine I've installed a VM hosting environment called Parallels. With Parallels I have created three VM images. On the first, shown in the upper left-hand corner of figure 2-1, I have installed Windows 7 Ultimate. I'll be using the Windows 7 VM as my primary development and testing environment. On this VM image I've loaded Microsoft Visual Studio 2010 and the other software listed in the diagram. I've named this computer Aquarius.

I created two more VM images and loaded Windows Server 2008 on each one. On the application server named Janus I enabled IIS 7 and loaded the .NET Framework 4 and Notepad++. On the database server named Apollo I've loaded SQL Server 2008 R2, .NET Framework 4, and Notepad++. I will use these two VMs to test distributed transactions in a realistic deployment environment.

Note that today it matters little what host operating system you are using. There are several outstanding VM hosting applications on the market. For the Apple OSX operating system there's Parallels. I could have also used Oracle VirtualBox. For Windows you can use VMware Workstation or Oracle VirtualBox. For Linux users you can also use VMware Workstation or Oracle VirtualBox. For Solaris users there's Oracle VirtualBox. These are just the ones with which I'm personally familiar.

Table 2-2 lists virtual machine hosting applications and where to get them.

VM Hosting Application	Description
Parallels	A VM hosting application that runs on Apple OSX and Intel-based Apple Mac computers. Available from http://www.parallels.com.
VMware Workstation	A VM hosting application that runs on Windows and Linux machines. Available from http://www.vmware.com.
Oracle VirtualBox	An open-source VM hosting application that runs on Windows, Apple OSX, Linux, and Solaris. Available from https://www.virtualbox.org.

Table 2-2: Virtual Machine Hosting Applications and Sources

VM Networking – Bridged vs. NAT

When creating a virtual machine image in Oracle VirtualBox or VMware Workstation, you can configure the VM's networking option to use either Bridged or NAT. If you only intend to test VM connectivity between VMs on one physical machine then NAT or Shared networking will work fine. The Bridged option makes it easier to test VM-to-VM connectivity across physical machine boundaries by making the VMs appear to be separate physical machines on the same network. Each of my Parallels VM images is configured to use Shared networking. You should be able to ping from one VM to another, say, from your development VM to the database server VM or from your application server VM to the database server VM.

Allocating VM Processors, Memory and Hard Disk Space

To the Windows 7 Development VM I've allocated 2 processors, 4 Gigs of memory, and 50 Gigs of hard disk space. Since the workload on the Window Server 2008 VMs will be relatively low, I'll only allocate one processor each, 4 gigs of RAM and 50 Gigs of hard disk space.

 © 2014 Rick Miller — All Rights Reserved

Turning On Windows Features In Windows 7

Upon first loading Windows 7 Professional or Ultimate, many advanced features and services like IIS 7 web hosting are, by default, switched off. To turn them on click **Start->Control Panel** and click **Programs**. Click **Turn Windows features on or off**. This will present you with a Windows Features dialog that will allow you to check the features you want to activate. Figure 2-2 shows a partial view of my Windows Features dialog window.

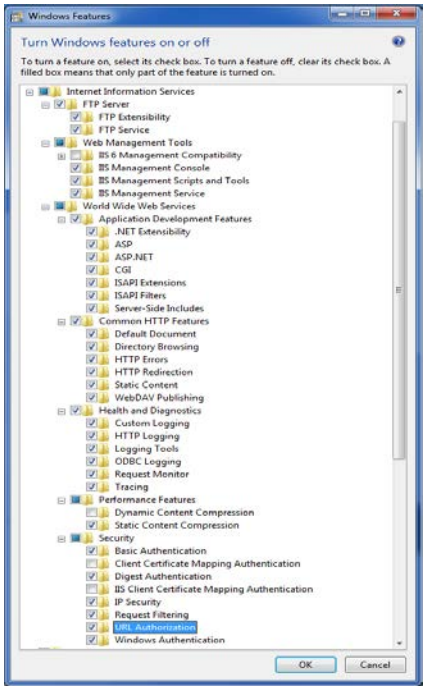


Figure 2-2: Turning On Windows Features

Most of the services I've activated relate to Internet Information Services (IIS). I turned on the IIS Management Console, IIS Scripts and Tools, and IIS Management Service. I've activated almost all the World Wide Web Services. Under the Security sub-list I've checked Basic Authentication, Digest Authentication, IP Security, Request Filtering, URL Authorization, and Windows Authentication. Some of these were checked by default. I specifically checked Windows Authentication to test Intranet applications. Note that if you don't enable Windows Authentication here it won't show up in the list of available authentication choices in the IIS 7 configuration management console.

I've enabled all the Application Development Features. These include .NET Extensibility, ASP, ASP.NET, CGI, ISAPI Extensions, ISAPI Filters, and Server-Side Includes. I've also enabled all the Common HTTP Features.

When you've finished make a note in your engineer's notebook that this is how you enable certain features and click **OK**.

If you enabled several new features it will take Windows a few minutes to start everything up so there will be a bit of a pause after you click the **OK** button.

Starting and Stopping Windows Services

The next window you should become familiar with is the Computer Management Console. To open the Computer Management Console click the **Start** button and right-click **Computer**. From the pop-up menu click **Manage**. Figure 2-3 shows the Computer Management Console as it appears in my Windows 7 instance.

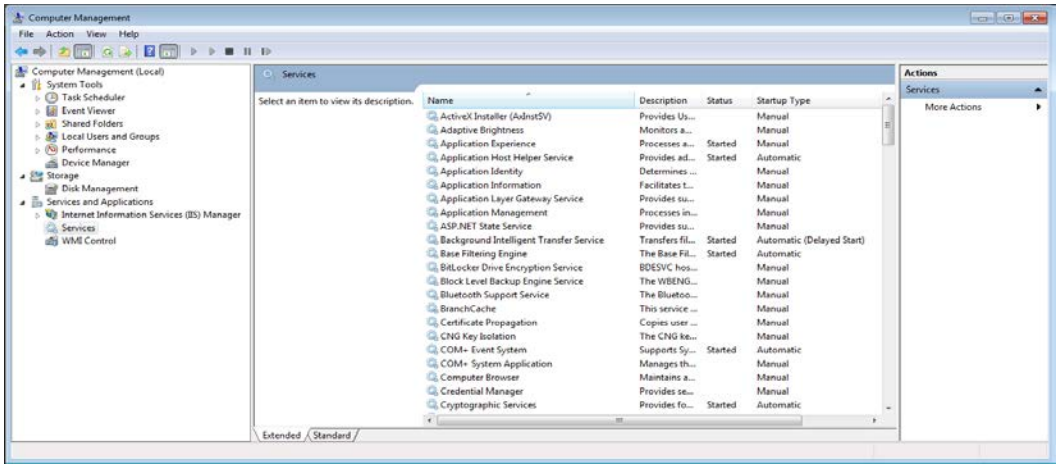


Figure 2-3: Computer Management Console with Services Window Displayed

© 2014 Rick Miller — All Rights Reserved

Referring to figure 2-3 — In the left-hand column click the triangle shape to the left of Services and Applications to expand the selection. Click on **Services** to bring up the list of services running on your machine. Take the time now to browse this list to get a feel for what's up and running. As you'll notice, some services are set to start automatically while others are set to manual start-up.

Often times during development of complex ASP.NET applications you'll need to visit the services window to start and stop services, and always having to bring up the Computer Management Console is a royal pain in the ass. That's why I like to keep the Services window, among others, as icons on my desktop and pinned to the taskbar for easy access. I'll show you how to do this in the following section.

An alternative way to access the frequently used computer administrator tools is to click **Start->Control Panel**. This will bring up the Control Panel window as shown in figure 2-4.

Figure 2-4: Control Panel Default View

Referring to figure 2-4 — The default view is by Category as indicated by the dropdown in the upper right corner. Change the View by: dropdown to **Small icons**. Your control panel window will now look similar to figure 2-5.

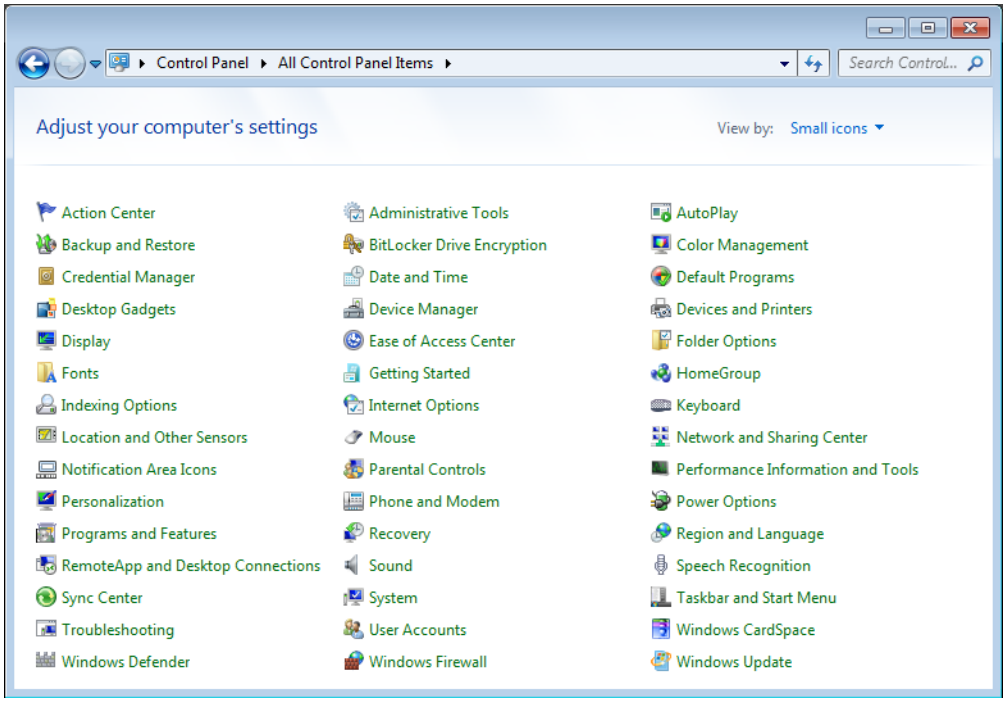


Figure 2-5: Control Panel Arranged by Small Icons

Referring to figure 2-5 — At the top of the center column of icons you'll see one called **Administrative Tools**. Click it to open the Administrative Tools window as shown in figure 2-6.

 © 2014 Rick Miller — All Rights Reserved

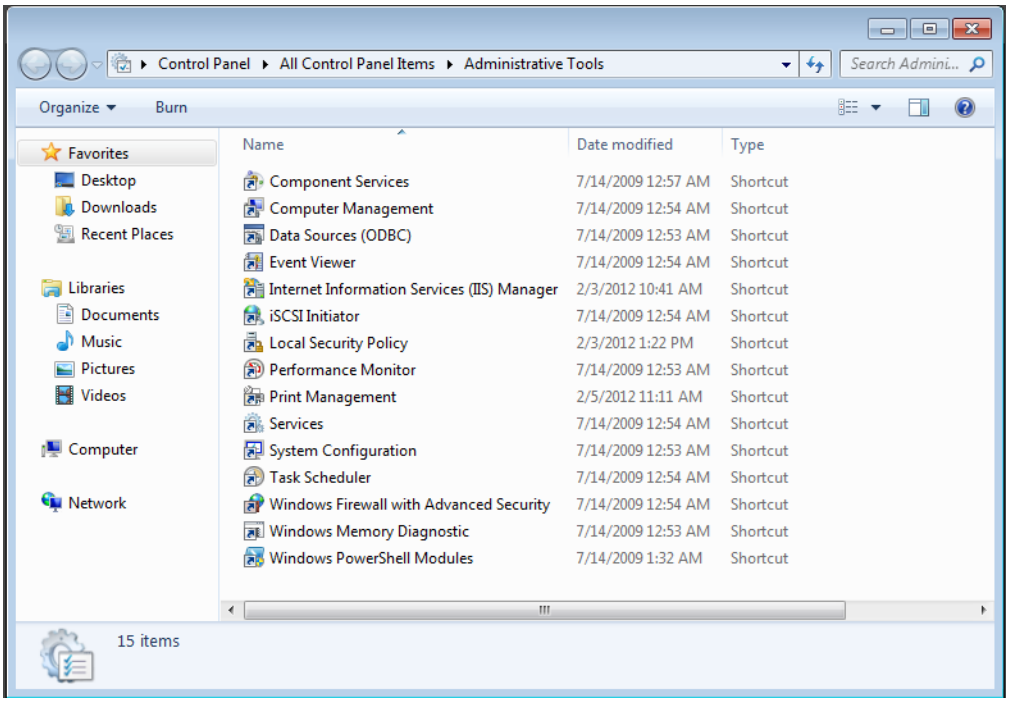


Figure 2-6: Administrative Tools Window

Referring to figure 2-6 — The Administrative Tools window offers convenient access to frequently used administrative tools like the Internet Information Services (IIS) Manager, and Services.

Easing Access To Development Tools

As a software engineer you'll quickly grow tired of fiddling with deeply nested windows just to pull up frequently-used tools and applications. In this section I want to show you how you can arrange your desktop to speed access to your favorite applications.

First, let's start with the Services window I introduced you to in the previous section. I like to keep this and other tools close at hand, either pinned to my taskbar, as icons on my desktop, or both. To find the Services application navigate to the C:\Windows\System32 folder and scroll down until you find the Services.msc file. If you haven't already done so, change your folder view properties to display known file types. (See the next section.) In this case, we're not looking for the Services.exe file. I know, a bit strange. It's the Services.msc file we're after. Right-click the Services.msc (Microsoft Common Console Document) and roll the cursor up to **Send to** and when the pop-up menu appears to the right select **Desktop** (create shortcut). You should see the geared icon appear on your desktop. Next, right-click **Services.msc** again and this time select **Pin to Taskbar**. You should see the geared icon appear on your taskbar. Figure 2-7 shows you what my Windows 7 development environment desktop looks like after I've done this.

Referring to figure 2-7 — Note that the Services console is now readily available from two convenient locations. No more mucking around through the Computer Management Console. Note too that I have already added the Command Prompt icon to my taskbar. Go ahead now and do the same. While you're at it, add it to your desktop. You'll find the Command Prompt by clicking **Start->All Programs->Accessories**. Right-click the **Command Prompt** icon and click **Send To** to send it to the desktop.

Another icon I like to have handy on my desktop is a shortcut to the hard drive. Click **Start->Computer** and right click on Local Disk (C) and click **Create shortcut** to automatically send the shortcut to your desktop. Repeat this process for any other tools and applications to which you want handy access.

In the next section, I want to show you how you can prevent shooting yourself in the foot by mistakenly misnaming files.

 © 2014 Rick Miller — All Rights Reserved

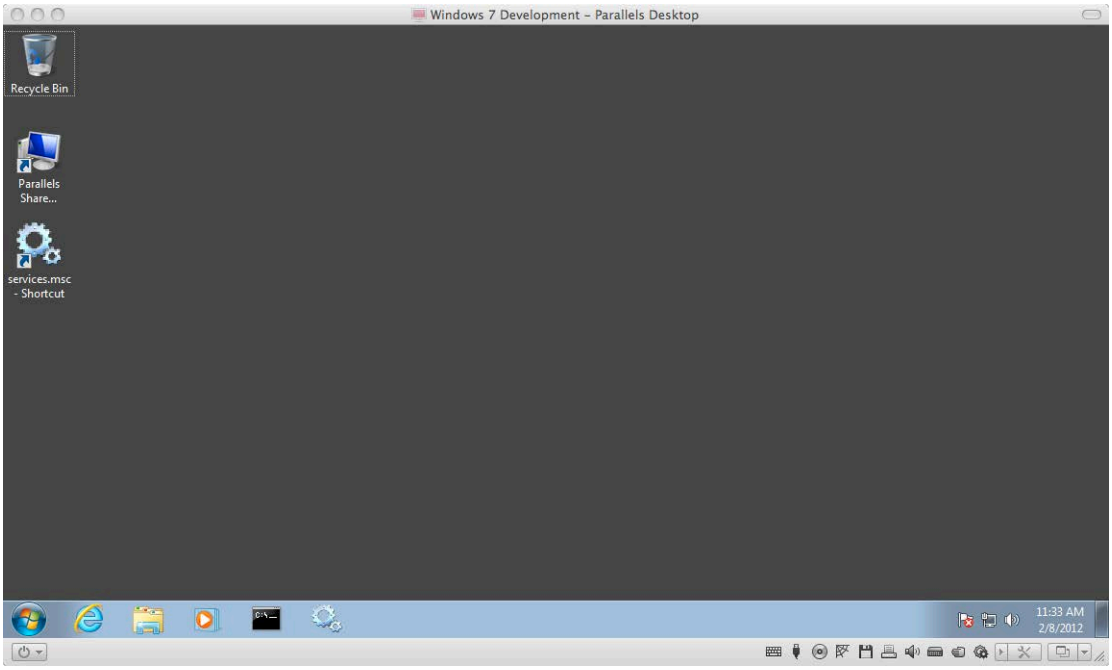


Figure 2-7: Desktop and Taskbar with Services Readily Available

Setting Folder View Properties

I like mysteries except when they involve what I'm doing on my computer. One easy thing you can do to take some of the mystery out of your programming life is to show common file name suffixes in your folder listings. The reason you want to do this is because if you create a source file with a text editor like Notepad and don't pay attention to the file suffix, you might have mistakenly appended the .txt suffix to the file. If common file suffixes are hidden you will not see the .txt suffix. For example, take a look at figure 2.8.

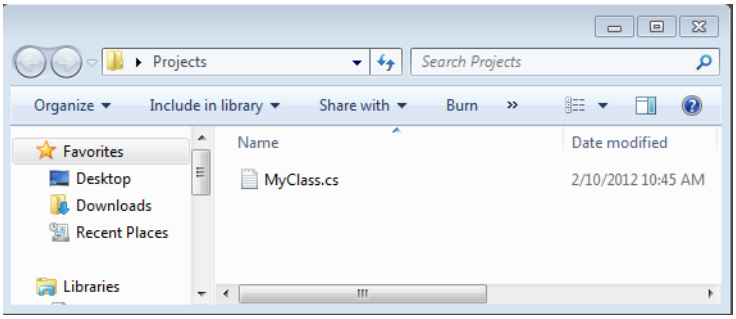


Figure 2-8: Folder with File Suffixes Hidden

Referring to figure 2-8 — At first glance there appears to be a file named MyClass.cs in the projects folder. Let's say you try to compile this file. (I see this happen to novice C# programmers all the time!) You crank up the command prompt and go to the Projects directory and try compiling the MyClass.cs as is shown in figure 2-9.

Now you're scratching your head and asking yourself "How can that be? I can see the file in the folder window!" and if you're really stubborn like me, you'll try compiling the file five or six more times just to be sure you're not doing something wrong. When suddenly you get the itch to try something new and issue a `dir` command from the command prompt, which reveals the reason the compiler can't find MyClass.cs as is shown in figure 2-10.

 © 2014 Rick Miller — All Rights Reserved

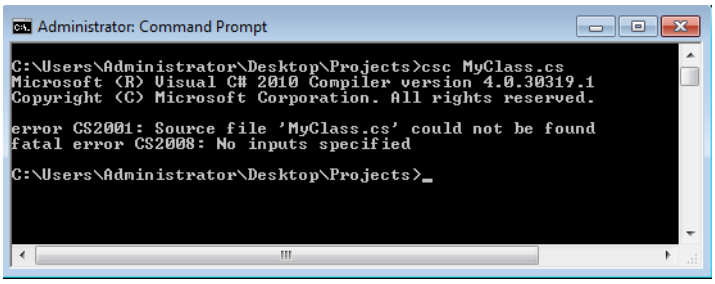
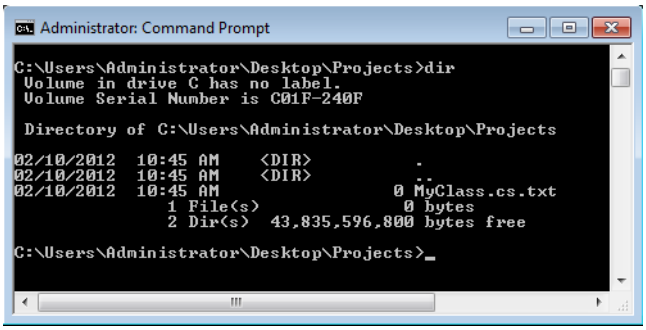


Figure 2-9: Bad Compile Result - MyClass.cs Source File Not Found!



Source file has a .txt file suffix that causes problems when trying to compile.

Uncheck this box

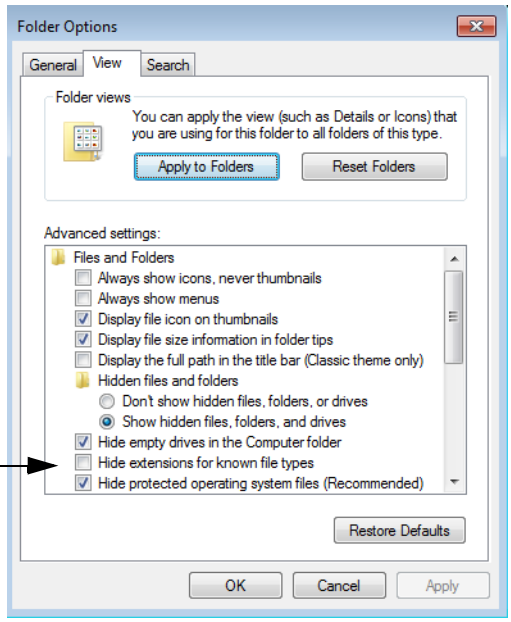


Figure 2-10: Setting Folder View Options

To show hidden file suffixes open a folder window and click **Organize->Folder and search options**. Click the **View** tab and uncheck **Hide extensions for known file types**. Optionally select the radio button above that says **Show hidden files, folders, and drives**. Figure 2-10 shows my Folder Options dialog settings.

Referring to figure 2-10 — when you've made the necessary setting changes, click **Apply** at the bottom of the Folder Options View tab dialog and then click **Apply to Folders** to make the change apply to all folders. Click **OK** to dismiss the dialog. Now the full file names are revealed in folder windows as is shown in figure 2-11.

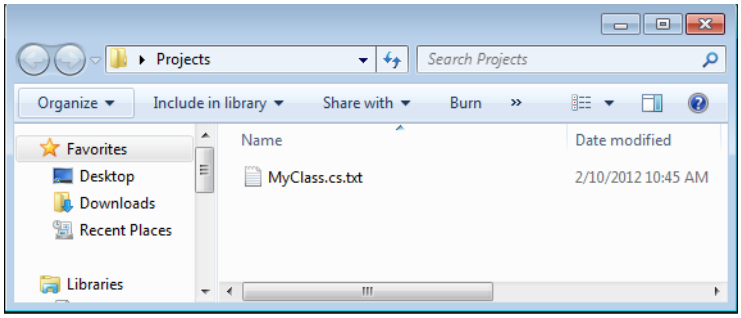


Figure 2-11: Full Filename Suffixes now Revealed

This simple tweak to your development environment falls under the "low hanging fruit" category. Practically the first thing I do when I install an instance of Windows 7 is to make this change to my folder settings.

In the following section I will show you how to set operating system environment variables.

 © 2014 Rick Miller — All Rights Reserved

Setting Operating System Environment Variables

Setting operating system environment variables is one of those system administration skills every software engineer needs to have to properly configure their development environment. I'll show you how to set two system environment variables and when the time comes you'll be able to create them as you need to.

The two variables I'm going to set will allow me to compile C# source files from a command prompt. I find this helpful even if Visual Studio provides a command prompt already set up for this purpose. (It's running a batch file behind the scenes to initialize the required environment variables.)

First, start by navigating to the .NET Framework v4.0.30319 folder. The absolute path on my machine is C:\Windows\Microsoft.NET\Framework\v4.0.30319. Copy this path into the clipboard. Next, click **Start**, right-click **Computer**, and from the pop-up menu select **Properties**. In the Control Panel Home panel click **Advanced system settings**. This will open the System Properties dialog as shown in figure 2-12.

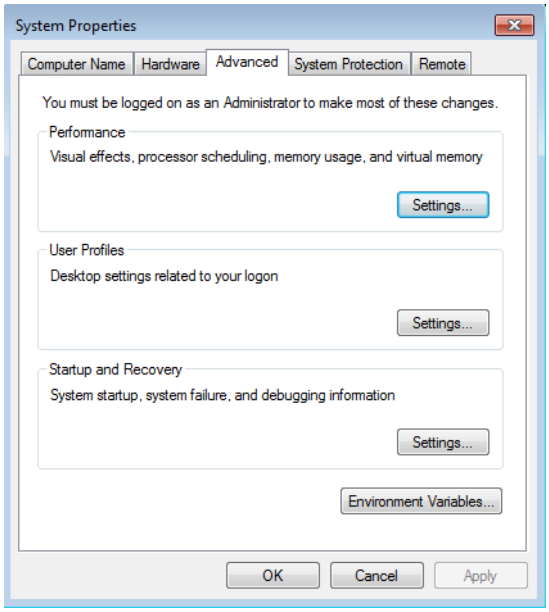


Figure 2-12: System Properties Advanced Tab

Referring to figure 2-12 — click the **Environment Variables...** button to open the Environment Variables dialog as is shown in figure 2-13.

Referring to figure 2-13 — There are two variable areas: the upper area is for use by the current user, in this case the Administrator. The bottom area shows System variables. Generally you need only set user environment variables unless you share the system with other users and want a particular variable to be available for all users. In situations where security is a concern, such as on school lab computers, you may only have access to the user variables section.

To create a new variable click the **New...** button under the upper user variables section. This opens a New User Variable dialog. In the Variable name text box enter the name of the variable you want to create. I use upper case letters with each word separated by an underscore character. Since we're setting a variable to represent the home directory of the .NET Framework name the variable DOT_NET_HOME. In the Variable value text box paste in the absolute path you copied earlier. Click the **OK** button to create the variable.

Next, create another environment variable named PATH and in the Variable value text box enter %DOT_NET_HOME%. This is the name of the DOT_NET_HOME environment variable surrounded by percent signs. This is an example of how to use an existing environment variable. When you're finished click the **OK** button. Click the **OK** button on the Environment Variables dialog to set the changes. If you have a command prompt window open, close it and reopen it before attempting to use the new variables. Figure 2-14 graphically illustrates the steps described above to set environment variables.

 © 2014 Rick Miller — All Rights Reserved

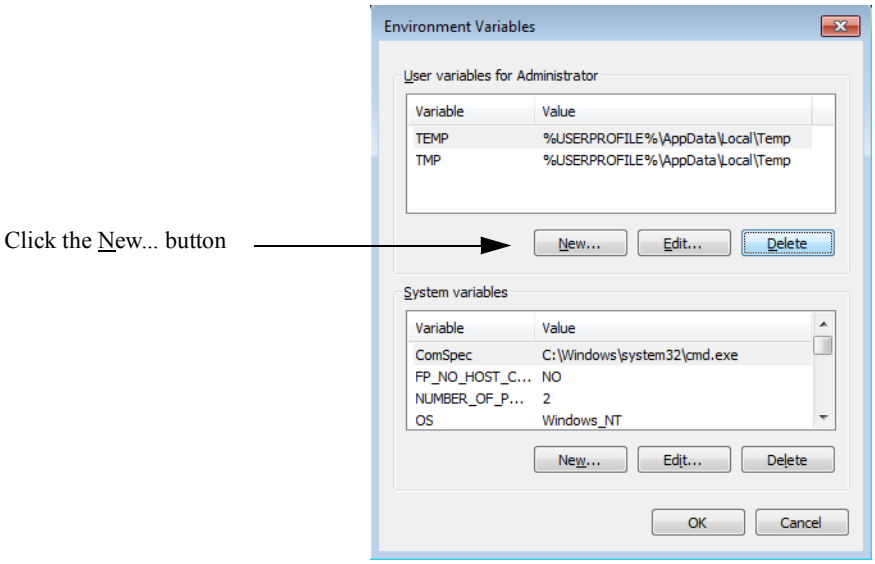


Figure 2-13: Environment Variables Dialog

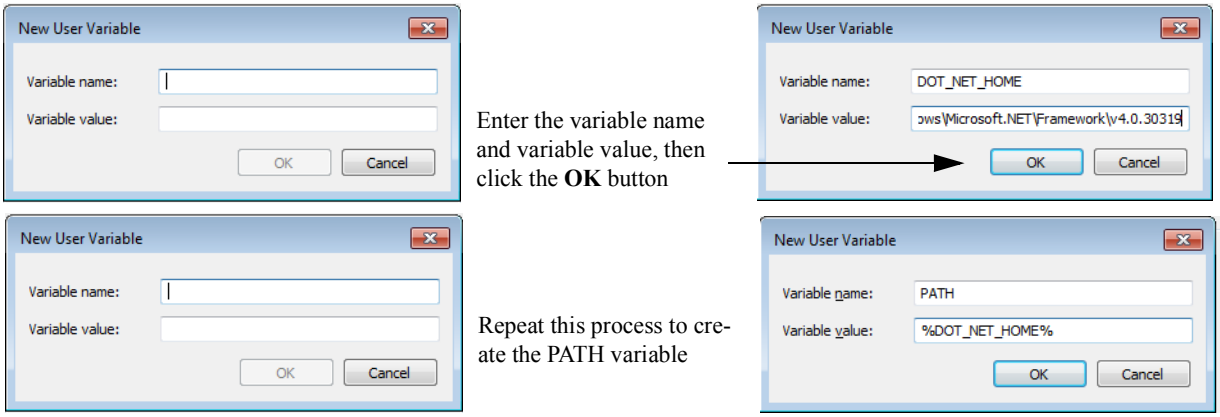


Figure 2-14: Creating New Environment Variables

The Purpose of the Path Environment Variable

The purpose of the PATH environment variable is to instruct the operating system where to find executable files. To compile C# source files from the command line, which I find myself doing frequently without Visual Studio, the operating system needs to know where to find the C# compiler (csc.exe). The new environment variables set above do exactly this.

Testing Environment Variables

To test the environment variables open a command prompt window and enter `csc` at the prompt to invoke the C# compiler. The results should look similar to those shown in figure 2-15.

Referring to figure 2-15 — Entering `csc` at the command line invoked the csc compiler, but because I failed to specify a C# source file, the attempt to compile resulted in an error. No problem. The whole point was to test the PATH variable. On the next command line I demonstrate how to use an environment variable directly. In this case I changed directories from C:\Users\Administrators to C:\Windows\Microsoft .NET\Framework\v4.0.30319 by using the DOT_NET_HOME variable surrounded by percent signs.

 © 2014 Rick Miller — All Rights Reserved

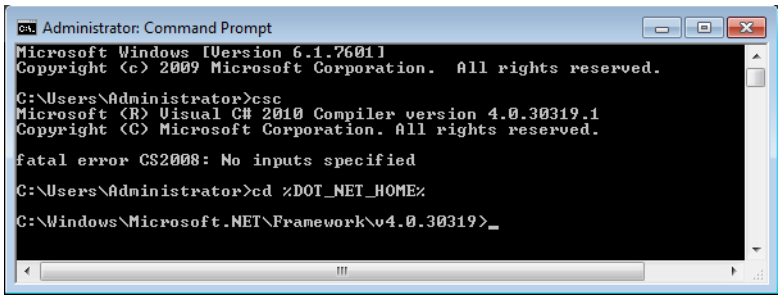


Figure 2-15: Testing Environment Variables

In the next section I want to show you how to adjust your command prompt environment to make it better suited for software development.

Configuring Command Prompt Window

By default, the command prompt window is set to open at a small physical size, a small buffer, and in the user's default or home directory. I like to change these settings to make my command prompts more programmer friendly.

To make these changes either pin a command prompt to the taskbar or create a shortcut on the desktop. Right-click the command prompt shortcut and from the pop-up menu select **Properties** at the bottom. This opens the Command Prompt Properties dialog as is shown in figure 2-16.

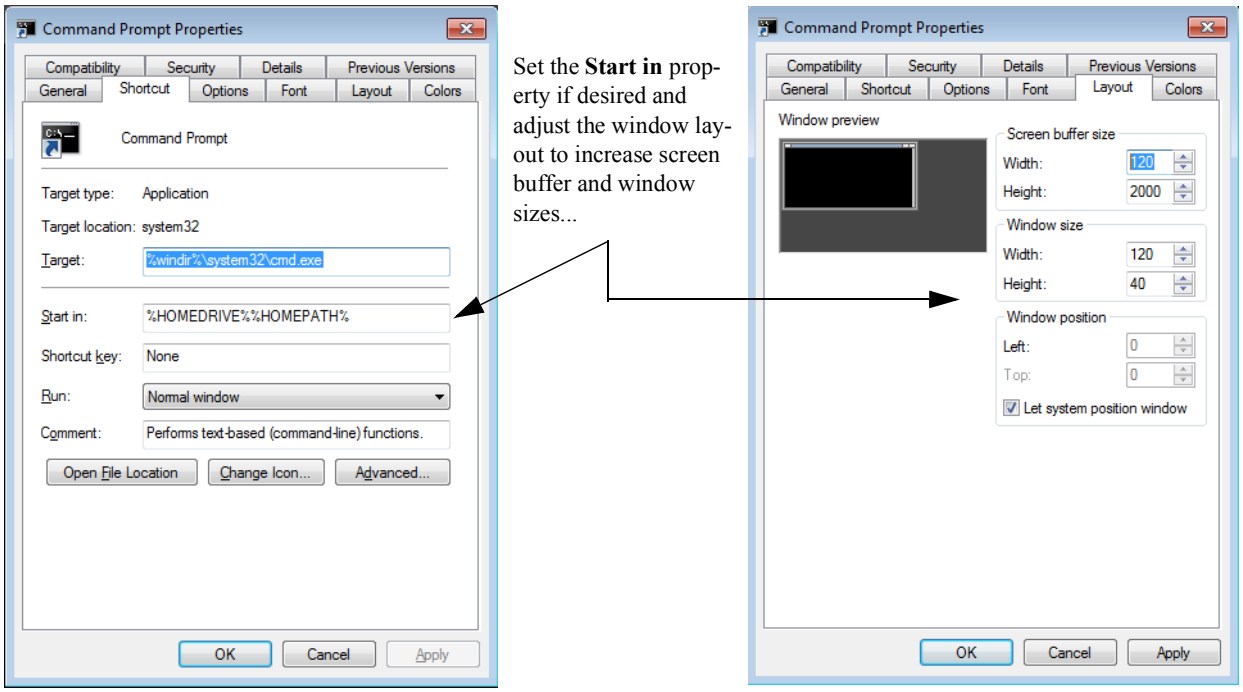


Figure 2-16: Command Prompt Properties Dialog Shortcut and Layout Tabs

Referring to figure 2-16 — The tabs you'll be most interested in initially are the **Shortcut** and **Layout** tabs. Later, on your own, you can experiment with different font and background colors. I like to change the **Start in** property to have the command prompt automatically go to a more convenient directory. This is more than likely a Projects folder I've created on the desktop. To change the **Start in** property simply type in the absolute path to the directory in which you want the command prompt to start in or navigate to that folder, copy the path, and paste it in the text box. This works best if you have a long path and don't want to make a mistake.

 © 2014 Rick Miller — All Rights Reserved

Next, click the **Layout** tab. Increase the width and height of the screen buffer size setting and adjust the window size according to your taste. I like the settings shown in figure 2-16. A large screen buffer height will allow compiler messages to scroll down without being lost. Think of the buffer as the absolute dimensions of the window and the window size as a partial portal view into this larger virtual window.

Click **Apply** to apply accept the settings and open a command prompt window to test things out. You may need to make adjustments to suit your personal preferences.

Installing Required Software

At this point you're ready to start installing the remaining tools and applications. This includes Visual Studio 2010, SQL Server 2008 R2, NUnit, Log4Net, Microsoft Enterprise Library 5.0, Notepad++, Subversion, and TortoiseSVN. As the installation procedures for all of these are straightforward I only want to make some general comments and observations.

Installing Microsoft Visual Studio 2010

You'll be presented with the option to do a Complete or a Custom installation. The primary difference between the two is that the Complete installation installs the whole shebang, including SQL Server 2008 R2 Express edition. If you don't want to install SQL Server Express then chose the Custom install and uncheck the SQL Server Express option.

There's no harm installing both the SQL Server Standard and Express editions on the same machine, and in fact that is what I will do so I can demonstrate the differences between their connection strings and minimum database files sizes. The other differences between these two versions will be virtually transparent and not an issue to the average developer. Well, I take that back. The biggest difference between the Express and Standard editions is that the Express edition does not come with SQL Server Management Studio. There is an Express edition of SQL Server Management Studio that you can download and install separately.

Upon successful installation you'll be presented with the option to install documentation locally. If you have a reliable Internet connection you can safely skip this step and save some hard disk space.

When installation is complete I recommend exploring the Start menu to see what's been added. You should see an application folder for Microsoft SQL Server 2008 (If you chose to install the Express edition.) and another folder for Microsoft Visual Studio 2010. There will also be several more folders: one for the Silverlight3 SDK and Sync Framework.

Go ahead now and create a shortcut to Visual Studio on your desktop and pin it to your Taskbar.

Installing Microsoft SQL Server 2008 R2 Standard Edition

If you have access to Microsoft SQL Server 2008 R2 Standard edition I recommend installing it for testing purposes. Click **New installation or add features to an existing installation** option to begin the installation process.

As you proceed through installation there will be a few things you'll want to write down in your engineer's notebook for future reference. One of the most important bits of information to note is the System Administrator (SA) password.

After the installation process certifies your system for installation and installs required supporting files, you'll be presented with three installation choices. I recommend installing the first choice: **SQL Server Feature Installation**. Click **Next>** to proceed. You'll then be presented with a checklist of features to install. Install everything except for Client Tools Backward Compatibility and SQL Server Books Online. I also accept the default installation location. Click **Next>** to proceed.

Now you'll need to make some decisions that you'll need to note in your engineer's notebook. This is the Instance Configuration dialog window shown in figure 2-17.

 © 2014 Rick Miller — All Rights Reserved

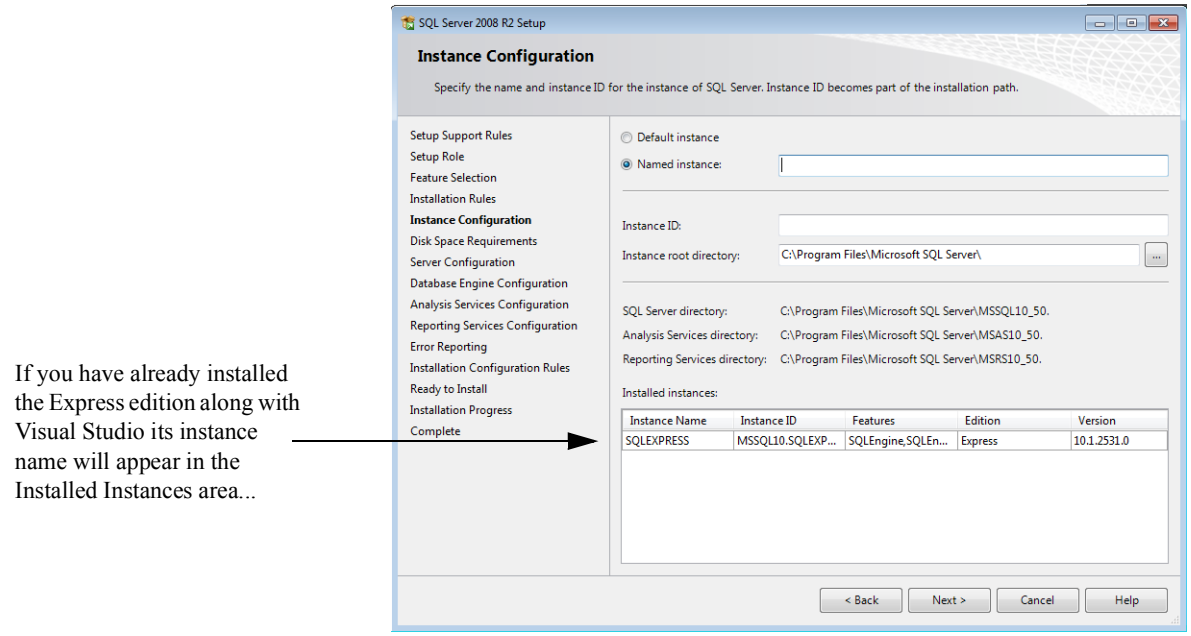


Figure 2-17: SQL Server Instance Configuration Dialog

Referring to figure 2-17 — If you have already installed SQL Server Express edition you'll see it listed in the Installed Instances section. Unless you have a burning desire to give a unique name to the Standard edition instance you are now installing, I recommend you click the **Default instance** radio button at the top of the dialog window and simply note the name of the instance. Click **Next>** to continue.

The next decision you'll need to make concerns under what machine accounts the various SQL Server services will run. For development purposes in a non-production environment you can simplify your life here and have the services run under the same account. Figure 2-18 shows the Server Configuration dialog window.

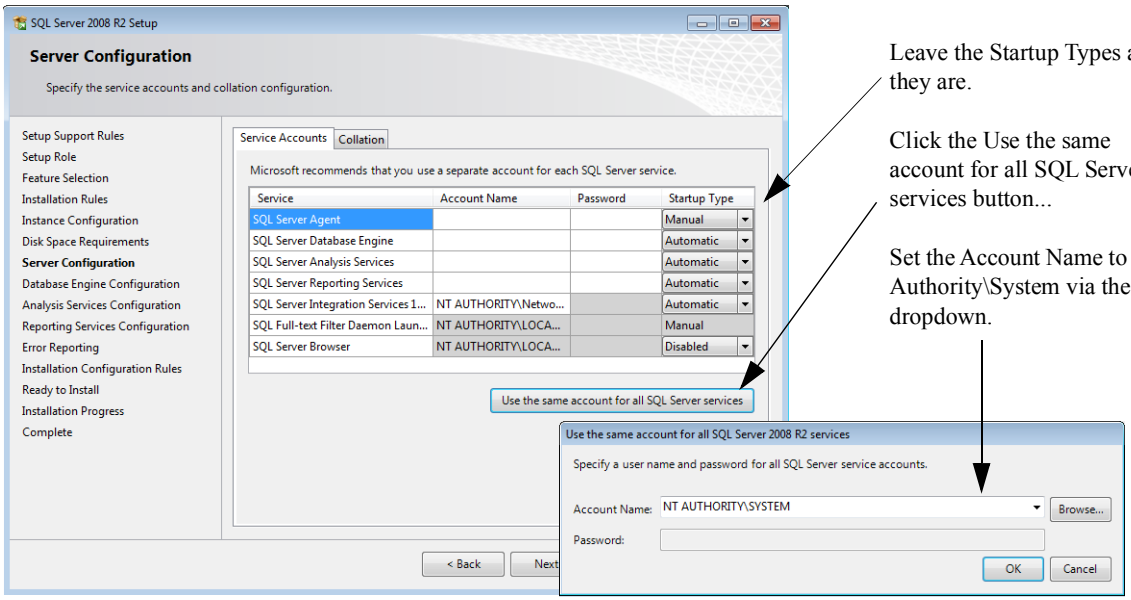


Figure 2-18: Server Configuration Dialog - Accounts Tab

Referring to figure 2-18 — Leave the Startup Types unchanged and click the long button named **Use the same account for all SQL Server Services** to bring up the account settings dialog. Set the Account Name to NT Authority\System. The Password text box will grey out. Click **OK**. Click **Next>** to proceed.

The next dialog window will be the Database Engine Configuration as is shown in figure 2-19.

 © 2014 Rick Miller — All Rights Reserved

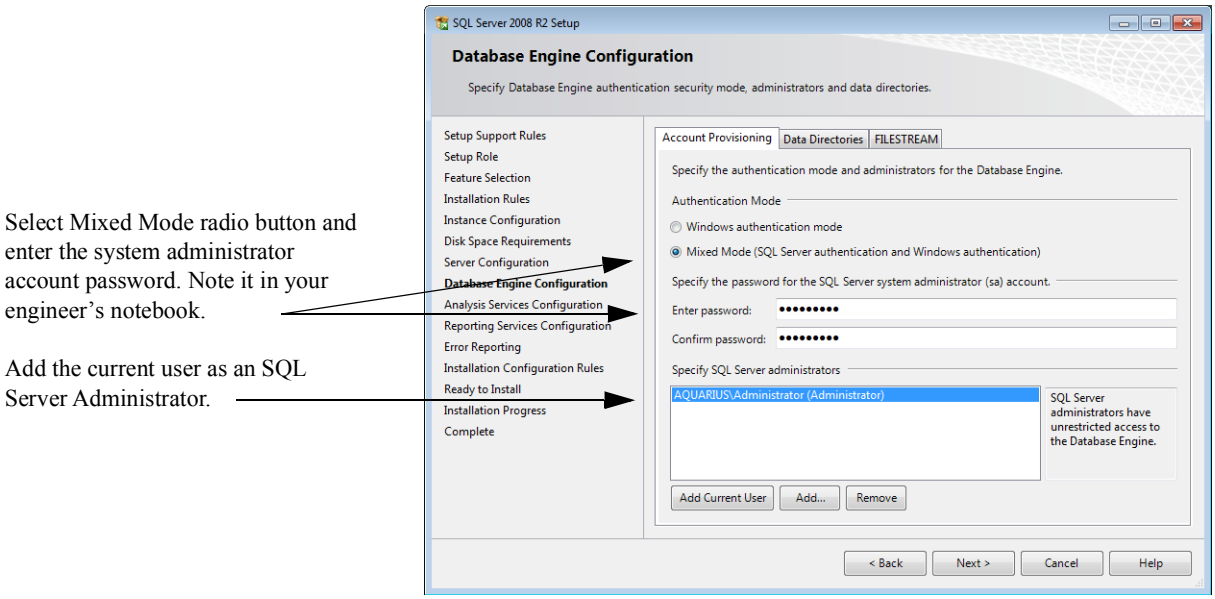


Figure 2-19: Database Engine Configuration Dialog - Account Provisioning Tab

Referring to figure 2-19 — The Database Engine Configuration dialog has three tabs. On the first tab, Account Provisioning, select **Mixed Mode** authentication and enter a password for the system administrator (SA) account. Write this password in your engineer's notebook. Next, add the current user to the SQL Server Administrators area using the **Add Current User** button. You can safely ignore the remaining two tabs for now. Click **Next>** to continue.

In the Analysis Services Configuration dialog add the current user as an administrator and click **Next>** to continue. In the Reporting Services Configuration dialog select the topmost radio button to install the native mode default configuration. Click **Next>** to continue.

Click **Next>** until you arrive at the Ready to Install dialog. Here you'll see a list of all the SQL Server services you are about to install. If you need to make any changes go back to each dialog to adjust as necessary. When you're ready, click the Install button and hang on to your pants.

When installation completes, open and print the Installation Summary Log and save it for future reference. You'll also need to restart your computer for the settings to take effect.

Testing the Installation

To test the installation open Microsoft SQL Server Management Studio and connect to the MSSQLServer instance you just installed. You can also try connecting to the SQLExpress instance.

You'll find SQL Server Management Studio under the SQL Server 2008 application start menu. Click the **Start** button, click on the SQL Server 2008 R2 folder and launch SQL Server Management Studio. You will also want to create a shortcut on the desktop and pin it to the taskbar. To connect to the local instance of SQL Server 2008 R2 use `(local)` in the Server name text box as is shown in figure 2-20.

Referring to figure 2-20 — To login to the SQL Server 2008 R2 Standard instance use the Server Name `(local)` including the parentheses. If you installed both the Standard and Express editions, you can log into the Express instance with the Server Name `(local)\SQLExpress`. Upon successful login you will be presented with the SQL Server Management Studio console window as is shown in figure 2-21.

If you have trouble connecting, review your installation and restart your computer to ensure all required SQL Server services are started properly. When you have successfully logged in to SQL Server Management Studio you can close the application. I'll talk more about SQL Server in Chapter 7: Creating Databases with SQL Management Studio.

© 2014 Rick Miller — All Rights Reserved

Use the Server Name `(local)` to log into SQL Server Standard

Use the Server Name `(local)\SQLExpress` to log into SQL Server Express

Figure 2-20: Logging Into SQL Server and SQL Server Express via Management Studio

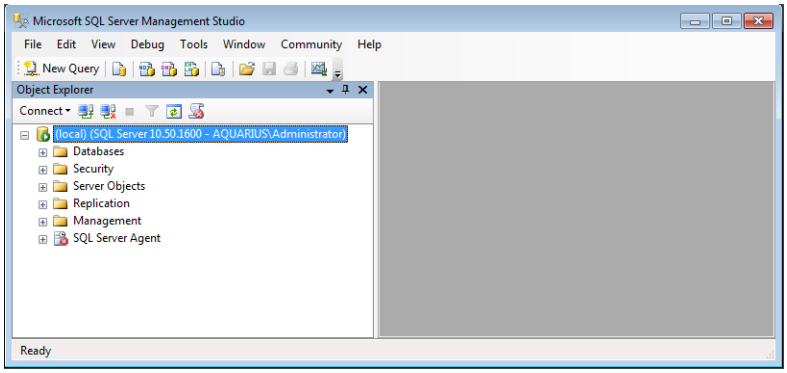


Figure 2-21: SQL Server Management Studio Console

Installing Microsoft Enterprise Library 5.0

To install the Microsoft Enterprise Library 5.0 you first need to download the .msi file from Microsoft's MSDN website. You have two download options: 1) the Enterprise Library 5.0 - Source Code.msi or 2) the Enterprise Library 5.0.msi file. Download the second option.

Installation is quick and painless. When you're asked to launch the source code installer I recommend declining the offer since you won't need the source code for the purposes of this book.

When installation completes all you really need to do is to locate the installation directory and make a note of it in your engineer's notebook for future reference. Eventually you'll need to locate various .dll files and the configuration tool, but for now this is all that's required.

Installing Notepad++

If you haven't already done so, download and install Notepad++. Accept the default values during installation and when finished, create a shortcut on your desktop and pin it to your taskbar.

Installing NUnit

NUnit is the testing framework library. It also comes with a nice little GUI interface to run test suites independently of Visual Studio. This is nice to do on many occasions, especially when tracking down testing errors.

 © 2014 Rick Miller — All Rights Reserved

You'll be presented with three installation choices: Typical, Custom, and Complete. Click the **Complete** button and click **Install** to proceed. Installation is over in the blink of an eye. Make a note of the installation directory in your engineer's notebook for future reference.

Eventually, you'll need to add the path to the NUnit bin directory to your PATH environment variable. I did this by first creating an environment variable named NUNIT_HOME and setting the variable's value to the absolute path to my NUnit bin directory. In my case I installed NUnit 2.6 and the absolute path to the bin directory is: C:\Program Files (x86)\NUnit 2.6\bin. Then, add the NUNIT_HOME variable to the PATH variable value. Separate each path entry with a semicolon. For example, earlier I showed you how to use the DOT_NET_HOME environment variable to set the PATH. Here's how the PATH variable value would look after you add the NUNIT_HOME variable:

```
%DOT_NET_HOME%;%NUNIT_HOME%
```

If you don't do this now you'll have to do it later before you can execute unit tests automatically as part of the Visual Studio build process.

Finally, navigate to the NUnit bin directory and locate the file named nunit.exe. This is the NUnit GUI console. Create a shortcut to it on your desktop and pin it to your taskbar.

Installing Log4Net

Download the latest version of Log4Net. You'll be downloading a zip file. The latest edition at the time of this writing is log4net-1.2.11-bin-newkey.zip. When download completes open the zip file and drag the folder into the Program Files (x86) folder. For now this is all you need to do. I'll talk more about how to configure Log4Net when I discuss unit testing in Chapter 9: The Infrastructure Layer.

Installing NAnt

Next, go to http://nant.sourceforge.net/ and look for the latest NAnt release. Download the Zip file to your desktop. The latest version of NAnt at the time of this writing is NAnt 0.92.

IMPORTANT: To properly unzip NAnt-0.92 perform the following procedures. If you fail to unzip NAnt-0.92 correctly, you'll encounter a rather cryptic log4net error when you attempt to execute NAnt.

To unzip, first right-click the NAnt-0.92-bin.zip file and click **Properties**. This will bring up the Properties dialog as is shown in figure 2-22.

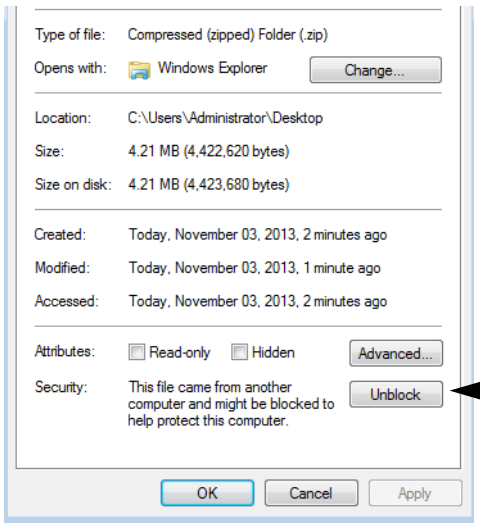


Figure 2-22: Properly Unzipping NAnt

Referring to figure 2-22 — At the bottom of the Properties dialog you'll see an **Unblock** button next to the Security section. Click the **Unblock** button then click the **Apply** button and then the **OK** button to dismiss the dialog. Next, double-click the NAnt zip file and drag the NAnt-0.92 folder to the `C:\Program Files (x86)` folder.

© 2014 Rick Miller — All Rights Reserved

Installing Subversion and TortoiseSVN

Lastly, you'll need to install and configure Subversion and TortoiseSVN. These are two separate tools available from two separate sites. While not strictly necessary for personal development, I recommend getting familiar with source code version control early in your career. I will talk more about installing and configuring these two tools in Chapter 3: Configuration Management with Subversion.

Summary

In this chapter I offered several tips on how to configure your development environment to increase your efficiency and productivity. I also discussed the benefits.

Upon completion of this chapter you should have installed all your development tools with the exception of Subversion and TortoiseSVN. I also recommended that you create shortcuts and pin icons to frequently used applications on your desktop and taskbar. Figure 2-23 shows the final arrangement of my Windows 7 development environment.

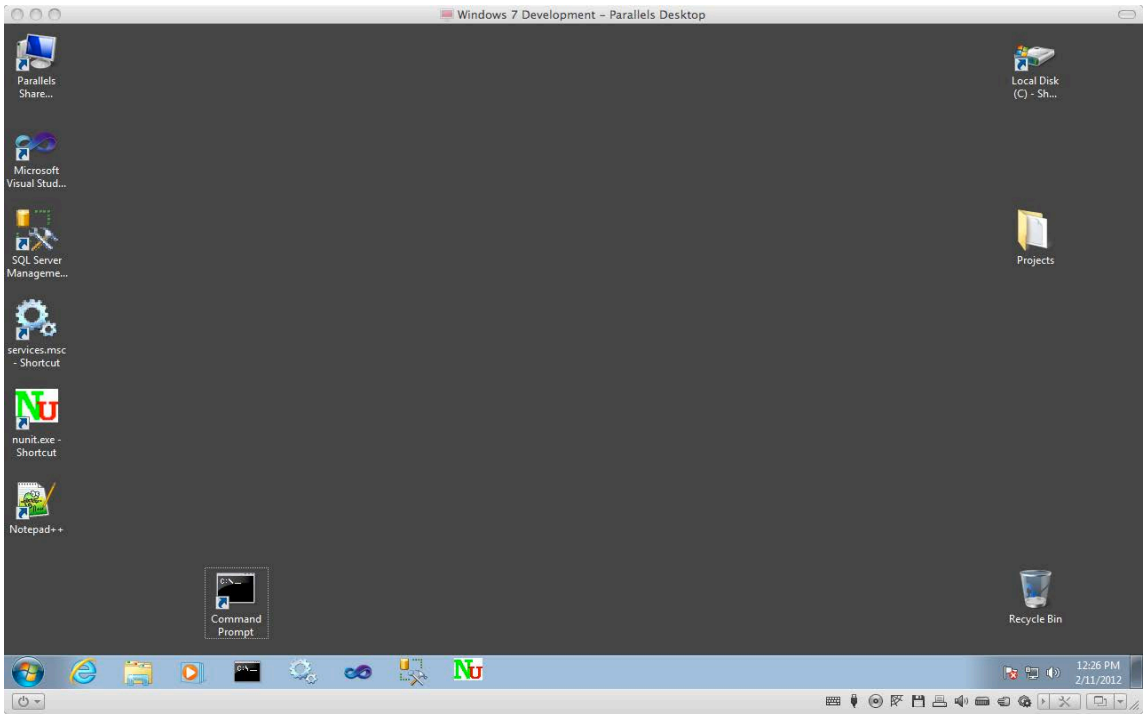


Figure 2-23: Final Arrangement of my Windows 7 Development Environment

Referring to figure 2-23 — Note that I tend towards a rather Spartan arrangement. I line up my application icons along the left side of the desktop. I put the Local Disk (C) shortcut in the upper-right corner and the Recycle Bin in the lower-right corner. I do this so I don't have to move or resize windows to access these frequently used icons. This is simply my preference. You may do things completely differently.

References

Microsoft Developer Network Online Documentation, MS SQL Server 2008 R2 Library, http://msdn.microsoft.com/en-us/library/bb545450.aspx

 © 2014 Rick Miller — All Rights Reserved

Microsoft Developer Network Online Documentation, Enterprise Library, http://msdn.microsoft.com/en-us/library/ff648951.aspx

Notepad++ Website http://notepad-plus-plus.org/

NUnit Website http://www.nunit.org/

Subversion Website http://subversion.tigris.org/

TortoiseSVN Website http://tortoisesvn.net/

NAnt Website http://nant.sourceforge.net/

Notes

 © 2014 Rick Miller — All Rights Reserved

 © 2014 Rick Miller — All Rights Reserved