

CHAPTER 3

ASP.NET Compilation Models

IN THIS CHAPTER

ASP.NET Compilation 32

Automating `aspnet_compiler.exe` in Visual Web Developer Express Edition 37

ASP.NET Compilation

In the previous chapter, I covered the basics of ASP.NET code models. In this chapter, we'll discuss the details of how ASP.NET applications are compiled. This information is not vital to your success as an ASP.NET developer, but having an understanding of the architecture of your development environment always makes you a better developer.

ASP.NET is nothing like the legacy ASP with which many developers are familiar. You develop ASP pages by using VBScript or JScript, and they are interpreted, meaning that they are executed just as they are written, directly from the page. ASP.NET is entirely different in that ASP.NET pages are compiled before they are executed.

When you write ASP.NET code, you do so in human-readable text. Before ASP.NET can run your code, it has to convert it into something that the computer can understand and execute. The process of converting code from what a programmer types into what a computer can actually execute is called *compilation*.

Exactly how compilation takes place in ASP.NET depends on the compilation model that you use. Several different compilation models are available to you in ASP.NET 3.5.

The Web Application Compilation Model

The web application compilation model is the same model provided in ASP.NET 1.0 and 1.1. When you use this model, you use the Build menu in Visual Web Developer to compile your application into a single DLL file that is copied to a bin folder in the root of your application. When the first request comes into your application, the DLL from the bin folder is copied to the Temporary ASP.NET Files folder, where it is then recompiled into code that the operating system can execute in a process known as *just-in-time (JIT)* compilation. The JIT compilation causes a delay of several seconds on the first request of the application.

NOTE

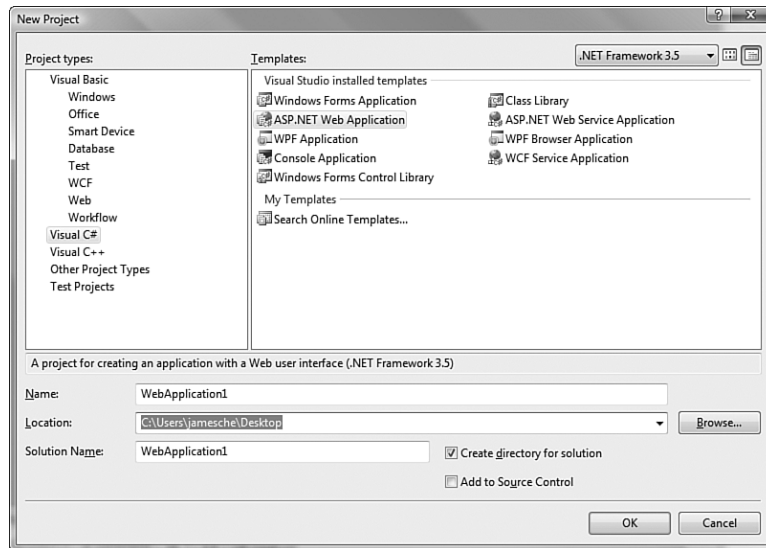
The web application model is available only in Visual Studio 2008. Visual Web Developer 2008 does not enable you to create ASP.NET applications using the web application model.

NOTE

The Temporary ASP.NET Files folder is located at Windows\Microsoft.NET\Framework\v2.0.50727\Temporary ASP.NET Files by default.

To create a new ASP.NET web application using the web application compilation model, select File, New Project, and then choose the ASP.NET Web Application template as shown in Figure 3.1.

FIGURE 3.1
Choose the New Project option on the File menu to create a new ASP.NET application that uses the web application compilation model.



The Website Compilation Model

The website compilation model is the model that developers using Visual Web Developer Express Edition will use because it's the only model available. In this model, all files in the application are copied to the remote web server and are then compiled on the fly by ASP.NET at browse time.

NOTE

The website compilation model was the only compilation model available when Visual Studio 2005 (the previous version of Visual Studio) was released. Microsoft added the web application model later as an add-on to Visual Studio 2005, and then incorporated it into Visual Studio 2008.

NOTE

You can use the website compilation model whether you are using inline ASP.NET code or code-behind code.

→ For more information on inline code and code-behind code models, see "Server-Side Code Models," p. 24.

When this compilation model is used, ASP.NET compiles the application into one or more DLLs in the Temporary ASP.NET Files folder when the first page is requested. The DLLs are in a subfolder with a name derived from a special naming convention that allows for dynamic and random directory names. Therefore, a website called MyWebSite might execute from a folder on the server that looks similar to this:

```
C:\Windows\Microsoft.NET\Framework\v2.0.50727\Temporary ASP.NET
Files\MyWebSite\650b10f9\e47ff097
```

Inside that folder will be the actual DLLs that contain the compiled code. The naming convention of the DLLs is `App_Web_<random_name>.dll`.

NOTE The compilation of the App_Web DLLs takes place without any explicit action on your part. It's all automatic.

The website compilation model is convenient because developers can open a code file or an ASPX page and make modifications to it on the live server. When those changes are saved, they go into effect immediately. However, using this method requires you to copy all the source code for your application to the live server, and this may be a concern to some developers.

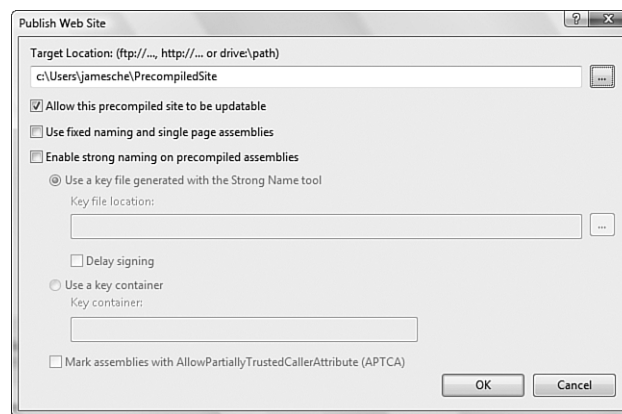
CAUTION ASP.NET explicitly forbids the download of code files from a website, so no one will be able to download your source code and access it. However, anyone with direct access to the web server can access your source code when using the website compilation model.

The Precompilation Model

The precompilation model allows you to compile your ASP.NET application into one or more DLLs that can then be copied to the web server in place of any code.

Select Build, Publish Web Site to precompile your website using the Publish Web Site dialog shown in Figure 3.2.

FIGURE 3.2
The Publish Web Site dialog makes pre-compiling a website simple.



NOTE The Publish Web Site menu option is available only in the full Visual Studio version. It is not available in Visual Web Developer Express Edition, but you can add the capability using the steps provided in the “Automating `aspnet_compiler.exe` in Visual Web Developer Express Edition” section later in this chapter.

If you'd like the option of updating any of your ASPX pages on the live server (for example, making a change to the HTML code), you should check the Allow This Precompiled Site to be Updatable check box. If your site is precompiled with this check box checked, you'll then be able to make modifications to the ASPX pages on the live server if necessary. If it is unchecked, you'll still need to copy the ASPX files to the server, but the precompilation process will remove all the code from them, so you won't be able to change any of them on the live server. We'll talk about that in greater detail a little later in this chapter.

If you are using Visual Web Developer Express Edition, you won't have the option to pre-compile your website within the user interface, but it can still be accomplished if you use the `aspnet_compiler.exe` utility that ships with the .NET Framework.

NOTE

The `aspnet_compiler.exe` utility is located in the `Windows\Microsoft.NET\Framework\v2.0.50727` directory.

The `aspnet_compiler.exe` utility runs from a command line. If you have the .NET Framework SDK v2.0 installed, you can select Start, All Programs, Microsoft.NET Framework SDK v2.0 and then click SDK Command Prompt to open a command prompt. This command prompt automatically sets the necessary environment variables to enable you to run `aspnet_compiler.exe` without changing into the `v2.0.50727` directory.

NOTE

You can download the .NET Framework SDK v2.0 from www.microsoft.com/downloads/details.aspx?familyid=fe6f2099-b7b4-4f47-a244-c96d69c35dec&displaylang=en. If you'd prefer not to type in that long URL, search on .NET Framework SDK 2.0 and you'll find it.

If you don't have the .NET Framework SDK v2.0 installed, you can still run `aspnet_compiler.exe` from a regular command line, but you need to change into the `v2.0.50727` directory first. You can do that by running the following command from a command prompt:

```
cd \windows\microsoft.net\framework\v2.0.50727
```

- For information on how to configure a menu item in Visual Web Developer Express Edition that will pre-compile your web application, see "Automating `aspnet_compiler.exe` in Visual Web Developer Express Edition," later in this chapter.

Numerous command-line parameters can be used with `aspnet_compiler.exe`. Table 3.1 lists a few that you are likely to use often.

Table 3.1 Frequently Used Parameters for `aspnet_compiler.exe`

Parameter	Description
-?	Prints a description of all parameters.
-v	Specifies the path that follows is a virtual path.
-p	The physical path of the application to compile.
-u	Specifies the compiled application can be updated.
-c	Causes the compiled application to be fully rebuilt, overwriting any existing files.
-d	Creates debug output, making the application easier to debug if a problem arises after it's copied.

Even though there are a lot of parameters for `aspnet_compiler.exe`, the command to precompile your ASP.NET application is less complex than you might think. For example, if you have an ASP.NET application located at `c:\myApp` and you want to precompile it and save the result to `c:\compiledApp`, you would run the following command:

```
aspnet_compiler.exe -p "c:\myApp" -v / "c:\compiledApp"
```

The `-p` parameter is used to point to the location of the application (`c:\myApp` in this case), and the `-v` parameter points to the virtual location of the application. In a file-based ASP.NET application, the virtual location is always `/`. That is followed by the path where the compiled files should be written.

If you look in the `c:\compiledApp` directory after this command runs, you can see what looks like the ASPX files for your application, but in fact, these are simply marker files. If you open one of them, you'll see a single line of text in it that says *This is a marker file generated by the precompilation tool, and should not be deleted!* ASP.NET creates this file so that, when the application is browsed on the live server, users won't get an error saying that the file wasn't found.

You'll also see a file called `preCompiledApp.config`. This file contains the version number of the precompilation tool (for ASP.NET 3.5, it is version 2) and specifies whether the site is capable of being updated. If your website has any other configuration files (such as a `web.config` file), it is also in the directory containing the precompiled application, along with any other supporting files and folders such as images and so on.

→ For more information on `web.config` files, see Chapter 6, "ASP.NET Configuration and Performance."

All the code for your application is compiled into a `bin` directory located at the root of the precompiled site. If you open that directory, you'll see one or more DLLs with names such as `App_Web_ekxytkat.dll`. All these DLLs start with `App_Web_` and then contain a random group of characters. These DLLs are ASP.NET assemblies, and when your website runs on the live server, it runs from these DLLs.

To publish a precompiled web application to a live web server, simply copy all the files and folders in the directory containing the precompiled website to the live server. When you do, ASP.NET automatically begins using the new files.

Automating aspnet_compiler.exe in Visual Web Developer Express Edition

As I pointed out previously, there is a menu option in Visual Studio 2008 that automates the use of the `aspnet_compiler.exe` so that you can easily precompile your applications. That menu option does not exist in Visual Web Developer Express Edition, but you can easily add it to the menu by configuring `aspnet_compiler.exe` as an external tool in Visual Web Developer Express Edition.

To configure menu options for precompiling your web application in Visual Web Developer Express Edition, follow these steps:

1. Launch Visual Web Developer Express Edition.
2. Select Tools, External Tools to access the External Tools dialog.
3. Type **Pre-&Compile (non-updatable)** in the Title text box.

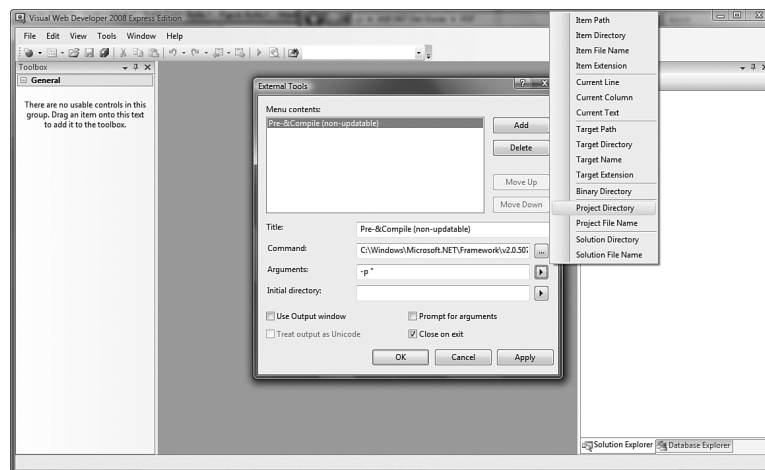
TIP

The ampersand in the title means that the character immediately after will be defined as a hotkey for the menu item.

4. Click the browse button next to the Command text box and browse to `aspnet_compiler.exe` in the `Windows\Microsoft.NET\Framework\v2.0.50727` directory.
5. Click Open to select the `aspnet_compiler.exe` application.
6. Type `-p "` in the Arguments text box.
7. Click the right-facing arrow button on the right edge of the Arguments text box and select Project Directory, as shown in Figure 3.3.

FIGURE 3.3

You configure the arguments for `aspnet_compiler.exe` by using the menu to the right of the Arguments text box.

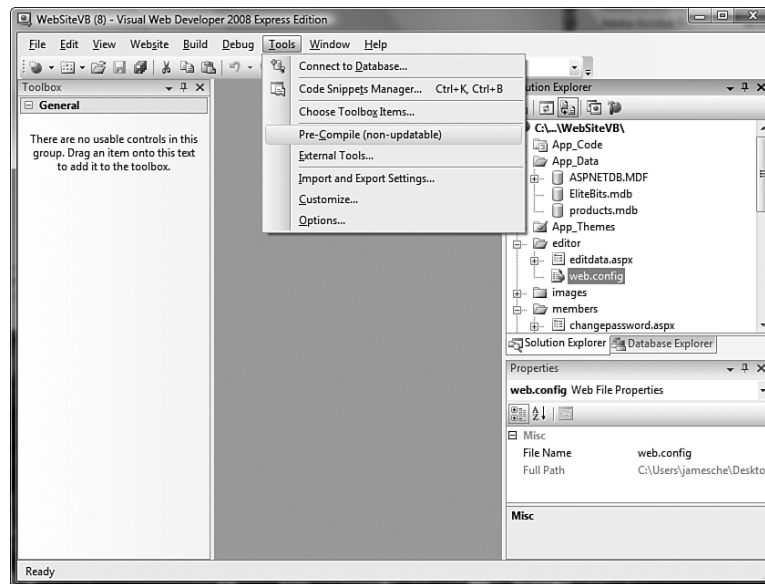


8. Enter a closing quote at the end of the arguments you've entered so far. At this point, the Arguments text box should contain the following text:
-p "\$(ProjectDir)"
9. Press the spacebar to add a space at the end of the arguments you've entered so far, and then type -v / ".
10. Click the right-facing arrow button at the right edge of the Arguments text box again and select Project Directory.
11. Type `..\CompiledApp` after the existing arguments. At this point, the Arguments text box should contain the following text:
-p "\$(ProjectDir)" -v / "\$(ProjectDir)\..\Compiled"
12. Check the Close on Exit check box.
13. Click OK to add the new external tool to the list.

You now have a new menu option in Visual Web Developer Express Edition (see Figure 3.4) that enables you to precompile your web application as a non-updatable application.

FIGURE 3.4

Adding a menu option in Visual Web Developer Express Edition to automate the `aspnet_compiler.exe` utility is simple and makes it much more convenient to precompile your applications.



Now let's add a second external tool that precompiles a website and makes it updatable.

1. Select Tools, External Tools to access the External Tools dialog.
2. Click Add to add a new external tool.
3. Enter **Pre-Co&mpile (updatable)** in the Title text box.

4. Configure all other settings as you did before, but this time, add the **-u** argument to the Arguments text box. The arguments text box should contain the following text:

```
-p "$(ProjectDir)" -v / "$(ProjectDir)\..\Compiled" -u
```

When you select one of the new menu items, your ASP.NET application is precompiled into a directory called Compiled. That directory is one level above your ASP.NET application. Therefore, if your ASP.NET application is located at `c:\mysites\myWebApp`, the precompiled website is saved to `c:\mysites\Compiled`.