第 3 章

网页文件夹

3.1 App_Browsers文件夹——浏览器文件夹

App_Browsers 文件夹包含 ASP.NET 用于标识个别浏览器并确定其功能的浏览器定义（.browser）文件。

放置到 App_Browsers 文件夹中的.browser 文件，即浏览器定义文件包含各个浏览器的定义。在运行时，ASP.NET 使用请求标头中的信息来确定发出请求的浏览器的类型。随后，ASP.NET 使用.browser 文件来确定浏览器的功能，以及如何向该浏览器呈现标记。

> 说明 对于希望创建可以在移动设备上查看的应用程序的 Web 开发人员，.browser 文件很有用。因为这样可以利用控件适配器根据设备类型改编 ASP.NET Web 服务器控件的行为。

示例 本示例实现的是在应用程序中创建一个 App_Browsers 文件夹，并在该文件夹中添加一个浏览器定义文件，即.browser 文件。

首先，右击应用程序的名称，在弹出的快捷菜单中选择“添加 ASP.NET 文件夹”→“App_Browsers”命令，便可创建一个 App_Browsers 文件夹，如图 3.1 所示。

创建完 App_Browsers 文件夹后，右击该文件夹，在弹出的快捷菜单中选“添加新项”命令，选择“浏览器文件”图标，即可在 App_Browsers 文件夹中添加一个浏览器文件，如图 3.2 所示。

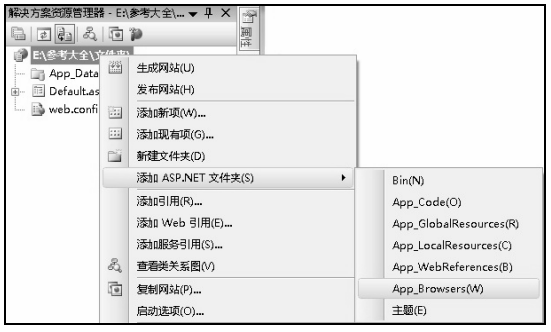

图 3.1 选择相应命令

图 3.2 添加浏览器文件

3.2 App_Code文件夹——公共类文件夹

可以在 App_Code 文件夹中存储源代码，在运行时将会自动对这些代码进行编译。Web 应用程序中的其他

任何代码都可以访问产生的程序集。在这个文件夹中，用户可以存储源代码，而非已编译的代码，因此用户可以创建自定义类和其他源代码文件，并在 Web 应用程序中使用它们，而不必单独对它们进行编译。

App_Code 文件夹内可包含用户希望作为应用程序一部分进行编译的实用工具类和业务对象（例如.cs 和.vb 文件）的源代码。在动态编译的应用程序中，当对应用程序发出首次请求时，ASP.NET 编译 App_Code 文件夹中的代码，然后在检测到任何更改时重新编译该文件夹中的项。

> **注意** 在 App_Code 文件夹中不允许使用用户控件。这包括单文件用户控件及使用代码隐藏模型的用户控件。将用户控件置于 App_Code 目录中，会导致不按用户控件代码所要求的顺序对其进行编译，因此是不允许的。请注意，不需要将用户控件置于 App_Code 文件夹中，因为处于应用程序中任何位置的页都已经可以使用这些控件。

关于 App_Code 文件夹还需要理解以下 3 方面内容，下面分别介绍如下：

- 推断 App_Code 文件夹的编程语言。

App_Code 文件夹并未显式标记为包含以任何一种编程语言编写的文件。相反，ASP.NET 是根据 App_Code 文件夹所包含的文件来推断应为 App_Code 文件夹调用哪一种编译器。如果 App_Code 文件夹包含.vb 文件，则 ASP.NET 使用 VisualBasic 编译器；如果包含.cs 文件，则 ASP.NET 使用 C#编译器，以此类推。

- 在 App_Code 文件夹中使用多种编程语言。

因为 App_Code 文件夹中的源代码要编译成单个程序集，所以 App_Code 文件夹中的所有文件必须使用相同的编程语言编写。例如，App_Code 文件夹不能同时包含采用 Visual Basic 和 C#编写的源代码。

- App_Code 文件夹的安全性。

App_Code 文件夹中的代码会在运行时编译成程序集，这比 Bin 文件夹要好一些的是，用户可以查看 App_Code 文件夹中文件的源代码。但是，如果用户不能完全理解代码，仍然会存在安全风险。因此，对待 App_Code 文件夹中源代码的态度，必须像对待基于同样的源代码生成的已编译代码一样谨慎。

示例　下面以在网站中创建公共类为例，讲解 App_Code 文件夹的应用。

在 Visual Studio 2008 中新建一个应用程序，将其命名为 MyText，右击应用程序名称，在弹出的快捷菜单中选择“添加新项”命令，然后在弹出的对话框中选择“类”图标，将这个类命名为 MyClass.cs，如图 3.3 所示。

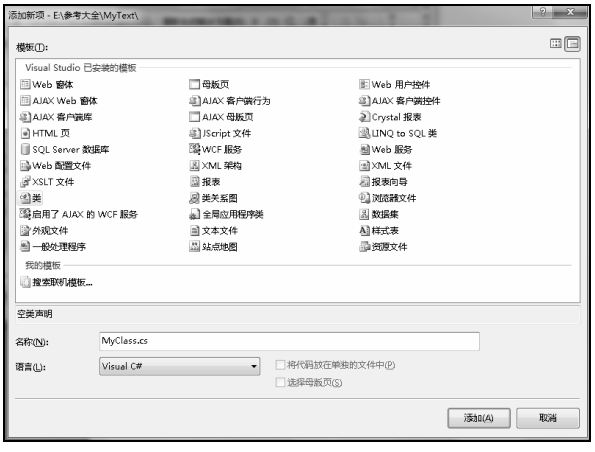

图 3.3　创建公共类 MyClass.cs

当单击图 3.3 中的“ 添加 ”按钮时，将会弹出如图 3.4 所示的对话框，单击对话框中的“ 是 ”按钮，应用程序将会自动创建 App_Code 文件夹，如图 3.5 所示。

图 3.4 单击“是”按钮

图 3.5 在应用程序中创建的 App_Code 文件夹

除了上述方法可以自动创建 App_Code 文件夹外，还可以在应用程序中手动添加，右击网站名称，在弹出的快捷菜单中选择“ 添加 ASP.NET 文件夹 ”命令，如图 3.6 所示，然后选择“ App_Code ”命令，便可手动添加一个 App_Code 文件夹。

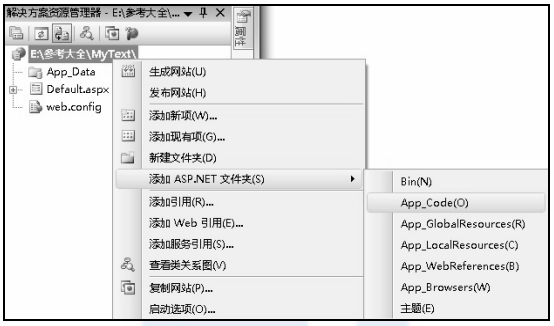

图 3.6 手动添加 App_Code 文件夹

3.3 App_Data文件夹——数据库文件夹

App_Data 文件夹包含应用程序数据文件，包括 MDF 文件、XML 文件和其他数据存储文件。

> 说明 ASP.NET 使用 App_Data 文件夹来存储应用程序的本地数据库，该数据库可用于维护成员资格和角色信息。

示例 在 Visual Studio 开发环境中，创建一个名为 MyText 的应用程序，在创建的应用程序中会自动附带一个 App_Data 文件夹，在该文件中可以放置应用程序应用的数据库文件、XML 文件等。右击 App_Data 文件夹，在弹出的快捷菜单中选择“ 添加现有项 ”命令，在弹出的对话框中选择要添加文件，即可成功添加文件，如图 3.7 所示。

图 3.7 在 App_Data 文件夹中添加文件

3.4　App_GlobalResources文件夹——全局资源文件夹

App_GlobalResources 文件夹包含编译到具有全局范围的程序集中的资源（.resx 和.resources 文件）。App_GlobalResources 文件夹中的资源是强类型的，可以通过编程方式进行访问。

将资源文件放入应用程序根目录的保留文件夹 App_GlobalResources 中，便可创建全局资源文件。App_GlobalResources 文件夹中的任何.resx 文件都具有全局范围。此外，ASP.NET 还会生成一个强类型对象，从而提供了一种以编程方式访问全局资源的简便方法。

> 说明　在 ASP.NET 中，可以创建具有不同范围的资源文件。可以创建全局资源文件（放在 App_GlobalResources 文件夹中），这意味着可以从位于网站中的任意页或代码读取这些资源文件。也可以创建本地资源文件（放在 App_LocalResources 文件夹中，稍后会进行介绍），这些文件存储单个 ASP.NET 网页（.aspx 文件）的资源。

示例　本示例将实现如何创建一个 App_GlobalResources 文件夹，并在 App_GlobalResources 文件夹中添加一个全局资源文件。

首先，右击应用程序的名称，在弹出的快捷菜单中选择“ 添加 ASP.NET 文件夹 ”→“ App_GlobalResources ”命令，便可创建一个 App_GlobalResources 文件夹，如图 3.8 所示。

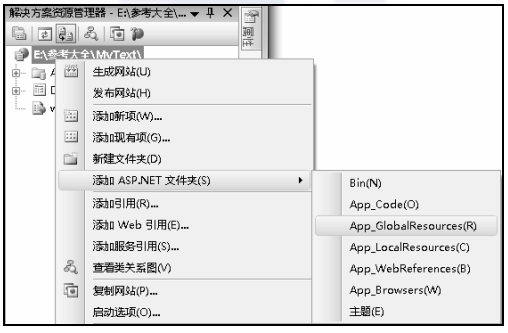

图 3.8　创建 App_GlobalResources 文件夹

创建完 App_GlobalResources 文件夹后，右击该文件夹，在弹出的快捷菜单中选“ 添加新项 ”命令，会弹出一个添加新项的对话框，如图 3.9 所示。在该对话框中选择“ 资源文件 ”选项，单击“ 添加 ”按钮，即可在 App_GlobalResources 文件夹添加一个全局资源文件，如图 3.10 所示。

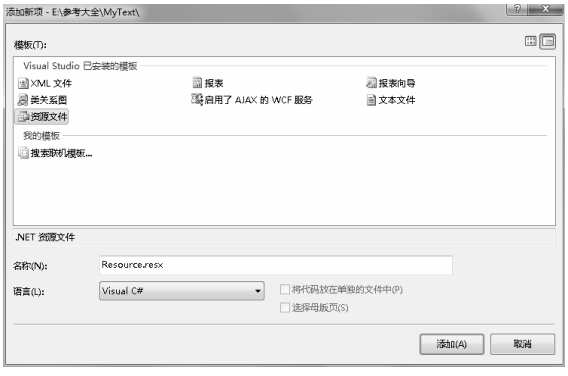

图 3.9　添加新项

图 3.10　添加资源文件

3.5 App_LocalResources文件夹——本地资源文件夹

App_LocalResources 文件夹称为本地资源文件夹，可包含与应用程序中的特定页、用户控件或母版页关联的资源（.resx 和.resources 文件）。

存放本地资源文件的文件夹具有 App_LocalResources 保留名称，它与根 App_GlobalResources 文件夹不同，App_LocalResources 文件夹可以位于应用程序的任意文件夹中，通过使用资源文件的名称，可以将一组资源文件与特定的网页关联起来。

例如，如果在 App_LocalResources 文件夹中有一个名为 Default.aspx 的页，则可以创建下列文件：

- Default.aspx.resx 是未找到语言匹配项时的默认本地资源文件（回退资源文件）。
- Default.aspx.es.resx 是西班牙语的资源文件，其中不包含区域性信息。
- Default.aspx.es-mx.resx 是专用于西班牙语（墨西哥）的资源文件。
- Default.aspx.fr.resx 是法语的资源文件，其中不包含区域性信息。

注意 文件的基名称与页文件名相同，后跟语言和区域性名称，最后以扩展名.resx 结尾。

示例 本示例将实现如何创建一个 App_LocalResources 文件夹，并在 App_LocalResources 文件夹中添加一个 XML 文件。

首先，右击应用程序的名称，在弹出的快捷菜单中选择“ 添加 ASP.NET 文件夹 ”→“ App_LocalResources ”命令，便可创建一个 App_LocalResources 文件夹，如图 3.11 所示。

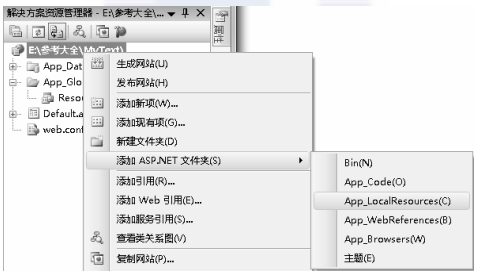

图 3.11 创建 App_LocalResources 文件夹

创建完 App_LocalResources 文件夹，右击该文件夹，在弹出的快捷菜单中选择“ 添加新项 ”命令，会弹出一个添加新项的对话框，如图 3.12 所示。在该对话框中选择“ XML 文件 ”选项，单击“ 添加 ”按钮，即可在 App_LocalResources 文件夹中添加一个 XML 文件，如图 3.13 所示。

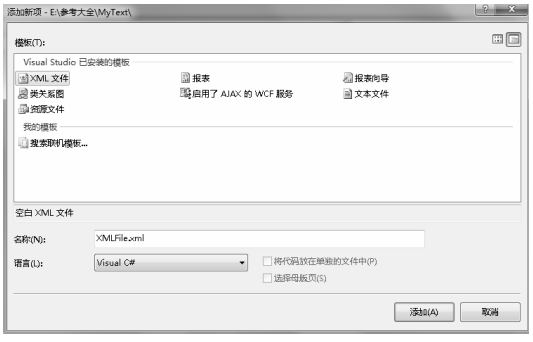

图 3.12 添加新项

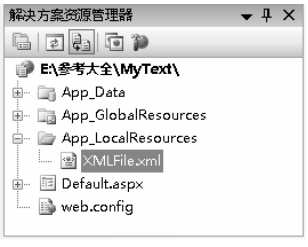

图 3.13 在 App_LocalResources 文件夹中添加一个 XML 文件

3.6　App_Themes文件夹——主题文件夹

App_Themes 文件夹包含用于定义 ASP.NET 网页和控件外观的文件集合（.skin 和.css 文件及图像文件和一般资源）。

主题是属性设置的集合，使用这些设置可以定义页面和控件的外观，然后在某个 Web 应用程序中的所有页、整个 Web 应用程序或服务器上的所有 Web 应用程序中一致地应用此外观，由此可见创建网站的 App_Themes 文件夹也非常重要。

示例　下面以创建一个 TextBox 控件外观文件 SkinFile.skin（控件外观文件）为例，来了解 App_Themes 文件夹的应用。

右击应用程序中的网站名，在弹出的快捷菜单中选择“ 添加新项 ”命令，在弹出的对话框中选择一个主题的附属文件，如选择“ 外观文件 ”选项，如图 3.14 所示。

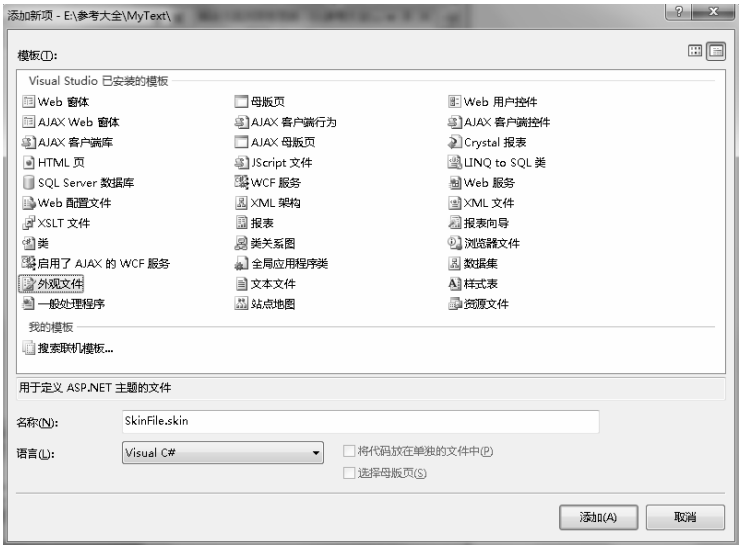

图 3.14　向一个 ASP.NET 主题添加一个附属文件

完成上步操作后，再单击“ 添加 ”按钮，此时将会弹出如图 3.15 所示的对话框，用户只需单击“ 是 ”按钮，即可创建一个 App_Themes 文件夹。

图 3.15　应用 Visual Studio 自动创建一个 App_Themes 文件夹

除了上述创建主题文件夹的方法外，还可以手动添加，右击应用程序中的网站名，在弹出的快捷菜单中选择“ 添加 ASP.NET 文件夹 ”命令，然后选择“ 主题 ”命令，便可在应用程序中添加一个主题文件夹，如图 3.16 所示。

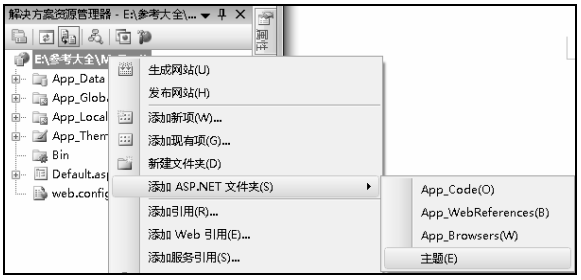

图 3.16 手动添加主题文件夹 App_Themes

3.7 App_WebReferences文件夹——Web引用文件夹

App_WebReferences 文件夹包含用于定义在应用程序中使用的 Web 引用的引用协定文件(.wsdl 文件)、架构（.xsd 文件）和发现文档文件（.disco 和.discomap 文件）。

示例 本示例实现的是如何创建一个 App_WebReferences 文件夹，并通过添加引用方式引用协定文件（.wsdl 文件），具体实现步骤如下：

首先,右击应用程序的名称,在弹出的快捷菜单中选择“ 添加 ASP.NET 文件夹 ”→“ App_WebReferences ”命令，便可创建一个 App_WebReferences 文件夹，如图 3.17 所示。

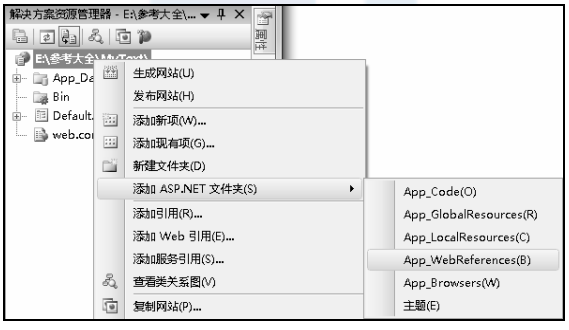

图 3.17 创建 App_WebReferences 文件夹

手动添加完 App_WebReferences 文件后，如要将编译好的 XMLFile.wsdl 文件添加到 App_WebReferences 文件夹中，可右击手动创建的这个 App_WebReferences 文件夹，在弹出的快捷菜单中选择“ 添加 Web 引用 ”命令，如图 3.18 所示，即可完成通过添加引用类库文件添加 XMLFile.wsdl 文件的操作。

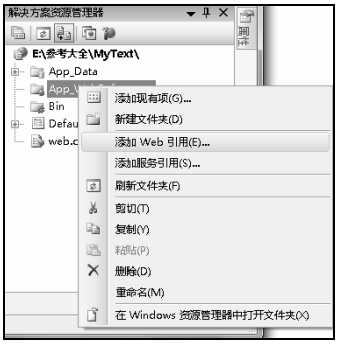

图 3.18 通过添加 Web 引用添加 XMLFile.wsdl 文件

> **说明** .wsdl 文件是一个使用称为 Web 服务描述语言(WSDL)的 XML 语法编写的 XML 文档。此文件定义 XML Web services 的行为方式，并指示客户端如何与该服务交互。

3.8　Bin文件夹——编译程序集文件夹

Bin 文件夹用来存放用户要在应用程序中引用的控件、组件或其他代码的已编译程序集（.dll 文件），在应用程序中将自动引用 Bin 文件夹中的代码所表示的任何类。

Bin 文件夹中的程序集无须注册。只要.dll 文件存在于 Bin 文件夹中，ASP.NET 就可以识别它；如果用户更改了.dll 文件，并将它的新版本写入到了 Bin 文件夹中，则 ASP.NET 会检测到并进行更新，并对随后的新页请求使用新版本的.dll 文件。

关于 Bin 文件夹的安全性简单介绍如下。

将编译后的程序集放入 Bin 文件夹中会带来安全风险。如果是用户自己编写和编译的代码，那么用户了解代码的功能。但是，用户必须像对待任何可执行代码一样来对待 Bin 文件夹中已编译的代码。在完成代码测试并确信已了解代码功能之前，要对已编译的代码保持谨慎的态度。

> **注意** Bin 文件夹中程序集的作用范围为当前应用程序。因此，它们无法访问当前 Web 应用程序之外的资源或调用当前 Web 应用程序之外的代码。

示例　可以在 Bin 文件夹中存储编译的程序集，并且 Web 应用程序任意位置的其他代码（如页代码）会自动引用该文件夹。典型的示例是将自定义类编译好的代码编译成程序集，然后复制到 Web 应用程序的 Bin 文件夹中，这样所有页都可以使用这个类。

下面以在应用程序的 Bin 文件夹中添加引用 Server.dll（自定义编译好的程序集）文件为例，来介绍如何创建 Bin 文件夹，并将自定义的 Server.dll 文件添加到创建的这个 Bin 文件夹中。

在菜单栏中选择“网站”→“添加引用”命令，弹出如图 3.19 所示的对话框，选择“浏览”选项卡，在相应的位置找到 Server.dll 文件并添加网站中。在开发过程中程序员可以自由地调用相关的方法并实现相应的功能。

图 3.19　添加引用类库文件

完成以上操作步骤后，单击图 3.19 中的“ 确定 ”按钮，将会在应用程序中自动创建一个 Bin 文件夹，如图 3.20 所示。

图 3.20 创建完成的 Bin 文件夹

创建 Bin 文件夹除了上述方法外，也可以通过手动来创建，右击应用程序的名称，在弹出的快捷菜单中选择“ 添加 ASP.NET 文件夹 ”→“ Bin ”命令，便可创建一个 Bin 文件夹，如图 3.21 所示。

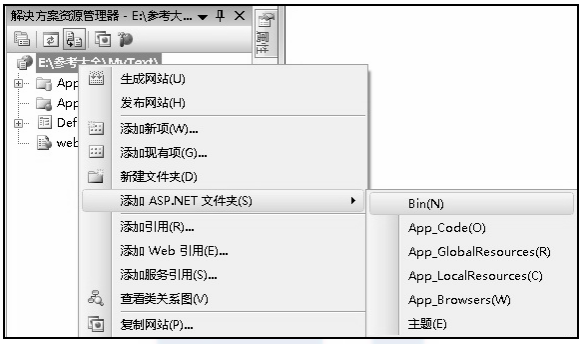

图 3.21 选择“添加 ASP.NET 文件夹”→“Bin”命令

手动添加完 Bin 文件夹后，如要将编译好的.dll 文件添加到 Bin 文件夹中，可右击创建的 Bin 文件夹，在弹出的快捷菜单中选择“ 添加引用 ”命令，如图 3.22 所示，将弹出与图 3.19 一样的对话框，可以以同样的操作步骤来完成.dll 文件的添加。

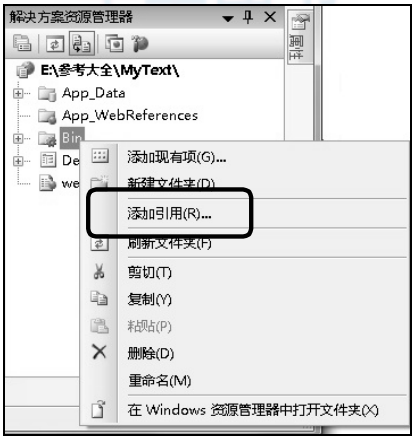

图 3.22 选择“添加引用”命令