高等院校计算机应用系列教材

ASP.NET Web开发技术

（第2版）（微课版）

彭玉华　邓谦　王先水　主编

清华大学出版社
北　京

内 容 简 介

ASP.NET Web 是目前软件开发市场比较流行的一种开发技术，可配合任何一种 .NET 平台的语言进行开发。本书以构建 SPOC 混合教学模式为基础，对 ASP.NET Web 开发技术课程进行总体设计。课程从“准职业人”的角度出发，以工作过程为导向、工作任务为驱动、学生能力提升为落脚点，重点培养学生的软件设计、代码编写和算法设计能力，通过课内与课外双线并行的方式实施教学。

本书共 10 章，内容包括 ASP.NET Web 技术概述、ASP.NET Web 标准服务器控件、用户控件和母版页技术、站点导航控件、ASP.NET 常用内置对象与数据传递、ASP.NET 状态管理、ADO.NET 数据库访问技术、数据绑定与数据绑定控件、ASP.NET AJAX 控件、基于三层架构开发课程学习选课平台。书中的所有案例均来自编者多年的教学手稿笔记及项目开发经验，具有很强的实用性。

本书可作为高等院校计算机相关专业的 Web 开发、网络程序设计、Web 数据库应用技术等课程的教材，也可供对 Web 应用开发感兴趣的技术人员自学使用。

图书在版编目 (CIP) 数据

ASP.NET Web 开发技术 : 微课版 / 彭玉华 , 邓谦 , 王先水主编 . -- 2 版 . -- 北京 : 清华大学出版社 , 2026. 4. -- (高等院校计算机应用系列教材). -- ISBN 978-7-302-71271-8

Ⅰ. TP393.092

中国国家版本馆 CIP 数据核字第 2026KA1267 号

责任编辑：刘金喜
封面设计：高娟妮
版式设计：思创景点
责任校对：马遥遥
责任印制：丛怀宇

出版发行：清华大学出版社
网　　址：https://www.tup.com.cn，https://www.wqxuetang.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-83470000　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn

印 装 者：三河市人民印务有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**22　　**字　　数：**563 千字
版　　次：2023 年 1 月第 1 版　　2026 年 6 月第 2 版　　**印　　次：**2026 年 6 月第 1 次印刷
定　　价：79.00 元

产品编号：115960-01

前言

党的二十大报告指出，教育、科技、人才是全面建设社会主义现代化国家的基础性、战略性支撑。必须坚持科技是第一生产力、人才是第一资源、创新是第一动力，深入实施科教兴国战略、人才强国战略、创新驱动发展战略。这三大战略共同服务于创新型国家的建设。高等教育与经济社会发展紧密相连，对促进就业创业、助力经济社会发展、增进人民福祉具有重大意义。

ASP技术为早期Web领域的发展注入了强劲动力，作为一套能够高效解决Web开发痛点的服务端技术栈，它支持Web服务器端逻辑与客户端应用程序的协同开发，大大降低了动态网站的开发门槛。本书在内容组织和编写过程中体现了以下特点。

本书以构建SPOC混合教学模式为基础，对ASP.NET Web开发技术课程进行总体设计。课程从“准职业人”的角度出发，以工作过程为导向、工作任务为驱动、学生能力提升为落脚点，重点培养学生的软件设计、代码编写和算法设计能力，通过课内与课外双线并行的方式实施教学，旨在培养在Web开发设计、数据库设计，以及ASP.NET和ADO.NET等核心技术领域兼具高技能与高素质的应用型人才。依据职业岗位能力要求，课程内容被划分为四大模块：Web网站设计基础模块、Web开发控件基础模块、ADO.NET数据库访问技术模块和ASP.NET项目上机实验模块。

本书的编写目的在于帮助广大学生和开发人员更快、更好地理解和掌握ASP.NET Web开发技术的核心知识点。本书在编写过程中参考了目前市面上已有的相关书籍，集各家之所长，结合编者多年的教学手稿笔记进行扩展与整理，将一些原本深奥难懂的开发技术思想融入简单的案例进行解析，让学生能够轻松掌握其精髓。

本书以“案例驱动教学”为整体编写原则，所有开发技术知识要点的讲解均围绕一至两个案例展开，通过案例帮助读者加深对项目开发技术思想的理解。书中案例来自企业真实项目的拆分与提炼，上机实验部分体现了知识的综合应用及设计开发能力的培养。书中的例题与上机实验内容均在Visual Studio 2022及以上版本的开发环境中通过测试且运行无误。在这种指导思想下，组织本书的内容如下。

第1章 ASP.NET Web技术概述：主要介绍软件体系架构、Web工作原理、Web网页开发技术，以及ASP.NET的运行、开发环境和配置。

第2章 ASP.NET Web标准服务器控件：主要介绍常用的ASP.NET Web标准服务器控件和ASP.NET 验证控件的属性、事件、方法及应用。

第3章 用户控件和母版页技术：基于项目开发中保持页面风格一致的需求，讲解了用户控件和母版页的创建和使用方法。

第4章 站点导航控件：主要介绍站点导航控件在网站开发中的基本应用。

第5章 ASP.NET常用内置对象与数据传递：主要介绍ASP.NET各内置对象的属性、方法及基本应用，讲解了跨页传递数据的基本方法和应用。

第6章 ASP.NET状态管理：主要介绍ViewState对象、Cookie对象、Session对象和Application对象的常用属性及基本应用。

第7章 ADO.NET数据库访问技术：主要介绍ADO.NET的数据模型及常用对象、数据库连接字符串、数据库连接对象Connection、数据库命令对象Command、读取数据对象DataReader、DataSet对象、DataAdapter对象和DataTable对象。

第8章 数据绑定与数据绑定控件：主要介绍通过数据绑定表达式实现数据绑定的基本原理，常用数据绑定控件(GridView、DetailsView、FormView等)的常用属性，以及数据绑定的基本原理和基本应用。

第9章 ASP.NET AJAX控件：本章致力于提升用户体验，重点介绍AJAX技术的工作原理、常用ASP.NET AJAX控件的基本语法和基本应用。

第10章 基于三层架构开发课程学习选课平台：通过案例简单介绍三层架构的基本原理和MVC开发技术的入门知识。

本书由武昌理工学院人工智能学院计算机科学与技术系彭玉华、软件工程系邓谦，武汉工程科技学院计算机与人工智能学院计算机系王先水共同编写完成。

本书案例主要源自编者多年教学积累的手稿笔记和讲稿，同时参考了相关文献中的部分内容，并汲取了同行的宝贵经验，在此谨表谢意。限于编者水平，书中难免存在疏漏与不足之处，欢迎广大读者批评指正。

为方便教师教学和学生自学，本书提供PPT课件、案例源代码、教学大纲和教学讲稿，可通过扫描下方二维码下载。微课视频可通过扫描书中二维码观看。


PPT 课件


案例源代码


教学大纲


教学讲稿

服务邮箱：476371891@qq.com。

编　者

2026年1月于武汉

目录

第 1 章

ASP.NET Web 技术概述

Web技术的飞速发展增强了人们对现实世界的认识，为人们的生活提供了极大的便利。ASP.NET是进行Web应用程序开发的主流技术之一，该技术易学易用、开发效率高，可以配合任何一种.NET语言进行Web开发。

1.1 软件体系架构

1.1 软件体系架构 −1.3 Web 网页开发技术

架构设计的初衷是为了进行超越算法和数据结构的设计，以适应软件规模和复杂性的增长。美国Borland公司最早研发的C/S(client/server，客户/服务器)架构和美国微软公司研发的B/S(browser/server，浏览器/服务器)架构是当今主流软件开发领域中的两大核心体系结构。

1.1.1 C/S架构

C/S架构是典型的两层架构，包含客户端和服务器端。客户端包含一个或多个在用户计算机上运行的程序。服务器端有两种形式：一种是数据库服务器端，客户端通过数据库连接访问服务器端的数据；另一种是Socket服务器端，服务器端的程序通过Socket与客户端的程序通信。

在C/S架构中，客户端需要实现绝大多数的业务逻辑和页面展示，通过与数据库的交互(通常是SQL或存储过程)实现数据持久化，以此满足实际项目的需要。

C/S架构的优点：界面和操作丰富；安全性能容易保证，容易实现多层认证；只有一层交互，响应速度非常快。

C/S架构的缺点：适用面窄，常用于局域网中；用户群固定，程序需要安装才可使用；维护成本高，软件每升级一次，其所涉及的所有客户端程序都需要升级。

1.1.2 B/S架构

B/S架构是三层架构，由浏览器端、Web服务器端、数据库端构成。其中，浏览器指的是

Web浏览器，其极少数业务逻辑在前端实现，主要的业务逻辑在服务器端实现。B/S架构系统只要有Web浏览器即可访问，无须安装软件。

B/S架构中的显示逻辑在Web浏览器上完成，而业务逻辑则在Web服务器上完成，减轻了客户端压力。

B/S架构的优点：客户端无须安装，只需Web浏览器即可；B/S架构可直接接入局域网，通过一定的权限控制实现多客户访问的目的，交互性强；B/S架构只需升级服务器软件。

B/S架构的缺点：在跨浏览器兼容方面，B/S架构不尽如人意；在速度和安全性方面需要投入较高的设计成本。

1.2 Web技术概述

1.2.1 Web工作原理

Web是采用超文本传输协议(hypertext transfer protocol，HTTP)、基于客户端/服务器(C/S)架构的一种体系结构，客户端在计算机上使用浏览器向Web服务器发出请求，服务器响应客户端请求，并向客户端返回请求的网页，客户端在浏览器窗口上显示网页的内容。

Web体系结构由以下3部分构成。

1) Web服务器

用户要访问Web页面或其他资源，必须事先有一个服务器来提供Web页面和这些资源，这种服务器称为Web服务器。

Web服务器是向浏览器提供服务的程序，主要功能是提供网上信息浏览服务。Web服务器在应用层使用超文本传输协议(HTTP)，其信息内容采用HTML(hypertext markup language，超文本标记语言)格式，信息定位则使用URL(uniform resource locator，统一资源定位符)。

比较常用的Web服务器有Apache和IIS(Internet Information Services，互联网信息服务)。Apache已经发展成为Internet上最流行的服务器。IIS是Microsoft公司开发的、专用于Windows平台的Web服务器，该服务器占市场份额三分之一左右，也是本书所使用的服务器。

2) Web浏览器

Web浏览器是Web服务的客户端程序，它能够向Web服务器发送各种请求，并对服务器发来的网页和各种多媒体资源进行解释、显示和播放。浏览器的主要功能是解析网页文件内容并正确显示，网页一般采用HTML格式，浏览器是最常使用的客户端程序。

3) 通信协议

客户端与服务器之间采用超文本传输协议(HTTP)进行通信。该协议详细规定了Web客户端与服务器之间如何通信。

客户端若要访问Web服务器上的某个资源，就必须知道该资源的位置，这一位置是通过URL来表示的。URL用于指向Internet上位于某个特定位置的某个资源，涵盖HTML文件、图像文件、JSP页面文件、ASP页面文件及Vue.js组件等。

例如，下面是合法的URL。

```
http://localhost:6901/CountUser.aspx
http://localhost:5763/UserCount.aspx
https://tch.ityxb.com/textbook/detail/ff8080816a537512016a575beb390003
https://www.baidu.com/index.php?tn=monline_3_dg
```

URL由4部分组成：协议名称、域名、可选的端口号，以及资源路径和名称。其中，端口号和资源名称可以省略。

人们使用最多的协议是HTTP，其他常用的协议有FTP、TELNET、MAIL和FILE。

域名由域名段组成，如www、edu、cn等。

端口号用于标明服务由服务器的哪个端口提供，一些常见的服务都有固定的端口号，如HTTP服务的默认端口号为80，如果访问的是默认端口上的服务，则端口号可以省略。

URL最后一部分为资源在服务器上的相对路径和名称，如/index.html，它表示服务器上根目录下的index.html文件。

1.2.2　HTTP和HTML

1. HTTP

超文本传输协议(HTTP)是因特网上应用较广泛的一种协议。HTTP是一种基于请求—响应(request-response)的无状态协议。这种请求—响应的过程如图1.1所示。

图 1.1　HTTP 请求—响应过程

在图1.1中，客户端首先通过浏览器程序与Web服务器建立连接，并向服务器发送HTTP请求消息；Web服务器接收到客户端的请求后，对请求进行处理，然后向客户端发送HTTP响应消息。客户端接收服务器发送的响应消息，对消息进行处理并关闭连接。

例如，在浏览器地址栏中输入https://www.wuhues.com/，按Enter键后，浏览器就会创建一个HTTP请求消息，使用DNS获得www.wuhues.com主机的IP地址，创建一条TCP连接，通过这条TCP连接将HTTP请求消息发送到服务器，并从服务器接收回一条响应消息，该消息包含将显示在浏览器客户端中的内容。

HTTP的特点主要有：以B/S架构为基础；简单快速，浏览器向服务器请求服务时只需传送请求方法和路径，协议的简洁性使得服务器和程序规模小，通信速度快；HTTP允许传输任意类型的数据对象；浏览器向服务器发出一个请求，服务器响应处理客户端的请求并收到客户端的应答后，断开服务器与浏览器的连接，从而节约传输时间；HTTP对于事务处理没有记忆能力。

2. HTML

超文本标记语言(HTML)是一种用来制作超文本文档的简单标记语言。超文本是指用HTML编写的文档中可以包含指向其他文档或资源的链接，该链接称为超链接(hyperlink)。通过超链接，用户可以很容易地访问所链接的资源。

HTML文档是由一些标签组成的文本文件，标签标识了内容和类型，Web浏览器通过解析

这些标签进行显示。HTML文档可以用任意文本编辑器创建，但创建的文件扩展名必须是.htm或.html。目前较流行的编辑器是Visual Studio Code编辑器。

1.2.3 网页开发技术

网页分为静态网页和动态网页两大类，相应的网页开发技术也分为静态网页开发技术和动态网页开发技术。

1. 静态网页开发技术

静态网页是指用纯HTML代码编写的网页，并以扩展名为.html或.htm的文件形式保存。这种网页在设计完成后，任何人在任何时候采用任何方式浏览该页面，其效果都是相同的。因此，这种网页的内容更新较为烦琐，必须在设计制作完成后，使用专门的软件上传到服务器上才能更新。

静态网页开发技术适用于更新较少的展示型网站，可用于传统的媒体广告。页面上会出现各种动态效果，如GIF格式的动画、滚动字幕等，但这些动态效果只是视觉上的。

静态网页的执行过程：用户通过客户端浏览器输入网址并按Enter键后，发出WWW请求；服务器收到静态网页请求后，从硬盘的指定位置查找相应的HTML文件，将找到的HTML文件返回给服务器；再由服务器向客户端返回请求的文件；客户端浏览器收到请求的文件后，解析这些HTML代码并将其显示出来。

静态网页中通常没有程序代码，只有HTML标记，但静态网页中可以包含客户端脚本，常见的客户端脚本语言有JavaScript。客户端脚本能够在特定的网页中改变界面及行为，或者响应鼠标或键盘操作。也就是说，静态网页的动态行为都是在客户端进行的，而网页本身是静态的。

2. 动态网页开发技术

动态网页是指在执行时，用户可以输入各类允许的信息，以实现人机交互，网页能根据不同时间、不同访问者显示不同的内容。采用动态网页开发技术，可实现用户注册、用户登录、用户管理、订单管理等操作。动态网页的文件格式根据不同的程序设计语言而定，常见的有ASP、JSP、PHP等。

动态网页的执行过程：用户通过客户端浏览器输入网址并按Enter键后，发出WWW请求；服务器收到动态网页请求后，从硬盘的指定位置查找相应的动态网页文件，并将该动态网页文件返回给服务器；服务器执行其中的程序代码，生成HTML文件；服务器向客户端返回HTML文件，客户端浏览器收到请求的文件后，对文件中的HTML标记进行解析，将其渲染为可视化网页内。

对于动态网页而言，服务器的主要功能是通过文件系统找到用户要访问的动态网页文件，执行该网页文件中的程序代码，生成HTML文件，再将HTML文件传给客户端浏览器。

1.3 Web网页开发技术

WWW是一种典型的分布式应用架构，其应用中的每次信息交换都涉及客户端和服务器端两个层面。因此，Web网页开发技术主要分为Web客户端技术和Web服务器端技术。

1.3.1　Web客户端技术

Web客户端的首要任务是展现信息内容，而HTML是信息展现的最有效载体。最初的HTML只能在浏览器中展现静态的文本或图像信息，满足不了人们对信息丰富性和多样性的强烈要求，因此由静态网页技术向动态网页技术转变是Web客户端技术演进的必然趋势。目前，支持Web客户端动态技术的语言主要有VBScript、JavaScript脚本语言。

1. HTML

HTML是在Internet上用于编写网页的主要语言，它提供了简洁且功能强大的文件定义方式，可以设计出多姿多彩的超媒体文件。借助HTTP，HTML文件可以在全球互联网上进行跨平台的文件交换。

HTML文件是纯文本格式，文件中的文字、字体、字号、段落、图片、表格及超链接，甚至文件名称等，都是用不同意义的标签来描述的，以此定义文件的结构与文件间的逻辑关联。简而言之，HTML使用标签来描述文件中的多媒体信息。

客户端浏览器按顺序解释执行HTML文件，对于HTML文件中的标签错误或属性错误，既不报错，也不中止执行。不同的浏览器对同一标记会有不同的解释，也就有不同的显示效果。

2. CSS

CSS(cascading style sheets，层叠样式表)属于动态HTML技术，扩充了HTML标记的属性设置，使得页面显示效果更加丰富，表现方式更加灵活，它与DIV配合使用可以很好地对页面进行分割和布局。CSS能对页面元素和布局进行更精确的控制，同时能够实现页面内容与表现形式的分离，使得网站的设计风格趋向统一，维护更加容易。

3. JavaScript脚本语言

JavaScript是一种轻量级的直译式(解释型)编程语言，基于ECMAScript标准(该标准由 ECMA国际组织通过ECMA-262规范制定，是脚本程序语言的通用标准)，于1995年由网景(Netscape)公司开发，具备基于对象、事件驱动的核心特性。JavaScript、HTML和CSS被称为“Web前端开发的三大技术”。目前，JavaScript已经被广泛应用于Web开发，几乎所有的浏览器都支持JavaScript，无须额外安装第三方插件。

JavaScript脚本语言的特点：JavaScript是一种直译式(解释型)的脚本语言，无须事先编译，在程序运行的过程中以逐行解释的方式执行；JavaScript具有非常简单的语法，无须定义变量类型，所有变量的声明都可以用统一的类型关键字表示；JavaScript是一种Web程序开发语言，只与浏览器的支持有关，与操作系统的平台类型无关；JavaScript语言对大小写非常敏感。

1.3.2　HTML常用标签

1. 段落、行内和换行标签

为了使网页中的文字有条理地进行显示，HTML提供了段落标签<p>和行内标签。如果希望某段文本换行显示，则需要使用换行标签
。

段落、行内和换行标签的示例代码如下。

```
<body>
    <p>
        我是一名软件开发爱好者
        <span>通过br标签</span>可能实现<br />换行效果
    </p>
</body>
```

使用浏览器解析上述代码，效果如图1.2所示。

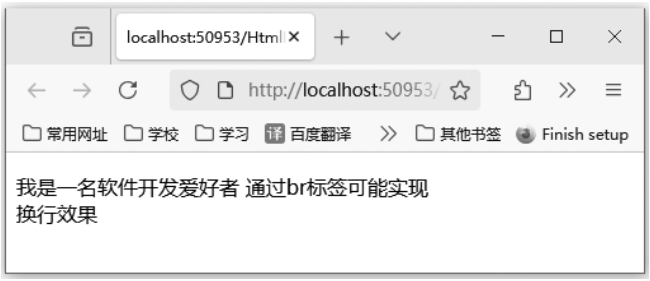

图 1.2　段落、行内和换行标签代码解析效果

从图1.2中可以看出，使用标签对文本没有影响，但使用换行标签
，则会对文本进行强制换行。

2. 文本标签

在HTML中，使用<font>标签可以控制网页中文本的样式，如字体、字号和颜色。<font>标签的基本语法如下。

```
<font 属性="属性值" > 文本内容</font>
```

<font>标签示例代码如下。

```
<body>
    <font face="隶书" size="3" color="green">
        文本字体是隶书，字体大小是3号，颜色是绿色
    </font>
</body>
```

使用浏览器解析上述代码，效果如图1.3所示。

图 1.3　<font> 文本标签代码解析效果

在<font>文本标签中使用face、size、color属性分别设置了文本的字体、大小和颜色。

3. 表格标签

在设计网页时，为了使网页中的数据能够有条理地显示，可以使用表格对网页进行设计。在HTML文档中，表格标签的基本语法格式如下。

```
<table>
    <tr>
```

```
            <td>单元格内容</td>
        </tr>
    </table>
```

说明：

语法格式中有3对HTML标签，分别是<table></table>、<tr></tr>、<td></td>，它们是创建表格的基本标签，缺一不可。

(1) <table></table>标签：用于定义一个表格。

(2) <tr></tr>标签：用于定义表格中的行，必须嵌套在<table></table>标签中。

(3) <td></td>标签：用于定义表格中的单元格，也称为表格中的列，必须嵌套在<tr></tr>标签中。

表格标签示例代码如下。

```
<table border="1px">
        <tr>
            <td>姓名</td>
            <td>编程</td>
            <td>数据库</td>
            <td>英语</td>
        </tr>
        <tr>
            <td>李四</td>
            <td>90</td>
            <td>85</td>
            <td>83</td>
        </tr>
    </table>
</body>
```

在示例代码中，<table>标签中的border="1px"属性用于为每个单元格设置边框，并用边框围绕表格。这里的px指的是像素单位，1px表示该表格边框的宽度是1像素。使用浏览器解析上述代码，效果如图1.4所示。

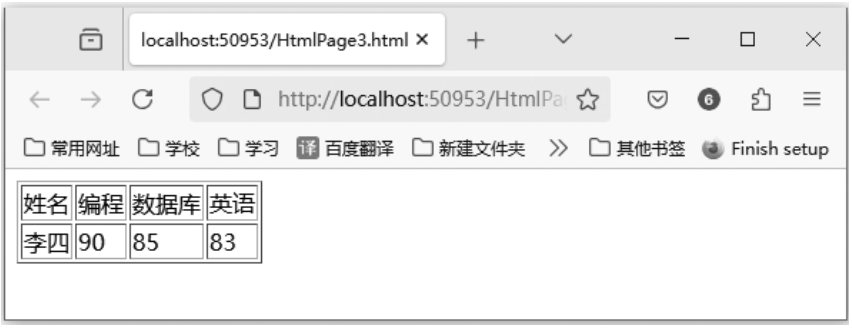

图 1.4　<table> 表格标签代码解析效果

4. 表单标签

表单是指在网页上用于输入信息的区域，主要功能是收集数据信息，并将这些信息传递给后台信息处理模块。表单主要由表单控件、提示信息和表单域组成。

1) 表单控件

表单控件是指具体的表单功能项，如单行文本输入框、密码输入框、复选框、提交按钮等。

在HTML中，表单控件<input/>的基本语法如下。

```
<input  type="控件类型"/>
```

其中，控件类型包括单行文本输入框(text)、密码输入框(password)、复选框(checkbox)、单选按钮(radio)、提交按钮(submit)、重置按钮(reset)、普通按钮(button)。

表单控件<input/>还可以定义很多其他属性，其中常用的有id、name、value、size，它们分别用于指定<input/>控件的ID值、名称、控件中的默认值和控件在页面中的显示宽度。

2) 提示信息

表单中通常需要包含一些说明文字，即表单控件前的说明文字，用于提示用户进行填写和操作。

3) 表单域

表单域相当于一个容器，用于容纳所有的表单控件和提示信息，用<form></form>标签表示。

在HTML中，<form>标签的基本语法如下。

```
<form action="地址" method="提交方式" name="表单名称">
     各种表单控件
 </form>
```

说明：

语法中的action="地址" method="提交方式" name="表单名称"为<form>标签的常用属性，采用键值对形式表示。

action属性用于指定表单提交的地址。method属性用于设置表单数据的提交方式，其值可以是GET或POST，其中GET为默认值。GET方式提交的数据将显示在浏览器的地址栏中，保密性差且有数据限制；POST提交方式保密性强，允许提交的数据量大，因此，POST方式比较常用。

表单控件的示例代码如下。

```
<body>
   <form action="#" method="post" >
      <div>
         用户注册
         <div>用户名：</div>
         <div>
            <input type="text" name="username"/>
         </div>
         <div>密码：</div>
         <dic>
            <input  type="password" name="userpassword"/>
         </dic>
         <div>性别：</div>
         <div>
            <input  type="radio" name="gender" value="male"/>男
            <input  type="radio" name="gender" value="female"/>女
         </div>
         <div>
            <input  type="submit" value="注册"/>
            <input  type="reset" value="重置"/>
         </div>
      </div>
   </form>
</body>
```

使用浏览器解析上述代码，效果如图1.5所示。

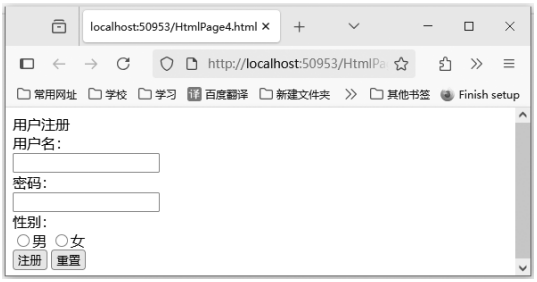

图 1.5　表单标签代码解析效果

5. 列表标签

列表标签是网页中常用的标签。列表可分为无序列表、有序列表和自定义列表。

1) 无序列表

无序列表是指没有特定顺序的列表，各列表之间没有先后顺序，通常是并列的。无序列表的基本语法如下。

```
<ul>
    <li>列表项1</li>
    <li>列表项2</li>
    <li>列表项3</li>
    <li>列表项4</li>
</ul>
```

其中，
标签用于定义无序列表，
标签嵌套在
标签中，用于描述具体的列表项，每对 标签中至少包含一对
标签。

无序列表示例代码如下。

```
<body>
    计算机课程列表
    <ul>
        <li>C语言</li>
        <li>Java语言</li>
        <li>C#语言</li>
        <li>Python语言</li>
    </ul>
</body>
```

使用浏览器解析上述代码，效果如图1.6所示。

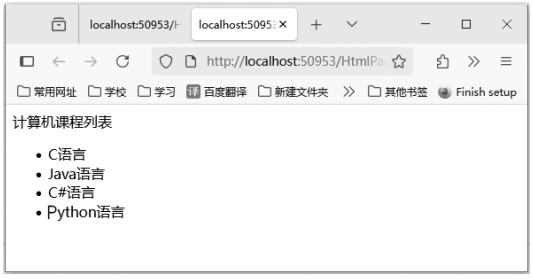

图 1.6　无序列表标签代码解析效果

2) 有序列表

有序列表是一种强调排列顺序的列表，使用<ol>标签定义，内部可以嵌套多个<li>标签。网页

中常见的游戏排行榜、歌曲排行榜等都可以通过有序列表来定义。有序列表的基本语法如下。

```
<ol>
    <li>列表项1</li>
    <li>列表项2</li>
    <li>列表项3</li>
    <li>列表项4</li>
</ol>
```

其中，
标签用于定义有序列表，
标签嵌套在
标签中，用于描述具体的列表项，每对
标签中至少包含一对
标签。

3) 自定义列表

自定义列表与有序列表、无序列表有所区别，它由<dl>、<dt>和<dd>来定义。自定义列表的语法基本格式如下。

```
<dl>
     <dt>标题1</dt>
     <dd>dd是标题1的描述内容1</dd>
     <dd>dd是标题1的描述内容2</dd>
     <dt>标题2</dt>
     <dd>dd是标题2的描述内容1</dd>
     <dd>dd是标题2的描述内容2</dd>
 </dl>
```

说明：

<dl>标签用于定义自定义列表，<dt>标签和<dd>标签并列嵌套于<dl>标签中。<dt>标签用于指定术语标题，<dd>标签用于对标题进行解释和描述。一对<dt></dt>标签可以对应多对<dd></dd>标签，即一个标题可以有多项解释。

自定义列表展示联系人信息，示例代码如下。

```
<body>
    <h3>定义列表展示联系人信息</h3>
    <dl>
        <dt>联系人</dt>
        <dd>李维民</dd>
        <dd>电话：1398716××××</dd>
        <dd>Email：123@qq.com</dd>
        <dt>联系地址</dt>
        <dd>湖北省武汉市 江夏区</dd>
        <dt>邮政编码</dt>
        <dd>430200</dd>
    </dl>
</body>
```

使用浏览器解析上述代码，效果如图1.7所示。

图 1.7　自定义列表标签代码解析效果

6. 超链接标签

超链接标签在网页中比较常用。一个网站通常由多个页面构成，打开网站时首先看到的是首页，如果要从首页跳转到其他子页，则需要在首页的相应位置添加超链接。在HTML中创建超链接的标签是<a>，其基本语法如下。

```
<a href="目标地址" target="目标窗口弹出方式" title="显示信息">文本或图像</a>
```

说明：

<a>标签是一个行内标签，用于定义超链接，href和target是<a>标签的常用属性。

href属性用于指定链接指向的页面URL，当在<a>标签中使用href属性时，该标签才具有超链接功能。

target属性用于指定页面的打开方式，其取值包括：_self，即在当前框架中打开链接；_blank，即在一个全新的空白窗口中打开链接；_top，即在顶层框架中打开链接；_parent，即在当前框架的上一层中打开链接。

1.3.3　CSS技术

随着Web技术的发展，单一的HTML属性样式已不能满足Web设计的多样化需求，开发者需要在页面上展示更加丰富的内容、视觉效果和动画等，因此在HTML中引入了CSS技术以满足这些需求。

1. CSS样式的引用方式

在网页中引入CSS样式主要有行内式、内嵌式和外链式3种方式，下面分别介绍这3种方式在网页中的应用。

1) 行内式

行内式又称为内联式，通过标签的style属性设置该标签的样式。行内式的基本语法格式如下。

```
<标签名 style="属性1:属性值1; 属性2:属性值2; ...">内容</标签名>
```

说明：

style是HTML标签的属性，用于设置该标签的行内样式。属性与属性值之间用英文冒号连接，多个属性之间用英文分号隔开。

行内式CSS只对所在行的标签及嵌套在其中的子标签起作用，并且无法实现结构与样式的分离。

行内式示例代码如下。

```
<body>
    <div style="width:300px;margin:auto;font-family:隶书;font-size:23px">
    使用行内式修饰DIV层中内容
    </div>
</body>
```

使用浏览器解析上述代码，效果如图1.8所示。

图 1.8　行内式 CSS 浏览器解析效果

此段行内式代码中定义了DIV内容在浏览器中的显示宽度、对齐方式，以及字体和字号。

2) 内嵌式

内嵌式是将CSS代码集中写在HTML文档的头部标签<head>中，并用<style>标签定义，其语法格式如下。

```
<style type="text/css">
    选择器 {
        属性1: 属性值1;
        属性2: 属性值2;
             ...
    }
</style>
```

说明：

<style>标签通常位于<head>标签内，且置于<title>标签之后，由于浏览器是从上到下解析代码的，为便于CSS代码提前被加载和解析，则将其放在头部，避免网页内容加载后没有样式修饰的问题。

在<style>标签中，只有设置type的属性值为“text/css”，浏览器才能识别<style>标签中含有CSS代码。

内嵌式CSS只对其所在的HTML页面有效，但在网站开发时，往往会有若干页面，因此它不能充分发挥CSS代码的重用优势。

内嵌式示例代码如下。

```
<html>
<head>
    <meta charset="utf-8" />
    <title></title>
    <style type="text/css">
        div{width:300px;margin:auto;font-family:隶书;font-size:23px}
    </style>
</head>
<body>
    <div>
        使用内嵌式修饰DIV层中内容
    </div>
</body>
</html>
```

使用浏览器解析上述代码，效果如图1.9所示。

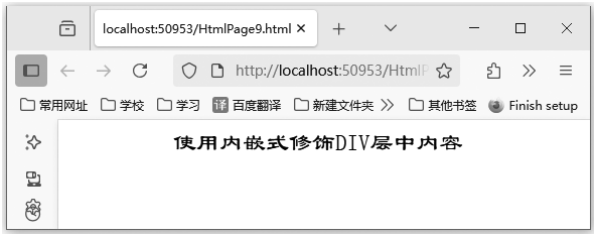

图 1.9　内嵌式 CSS 浏览器解析效果

上述代码中，在<head>标签中的<title>标签之后定义了<style>样式，在该样式中定义了width属性、margin属性、font-family属性和font-size属性，实现了DIV内容在浏览器中的显示效果。

3) 外链式

外链式是指所有的样式放在一个或多个以.css为扩展名的外部样式表文件中，通过<link>标签将外部样式表文件链接到HTML文件中，其基本语法格式如下。

```
<link  href="CSS文件的路径" type="text/css" rel="stylesheet"/>
```

说明：

<link>标签需要置于<head>标签内，并且通常需要指明<link>标签的3个属性，即href、type和rel属性。

href属性用于定义所链接外部样式表文件的地址，可以是相对路径，也可以是绝对路径。

type属性用于定义所链接文档的类型，这里需要指明类型为“text/css”，表明所链接的外部文件是CSS。

rel属性用于定义当前文档与被链接文档之间的关系，这里需要指定为“stylesheet”，表示被链接的文档是一个样式表文件。

外链式在实际项目开发中是最实用的引入方式，它将HTML代码与CSS代码分离为两个或多个文件，真正实现结构与表现的分离，且同一个CSS文件可以被多个HTML页面链接使用，极大地提高了网页开发效率。

外链式示例代码如下。

首先创建一个名为style的CSS文件。

```
div {
    width:300px;
    margin:auto;
    font-family:隶书;
    font-size:23px;
}
```

其次创建一个名为htmlPage1的HTML文件。

```
<html>
<head>
    <meta charset="utf-8" />
    <title>使用外链式</title>
      <link  href="Style.css" type="text/css" rel="stylesheet"/>
</head>
<body>
    <div>
        使用外链式修饰DIV层中内容
```

```
    </div>
</body>
</html>
```

使用浏览器解析上述代码，效果如图1.10所示。

图 1.10　外链式 CSS 浏览器解析效果

2. CSS选择器和常用属性

将CSS样式应用于特定的HTML标签元素，需要通过CSS规则来实现。CSS规则由选择器和声明组成，声明则由属性和属性值对组成。CSS提供了丰富的选择器类型，如标签选择器、类选择器、id选择器。

1) 标签选择器

标签选择器是指直接使用HTML标签名称作为选择器，它定义的样式作用于页面中所有与选择器同名的标签。标签选择器的基本语法格式如下。

```
标签名{属性1:属性值1;属性2:属性值2;...}
```

说明：

所有HTML标签都可作为标签选择器，用标签选择器定义的样式对页面中该类型的所有标签都有效。

2) 类选择器

任何合法的HTML标签都可以使用class属性，class属性用于定义页面HTML元素标签组，这些标签组具有相同的功能或作用。

类选择器由点号“.”和类名构成，类选择器的基本语法格式如下。

```
.类名 { 属性1:属性值1;属性2:属性值2;...}
```

3) id选择器

任何合法的HTML标签都可以使用id属性，且id属性的取值必须是唯一的，只能用于指定一个标签。id选择器由“#”和id名组成，且id名通常以字母开头，可包含字母、数字、下画线和连接符。id选择器的基本语法格式如下。

```
#id名{ 属性1: 属性值1;属性2:属性值2;...}
```

对于CSS来说，id选择器与类选择器的功能很相似。一般而言，类选择器更加灵活，既能实现id选择器的所有功能，也能应用于更复杂的场景；对于需要被唯一标识的页面元素，则可使用id选择器。

1.3.4　Web服务器端技术

Web服务器端技术是指服务器端的脚本编程技术，常用的服务器脚本语言有CGI、ASP、PHP、JSP、ASP.NET、Python等。这些脚本语言的共同特点是都运行于服务器端，能够动态生成网页，其运行不受客户端浏览器的限制。脚本程序通常将脚本语言嵌入HTML文档中，执行后返回给客户端的是HTML代码。

1. CGI

CGI即通用网关接口，是一种早期的动态网页技术，可以使用不同的程序设计语言(如VB、C/C++等)编写适合的CGI程序。该技术已经发展成熟且功能强大，但因存在编程难度大、运行效率低、修改复杂等问题，已逐渐被新技术取代。

2. ASP

ASP是Microsoft开发的服务器脚本环境，内置于IIS 3.0及更高版本中。通过ASP可结合HTML网页、ASP指令和ActiveX组件建立动态、交互且高效的Web服务器应用程序。使用ASP动态网页技术开发的程序代码是在服务器端执行的，服务器端将程序执行的结果返回给客户端浏览器，从而减轻客户端浏览器的负担，极大提高了服务器端与客户端浏览器的交互速度。

3. PHP

PHP是一种易于学习和使用的服务器端脚本语言，用户只需要具备基础的编程知识就能使用PHP构建一个具有交互功能的Web网站。PHP不需要特殊的开发环境，不仅支持多种数据库，还支持多种通信协议。

4. JSP

JSP是由Sun公司推出的，基于Java Servlet及Java体系的Web开发技术。在HTML文档中嵌入Java程序片段(Scriptlet)和JSP标记可形成JSP文件。JSP脚本运行于服务器端，可以跨UNIX、Linux、Windows平台使用。JSP代码需编译成Servlet并由Java虚拟机执行，编译操作仅在对JSP页面第一次请求时进行。

5. ASP.NET

ASP.NET是继ASP之后推出的全新动态网页开发技术，其建立在.NET Framework的公共语言运行时上，可用于在服务器上生成功能强大的Web应用程序。

6. Python

Python由荷兰人Guido van Rossum发明，于1991年公开发行，它是一种脚本语言。Python是面向对象的解释型程序设计语言，具有简洁性、易读性和可扩展性，被广泛应用于系统管理任务的处理及Web开发领域。

1.4　ASP.NET基础知识

1.4 ASP.NET 基础知识

ASP.NET是微软公司推出的企业级Web应用开发框架，作为.NET技术体系的核心组成部分，它运行于IIS服务器环境。该框架支持以多种.NET兼容语言(如C#、VB.NET等)进行开发，为构建现代化Web应用提供了完整的解决方案。

1.4.1 ASP.NET技术

ASP.NET是一种开发动态网站的技术，是.NET框架的重要组成部分，可以使用任何.NET兼容的语言来编写ASP.NET网站。

使用ASP.NET开发网站非常简单，因为其设计目标是大幅精简应用程序代码，改变过去需要编写很多重复代码的状况，力求以最少的代码实现所需功能，因此它被视为Web开发技术的一个重要里程碑。

同其他Web开发技术相比，使用ASP.NET开发网站的速度是非常惊人的，维护起来也相当方便，且所需代码少。同时，用户还可以根据自己的需求向ASP.NET添加自定义功能。ASP.NET具有以下基本特点。

(1) 开发效率高：使用ASP.NET服务器控件和新增功能的现有控件，可以轻松、快捷地创建ASP.NET网站。

(2) 灵活和可扩展：ASP.NET的功能都可以扩展，从而可以轻松地将自定义功能集成到程序中。

(3) 性能优化：通过使用缓存和SQL缓存失效等功能，可以优化网站性能。

(4) 安全性：向网站程序中添加身份和授权比以往任何时候更加简单。

(5) 利用ASP.NET中自带的jQuery组件，可以创建更高效、更具交互性、高度个性化的Web体验。

1.4.2 .NET Framework框架

.NET Framework框架又称为.NET框架，它是微软公司推出的完全面向对象的软件开发与运行平台，它有两个主要组件，分别是公共语言运行时(common language runtime, CLR)和类库(framework class library, FCL)。.NET Framework框架体系结构如图1.11所示。

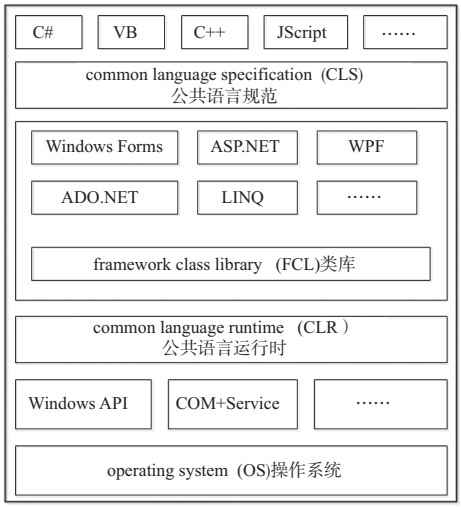

图 1.11 .NET Framework 框架体系结构

(1) 公共语言运行时。公共语言运行时负责管理和执行由.NET编译器编译产生的中间语言代码。公共语言运行时中包含两部分内容，分别是CLS和CTS。其中，CLS(公共语言规范)定义了许多应用程序所需的一套基本语言功能；CTS(通用类型系统)定义了在中间语言中使用的预定义

数据类型，所有面向.NET Framework的语言都可以生成基于这些类型的编译代码。

(2) 类库。类库实质上是一个存放编写好的类的集合，可供开发人员直接使用。例如，在进行多线程操作时，可以直接使用类库中的Thread类；在进行文件操作时，可以直接使用类库中的IO类；等等。

1.4.3　IIS的安装与配置

1. IIS的安装

ASP.NET作为一种服务，首先需要在运行它的服务器上建立互联网信息服务器(IIS)。IIS是微软公司主推的Web服务器。通过IIS，开发人员可以更方便地调试程序或发布网站。

下面介绍在Windows 10操作系统中安装IIS的步骤。

(1) 完成Windows 10操作系统安装后，执行“控制面板”|“程序”|“程序和功能”|“启用或关闭Windows功能”命令，系统弹出“Windows功能”对话框，如图1.12所示。

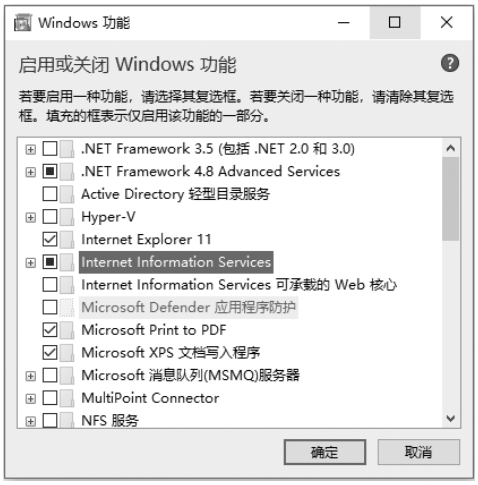

图 1.12　“Windows 功能”对话框

(2) 勾选“Internet Information Services”复选框，单击“确定”按钮，将进行IIS的安装，安装完成后系统自动关闭“Windows功能”对话框。

(3) IIS安装完成后，执行“控制面板”|“系统与安全”|“管理工具”命令，便可看到“Internet Information Services(IIS)管理器”。

2. IIS配置

IIS安装完成后就要对其进行必要的配置，这样才能使服务器在最优的环境下运行，下面介绍IIS配置与管理的基本步骤。

(1) 执行“控制面板”|“系统与安全”|“管理工具”命令，双击“Internet Information Services(IIS)管理器”，系统弹出“Internet Information Services(IIS)管理器”对话框，如图1.13所示。

图 1.13 “Internet Information Services(IIS) 管理器”对话框

(2) 在图1.13所示的左侧列表中执行“网站”|“Default Web Site”命令，在右侧列表中单击“绑定”超链接，系统弹出如图1.14所示的“网站绑定”对话框，在该对话框中可以添加、编辑、删除和浏览绑定的网站。

图 1.14 “网站绑定”对话框

(3) 在图1.14中，单击“添加”按钮，系统弹出“添加网站绑定”对话框(见图1.15)，在该对话框中可以设定绑定网站的类型、IP地址、端口及主机名等信息。

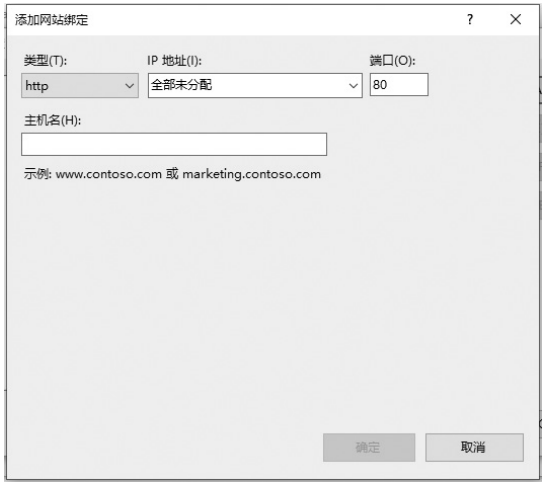

图 1.15 “添加网站绑定”对话框

(4) 设置完要绑定的网站后，单击“确定”按钮，返回“Internet Information Services(IIS)管理器”对话框，单击该对话框右侧列表中的“基本设置”超链接，系统弹出“编辑网站”对话框，在该对话框中可以设置应用程序池、网站的物理路径等信息。

1.4.4　ASP.NET引擎

在处理动态网页时，服务器既要查找动态网页文件，又要执行动态网页文件以生成HTML文件，为了减轻服务器的压力，可将Web服务器和动态网页源代码的执行分离开来。当一个Web请求到达时，Web服务器判断所请求的页面是静态网页还是动态网页。如果是静态网页，则服务器直接将该页面内容发送到请求浏览器；如果是动态网页，如ASP.NET网页，则服务器把执行该网页的任务转交给ASP.NET引擎，由ASP.NET引擎执行动态网页中的程序代码，以HTML文件形式返回给Web服务器，再由Web服务器将该HTML文件发送到请求的客户机浏览器。

ASP.NET动态网页的请求过程如图1.16所示。图1.16中涉及的典型组件包括：客户机浏览器，安装了IE浏览器；Web服务器，配置了IIS；数据库服务器，安装了SQL Server数据库管理系统。

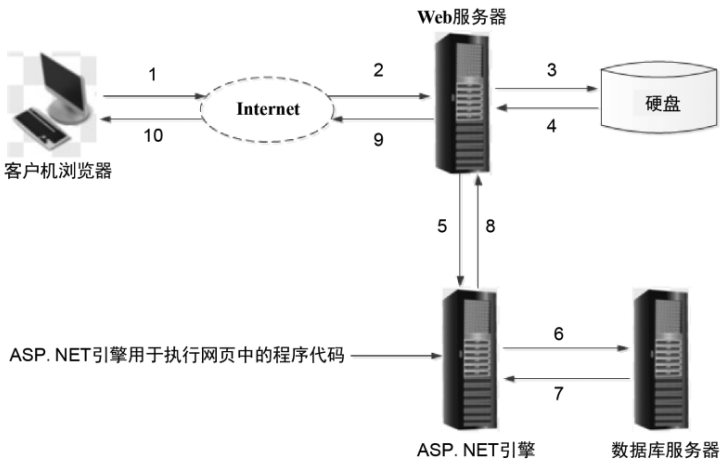

图 1.16　ASP.NET 动态网页的请求过程

图1.16中ASP.NET动态网页的请求过程如下。

(1) 客户端通过浏览器发出Web请求。

(2) Web服务器收到ASP.NET动态网页请求。

(3) Web服务器从硬盘的指定位置查找相应的ASP.NET动态网页文件。

(4) 将在硬盘中找到的ASP.NET动态网页文件返回给Web服务器。

(5) Web服务器将ASP.NET动态网页文件发送给ASP.NET引擎。

(6) ASP.NET引擎逐行读取该文件，并执行程序中的代码。如果需要访问数据库，则将这部分代码发送给数据库服务器；如果不需要访问数据库，则直接跳转到步骤(8)。

(7) 数据库服务器执行数据库访问，并将结果返回给ASP.NET引擎。

(8) ASP.NET引擎生成最终的纯HTML文件并返回给Web服务器。

(9) Web服务器将纯HTML文件发送给客户端浏览器。

(10) 客户端收到服务器返回的纯HTML文件，浏览器解析其中的HTML标记，并将其渲染为图形化的网页内容，显示在计算机屏幕上。

网页处理发生在Web服务器上，因此网页执行的每个操作都需要一次到服务器的往返过程。ASP.NET网页可以执行客户端脚本，而客户端脚本不需要与服务器进行往返通信，这对于客户输入数据验证和某些类型的用户界面编程十分有用。

1.4.5 ASP.NET应用程序开发工具

ASP.NET应用程序开发模式有ASP.NET Web窗体、ASP.NET MVC、ASP.NET Core等，实际开发时根据具体需求和开发人员的技术背景而定。本书采用ASP.NET Web窗体开发模式。

开发ASP.NET应用程序的主要工具是Visual Studio，本书采用Visual Studio 2022。

在开发ASP.NET Web窗体程序时，首要任务是开发ASP.NET动态网页，而执行网页程序代码的核心组件是ASP.NET引擎，在安装Visual Studio时，计算机自动配置好ASP.NET引擎。

Visual Studio是Microsoft推出的用于软件开发的重要平台，目前最高版本为Visual Studio 2022，内置.NET Framework 4.9及以上版本。Visual Studio开发平台将程序设计中需要的各个环节(如界面设计、代码编写、程序运行和调试)集成在同一个窗口中，极大地方便了开发人员的设计工作。通常，这种集多项功能于一体的开发平台称为集成开发环境(integrated development environment，IDE)。

1. Visual Studio 2022下载

用户可以到Microsoft的官方网站下载Visual Studio 2022版本的安装包，仅支持在线安装。

2. Visual Studio 2022安装和项目创建

Visual Studio 2022的安装步骤如下。

(1) 双击Visual Studio 2022安装包，在线下载安装所需要的安装包文件，下载安装包文件的过程如图1.17所示。

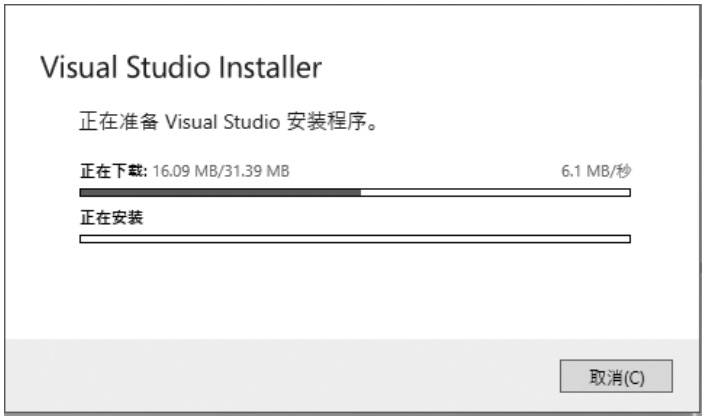

图 1.17　下载安装包文件的过程

(2) 下载完毕后，系统弹出如图1.18所示的选择安装应用程序界面。

图 1.18　选择安装应用程序界面

(3) 在选择安装应用程序界面中选择“ASP.NET和Web开发”，单击“安装”按钮，系统弹出如图1.19所示的安装过程。

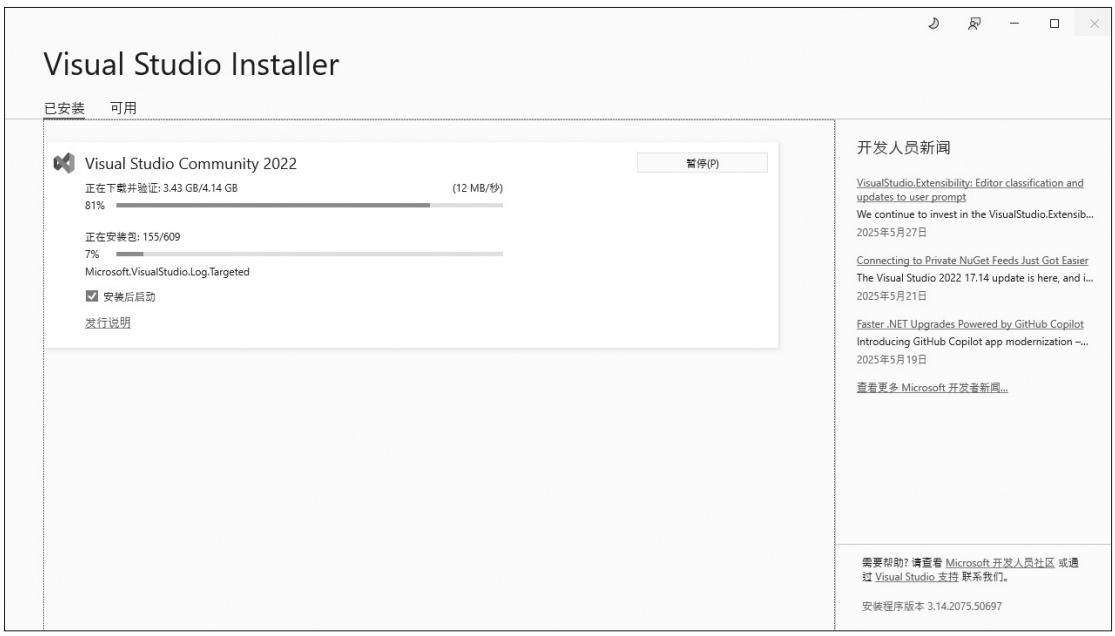

图 1.19　安装过程

(4) 安装完毕后，关闭安装程序窗口，重新启动Visual Studio 2022，将出现如图1.20所示的开始使用界面。

图 1.20　开始使用界面

(5) 选择“创建新项目”选项，打开“创建新项目”对话框，如图1.21所示。

图 1.21　“创建新项目”对话框

(6) 选择创建项目的类型。在搜索栏中输入“ASP.NET Web”，开发类型模板中将显示与ASP.NET Web类型相关的程序，如图1.22所示。

图 1.22　项目类型选择

(7) 选择开发程序为“ASP.NET Web应用程序(.NET Framework)”且语言为C#语言的ASP.NET Web应用程序，单击“下一步”按钮，打开“配置新项目”对话框，如图1.23所示。

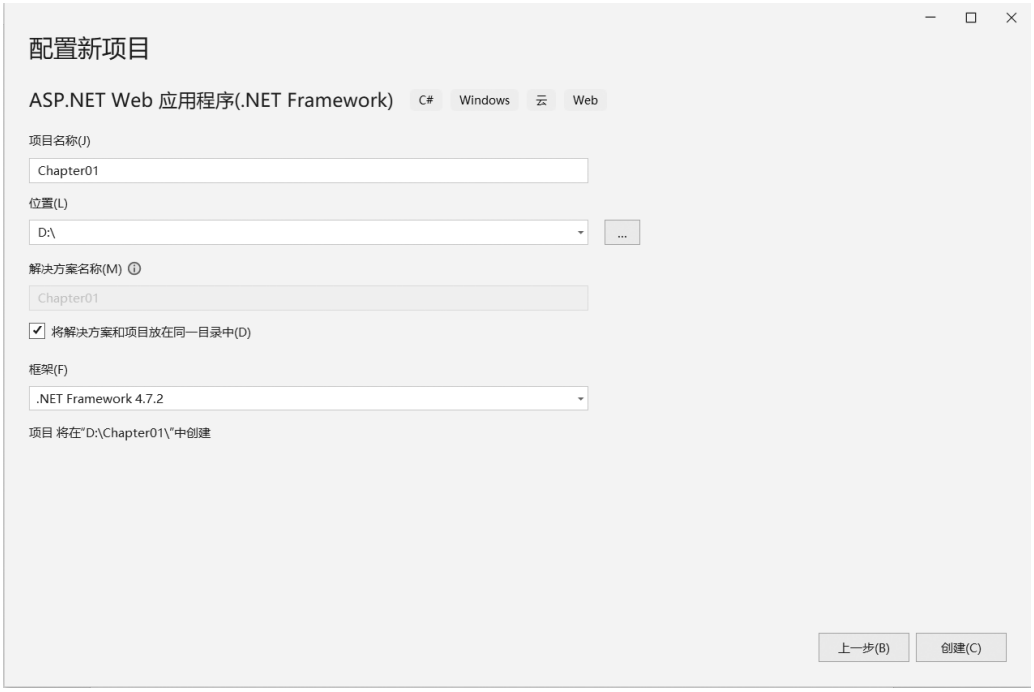

图 1.23　“配置新项目”对话框

(8) 在“配置新项目”对话框中，输入项目名称为“Chapter01”，在位置处选择D磁盘，并勾选“将解决方案和项目放在同一目录中”复选框，单击“创建”按钮，系统弹出“创建新的

ASP.NET Web应用程序”对话框(见图1.24)。

图 1.24　“创建新的 ASP.NET Web 应用程序”对话框

(9) 选择“空”，单击“创建”按钮，将创建一个新的ASP.NET Web应用程序空项目，如图1.25所示。

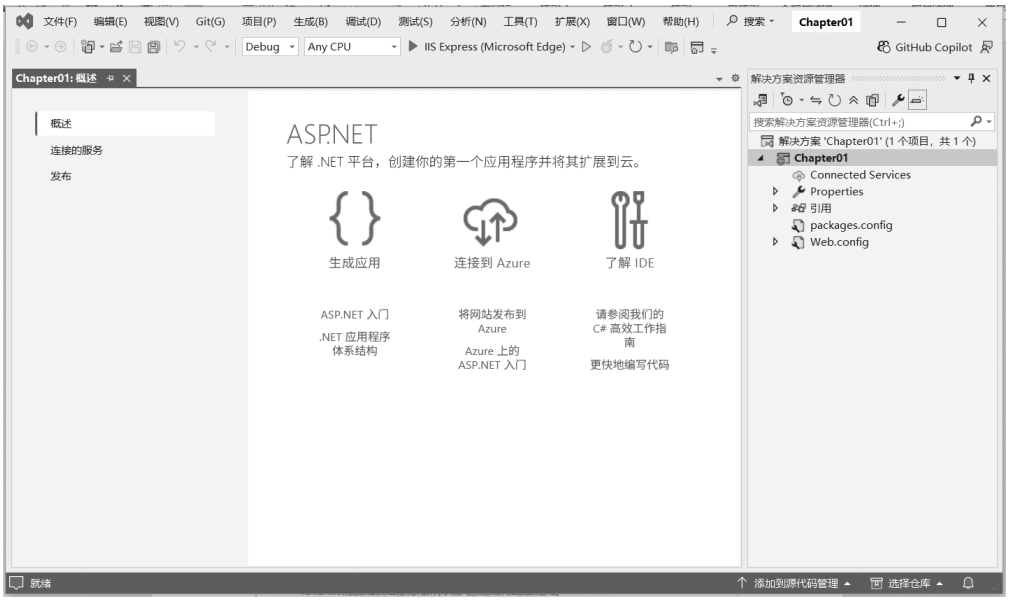

图 1.25　ASP.NET Web 应用程序空项目

说明：

创建ASP.NET Web应用程序项目的方式不止这一种，还有其他方式，请读者在学习过程中去摸索。

1.5　ASP.NET的开发模式

ASP.NET的开发模式有空项目模式、Web Forms模式、MVC模式、Web API模式和单页应用程序模式。下面主要介绍Web Forms模式和MVC模式。

1.5.1　Web Forms模式

Web Forms模式是传统的ASP.NET编程模式，它是集HTML、服务器控件和服务器代码事件驱动于一体的开发模型。Web Forms在服务器上编译和执行，再由服务器生成HTML显示为网页。Web Forms开发模式有数以百计的Web控件和Web组件，有助于提升页面开发效率。

Web Forms模式的工作原理：浏览器客户端向服务器发送窗体页面请求，服务器根据相应的后台代码文件进行业务逻辑处理，包括对数据库服务器的访问，最终将浏览器请求的窗体页面文件回传给浏览器并显示其内容。

本书主要讲解Web Forms开发模式，采用先创建空白解决方案，再创建ASP.NET Web应用程序的方式进行讲解。

1.5.2　MVC模式

MVC(model-view-controller，模型-视图-控制器)是一种程序架构模式，它能够强制性地将应用程序的输入、处理和输出分离开来。MVC将应用程序分为视图、模型、控制器3个核心部件，它们各自处理相应的任务。

(1) 视图。视图是用户看到并与之交互的界面。对Web应用程序来说，视图就是由HTML元素组成的界面。

(2) 模型。模型表示企业数据和业务规则，在MVC的3个部件中，模型承担着最多的处理任务。

(3) 控制器。控制器用于接收用户的输入并调用模型和视图去完成用户的需求，当用户单击页面中的超链接或提交HTML表单时，请求首先被控制器捕获。控制器本身不输出任何信息，也不执行业务处理，它只负责接收请求并决定调用哪个模型部件去处理请求，再确定用哪个视图来显示返回的数据。

1.6　ASP.NET Web项目的创建

1.6 ASP.NET Web 项目的创建

创建ASP.NET Web项目有两种方式：一种是创建ASP.NET Web应用程序项目；另一种是创建ASP.NET Web网站。本书均采用先创建空白解决方案，再在空白解决方案下添加相应ASP.NET Web项目的方式进行讲解。

1.6.1　创建ASP.NET Web应用程序项目

【例题1.1】创建一个ASP.NET Web应用程序的Web Forms项目，项目名称为Project1。ASP.

NET Web应用程序项目运行效果如图1.26所示。

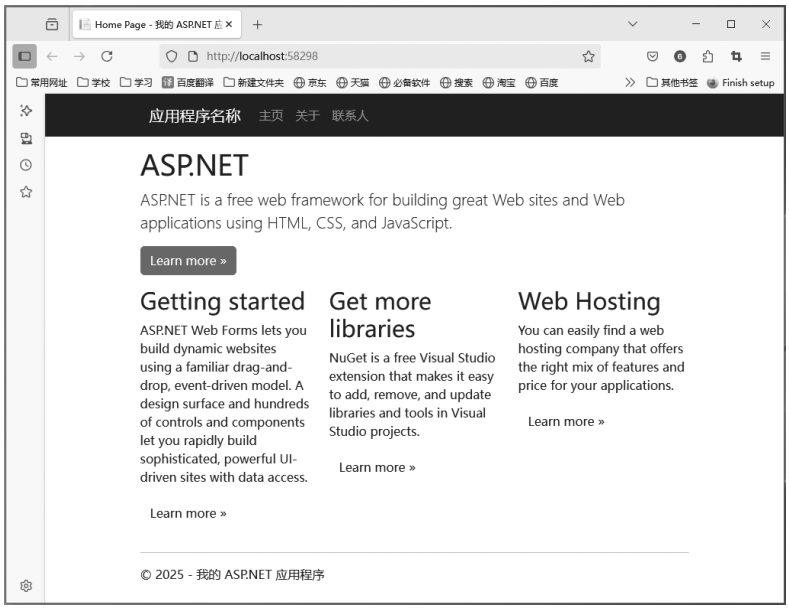

图 1.26　ASP.NET Web 应用程序项目运行效果

【实现步骤】

(1) 启动VS2022，在开始使用界面选择“创建新项目”选项，打开“创建新项目”对话框。

(2) 在“创建新项目”对话框的搜索栏中输入“空白解决方案”，在搜索结果中选择“空白解决方案”选项，如图1.27所示。单击“下一步”按钮，打开“配置新项目”对话框。

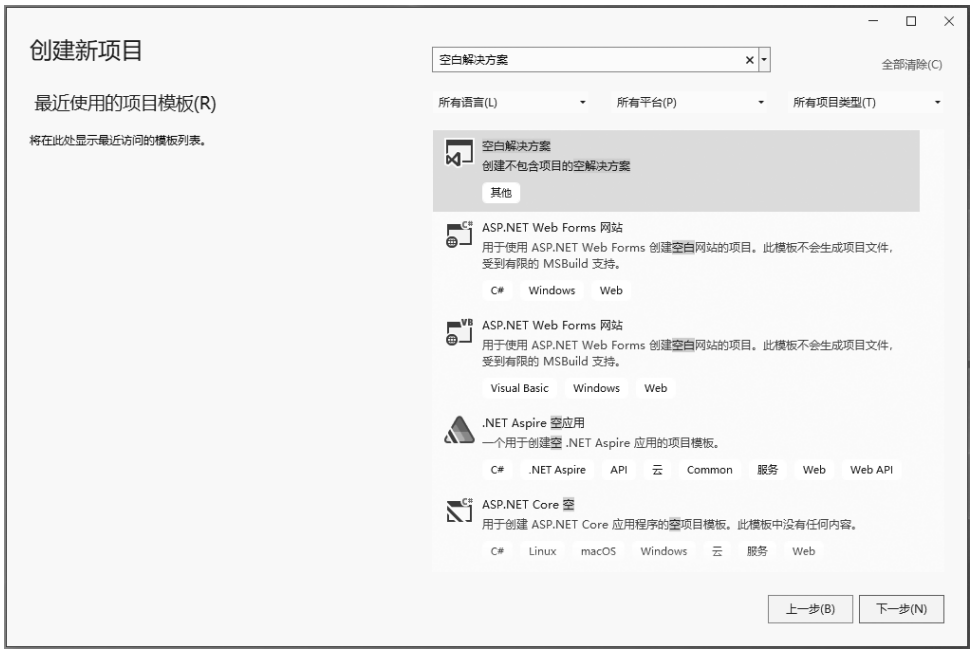

图 1.27　“创建新项目”对话框

(3) 在“配置新项目”对话框中，输入解决方案名称为“Chapter01”，选择保存路径，如

图1.28所示。单击“创建”按钮，完成空白解决方案的创建。

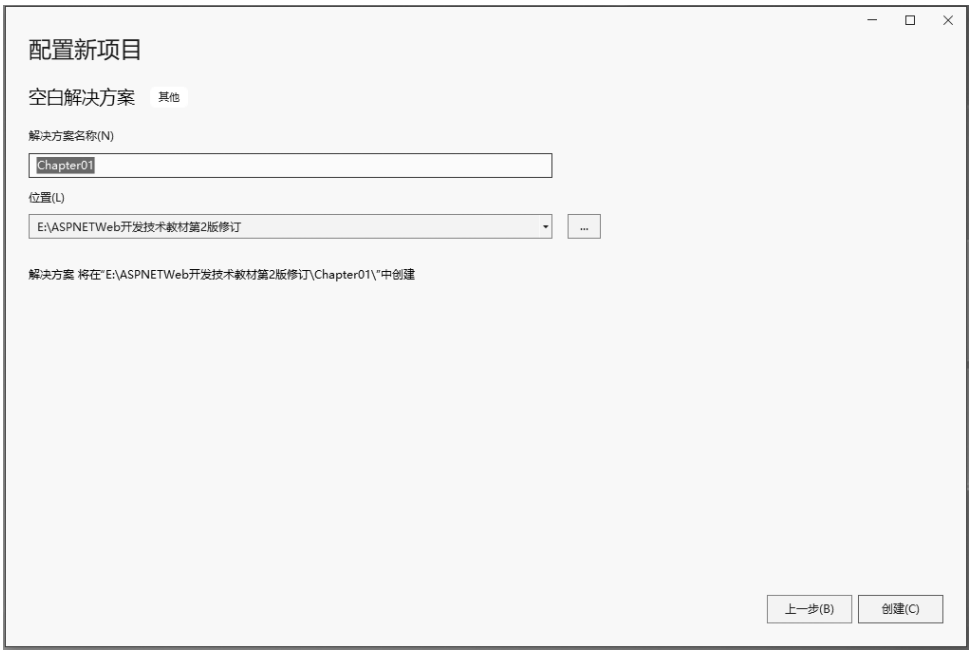

图 1.28　“配置新项目”对话框

(4) 右击已创建的解决方案Chapter01，执行“添加”|“新建项目”命令，打开“添加新项目”对话框。

(5) 在“添加新项目”对话框的搜索栏中输入“ASP.NET Web应用程序”，在搜索结果中选择开发项目类型为“ASP.NET Web应用程序(.NET Framework)”且开发语言是C#的选项，如图1.29所示。单击“下一步”按钮，打开“配置新项目”对话框。

图 1.29　“添加新项目”对话框

(6) 在“配置新项目”对话框中，输入项目名称为“Project1”，项目的保存位置默认为Chapter01，选择框架“.NET Framework 4.7.2”，如图1.30所示。单击“创建”按钮，打开“创建新的ASP.NET Web应用程序”对话框。

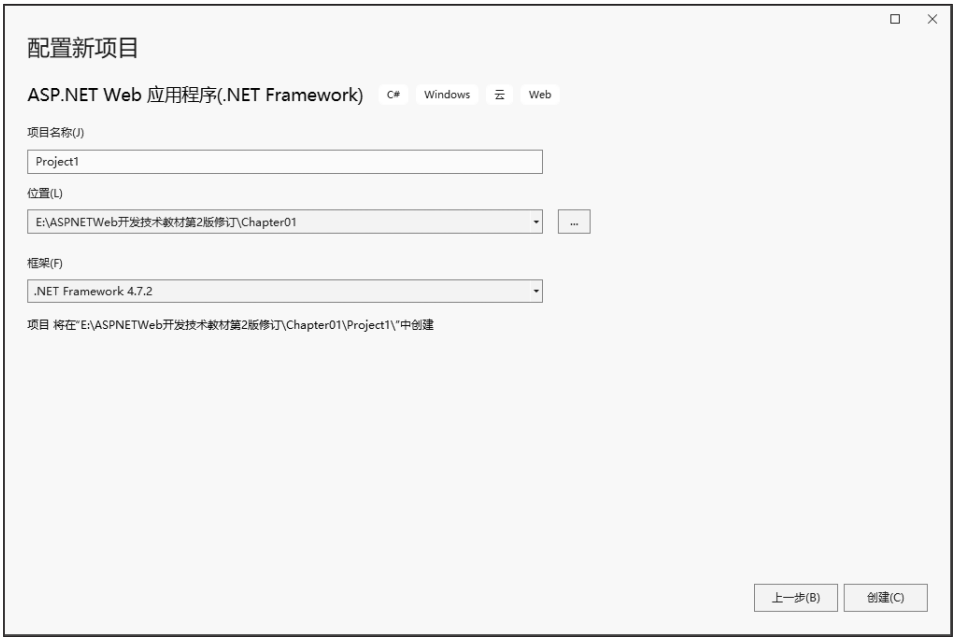

图 1.30 “配置新项目”对话框

(7) 选择项目类型为“Web Forms”，设置身份验证为“无”，在“添加文件夹和核心引用”处勾选“Web窗体”复选框，在“高级”处取消勾选“为HTTPS配置”复选框，如图1.31所示。单击“创建”按钮，完成Project1项目的创建。

图 1.31 “创建新的 ASP.NET Web 应用程序”对话框

(8) Project1项目的目录结构如图1.32所示。当使用Web Forms模板创建ASP.NET Web应用程序时，Visual Studio会自动创建一个名为Default.aspx的窗体页，默认该页为项目的主页，也是项目的启动页面，用户也可根据项目需求创建项目的功能页面。此外，还可将该程序改造成用户的功能程序，这需要读者去仔细体会。

Project1项目中定义了模板页文件、内容页文件、全局文件，页面样式运用了Bootstrap实现布局，同时还调用了JS，以及文件夹等内容，这些内容将在后续章节中学习。

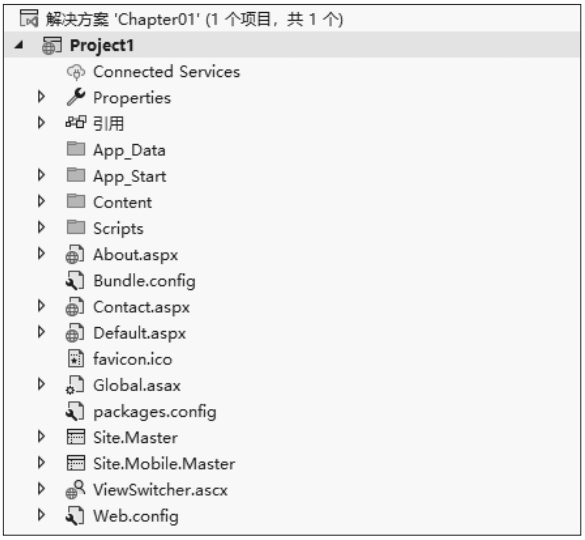

图 1.32　Project1 项目目录结构

(9) 编译运行程序：编译ASP.NET Web应用程序后就可以运行其中包含的页面。生成应用程序并运行Web窗体页的方法有以下3种。

第1种：使用调试器生成并运行Web窗体页。

在解决方案资源管理器中，右击要运行的Web窗体页，在弹出的菜单中执行“设为起始页”命令。在菜单栏中执行“调试”|“启动调试”命令，或者直接按F5键。该方式为重新编译后再运行，这样可以在程序代码中通过设置断点跟踪来调试程序。

第2种：不使用调试器生成并运行Web窗体。

在解决方案资源管理器中，右击要运行的Web窗体页，在弹出的菜单中执行“设为起始页”命令。在菜单栏中执行“调试”|“开发执行(不调试)”命令，或者直接按Ctrl+F5键。该方式直接运行生成的程序，不进行重新编译，因此运行速度较快。

第3种：在浏览器中查看生成并运行Web窗体页。

在解决方案资源管理器中，右击要预览的Web窗体页，在弹出的菜单中执行“在浏览器中查看”命令，Visual Studio会生成ASP.NET Web应用程序，并在默认浏览器中启动要预览的页面。

本项目中Default.aspx为起始页，因此直接单击工具栏中的启动按钮即可，程序运行效果如图1.26所示。

1.6.2　创建ASP.NET Web网站

【例题1.2】创建一个ASP.NET Web网站程序，程序名称为Project2，程序运行结果如图1.33所示。

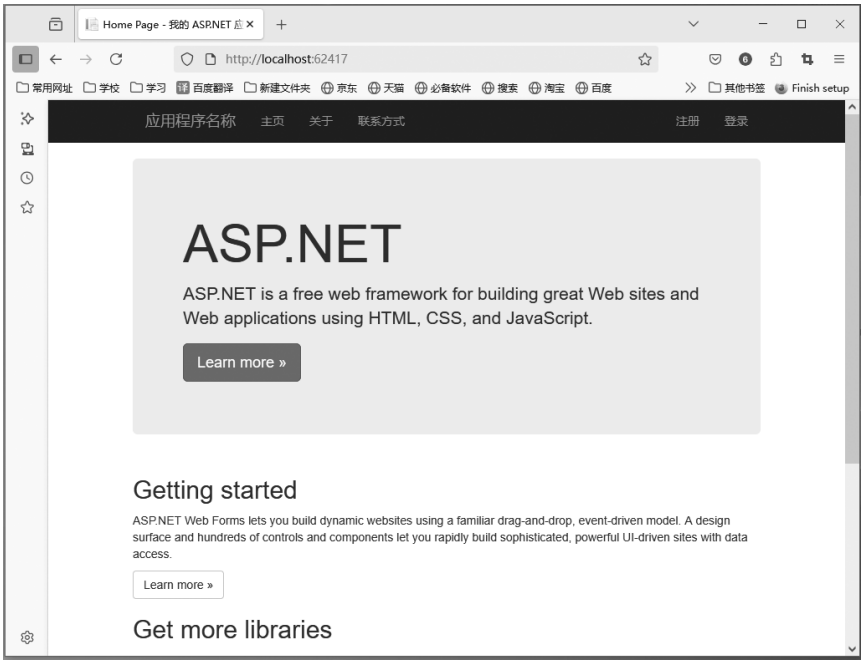

图 1.33　例题 1.2 程序运行结果

【实现步骤】

(1) 在已创建的Chapter01空白解决方案中，右击Chapter01，在弹出的菜单中执行“添加”|“新建项目”命令，打开“添加新项目”对话框。

(2) 在“添加新项目”对话框的搜索栏中输入“ASP.NET Web网站”，在搜索结果中选择开发项目类型为“ASP.NET Web Forms网站”且开发语言是C#的选项，如图1.34所示。单击“下一步”按钮，打开“配置新项目”对话框。

图 1.34　“添加新项目”对话框

(3) 在“配置新项目”对话框中，输入项目名称为“Project2”，位置和框架均为默认值，如图1.35所示。单击“创建”按钮，即可在解决方案Chapter01下创建网站Project2，其目录结构如图1.36所示。

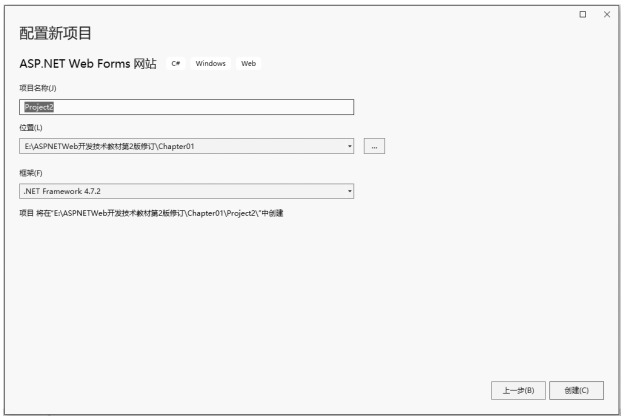

图 1.35　“配置新项目”对话框

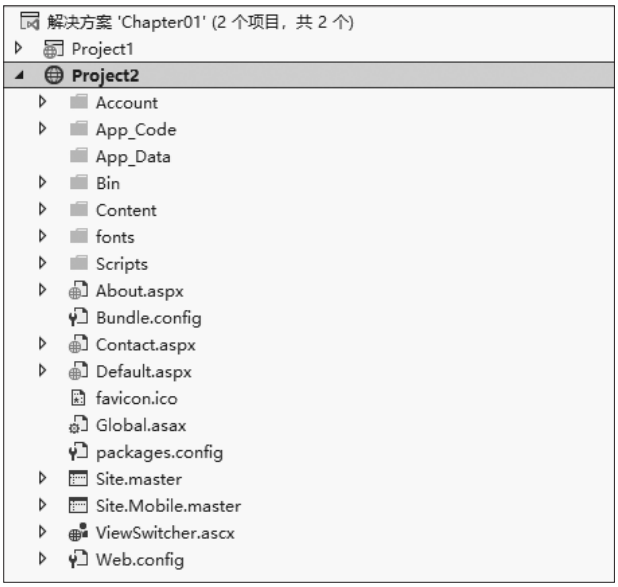

图 1.36　Project2 网站目录结构

在Project2网站的目录结构中，默认的起始页是Default.aspx，与ASP.NET Web应用程序相比，该网站目录下多了Account、Bin和fonts文件夹。创建的ASP.NET Web网站和ASP.NET Web应用程序均为一类开发模板，模板中集成了相关内容，用户可根据实际项目需求对其进行相应的改造，从而快速构建自己的项目。

(4) 编译运行程序，将Project2项目设为启动项目，其起始页为Default.aspx，因此直接单击工具栏中的启动按钮即可，程序运行结果如图1.33所示。

1.6.3　创建ASP.NET Web空应用程序

【例题1.3】创建一个ASP.NET Web空应用程序Project3，并在程序中添加名为userLogin.aspx

的用户登录页面，页面元素包括标签、文本框、按钮。程序功能为在文本框中输入信息后，单击“登录”按钮，即可在页面的指定位置显示相应的信息，程序运行结果如图1.37所示。

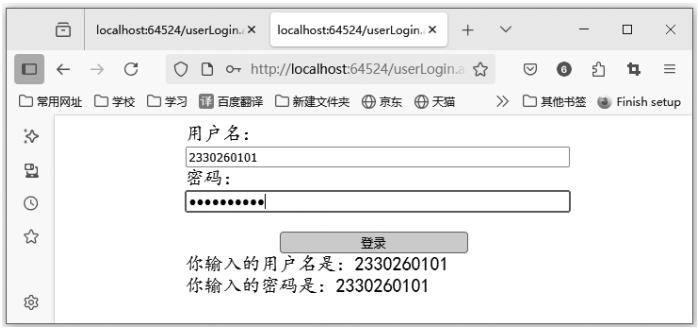

图 1.37　例题 1.3 程序运行结果

【实现步骤】

(1) 在空白解决方案Chapter01中，创建新项目Project3，在“创建新的ASP.NET Web应用程序”对话框中，选择项目类型为“空”，取消勾选“为HTTPS配置”复选框，单击“创建”按钮，即可完成ASP.NET Web空应用程序的创建。该项目只提供了项目所需的引用、Web.config配置文件和工具，需要用户添加新的页面来实现相应功能。

(2) 右击Project3，执行“添加”|“新建项”命令，打开 “添加新项”对话框。

(3) 在“添加新项”对话框的左侧，执行“C#”|“Web”命令，在中间模板中选择“Web窗体”，在名称处输入“userLogin.aspx”，如图1.38所示。单击“添加”按钮，即可完成ASP.NET Web空应用程序页面结构的创建。

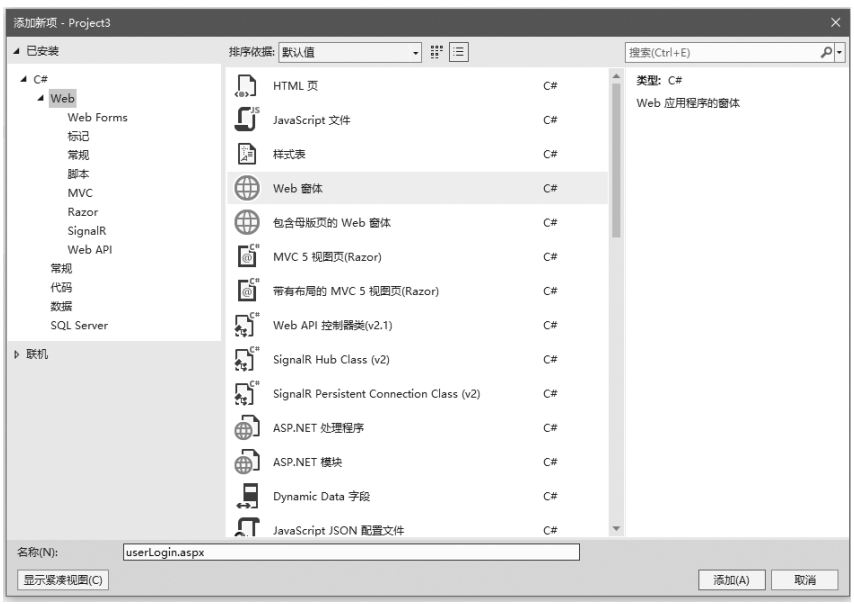

图 1.38　“添加新项”对话框

ASP.NET Web空应用程序页面由页面指令<%@ Page ···%>、<!DOCTYPE html>文档、<html>标签、<head>标签和<body>标签构成。其中，<body>标签中的所有元素封装在<form>表单内。

(4) 前端页面userLogin.aspx设计。代码设计参考如下。

```
<body>
    <form id="form1" runat="server">
        <div style="width:200px;margin:auto;font-family:楷体;font-size:20px">
            <div>用户名：</div>
            <div>
                <asp:TextBox ID="txtName" runat="server" Width="100%"></asp:TextBox>
            </div>
            <div>密码：</div>
            <div>
                <asp:TextBox ID="txtPwd" runat="server" TextMode="Password" Width="100%">
                            </asp:TextBox>
            </div>
            <div style="text-align:center">
                <asp:Button ID="btnLogin" runat="server" Text="登录" Width="100%" /></div>
            <div>
                <asp:Label ID="lblMessage" runat="server" Text=""></asp:Label></div>
        </div>
    </form>
</body>
```

(5) 后台功能逻辑实现。登录按钮“单击”事件代码设计如下。

```
protected void btnLogin_Click(object sender, EventArgs e)
 {
     lblMessage.Text = "你输入的用户名是：" + txtName.Text + "</br>"
                       + "你输入的密码是：" + txtPwd.Text;
 }
```

(6) 编译运行程序。右击Project3，执行“生成”|“重新生成”命令，检查后台功能逻辑代码语法的正确性，可通过输出窗口查看结果。在userLogin.aspx的设计视图下右击，执行“在浏览器中查看”命令，在浏览器的用户名文本框中输入2330260101，在密码文本框中输入2330260101，单击“登录”按钮，程序运行结果如图1.37所示。

1.7　上机实验

1. 实验目的

(1) 熟悉ASP.NET Web的开发环境Visual Studio 2022。

(2) 掌握利用解决方案管理网站和创建网站的基本过程。

(3) 掌握运用静态页面与动态页面技术设计用户登录页面的基本方法。

(4) 掌握在IIS中创建网站、Web应用程序、虚拟目录，以及设置默认文档的过程。

(5) 掌握利用Visual Studio 2022发布Web应用的过程。

2. 实验内容

创建一个名为“Experiment1_学号姓名”的解决方案，在该解决方案中分别添加IndexUserLogin.html静态页面和IndexUserLogin.aspx动态页面。在动态页面中实现用户名和密码的输入功能，并将输入的信息显示在浏览器页面上。动态页面运行效果如图1.39所示。

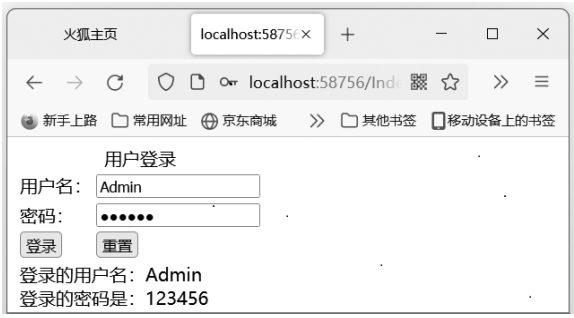

图 1.39　动态页面运行效果

3. 实验步骤

1) 实验内容分析

网站名称为“Experiment1_学号姓名”，在该网站下有两个互相独立的页面。页面中需要设计用户名和密码输入框，以及两个按钮，静态页面和动态页面所用的控件有所区别。信息显示可运用C#语言课程所学知识，通过标签来实现。

2) 实验过程

启动Visual Studio 2022，创建“Experiment1_学号姓名”解决方案，按实验内容添加页面。IndexUserLogin.html静态页面代码设计如下。

```
<body>
    <table>
        <caption>用户登录</caption>
        <tr>
            <td>用户名：</td>
            <td><input id="txtName" type="text" /></td>
        </tr>
        <tr>
            <td>密码：</td>
            <td><input id="txtPwd" type="password" /></td>
        </tr>
        <tr>
            <td><input id="btnLogin" type="button" value="登录" /></td>
            <td><input id=" btnReset" type="reset" value="重置" /></td>
        </tr>
    </table>
</body>
```

IndexUserLogin.aspx动态页面代码设计如下。

```
<body>
    <form id="form1" runat="server">
    <div>
     <table>
            <caption >用户登录</caption>
            <tr>
                <td>用户名：</td>
                <td> <asp:TextBox ID="txtName" runat="server"></asp:TextBox></td>
            </tr>
            <tr>
                <td>密码：</td>
                <td>
            <asp:TextBox ID="txtPwd" runat="server" TextMode
                          ="Password"></asp:TextBox></td>
            </tr>
```

```
            <tr>
                <td>
                    <asp:Button ID="btnLogin" runat="server" Text="登录"
                                OnClick="btnLogin_Click" /></td>
                <td>
                    <asp:Button ID="btnReset" runat="server" Text="重置" /></td>
            </tr>
            <tr>
                <td colspan="2">
                    <asp:Label ID="lblShow" runat="server" Text=""></asp:Label></td>
            </tr>
        </table>
    </div>
    </form>
</body>
```

IndexUserLogin.aspx动态页面的“登录”按钮后台代码设计如下。

```
namespace Experiment1_学号姓名
{
    public partial class IndexUserLogin : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void btnLogin_Click(object sender, EventArgs e)
        {
            if (txtName.Text == "Admin" && txtPwd.Text == "123456")
            {
                lblShow.Text += "登录的用户名：" + txtName.Text + "<br/>";
                lblShow.Text += "登录的密码是：" + txtPwd.Text;
            }
            else
            {
                lblShow.Text = "登录的用户名或密码错误！";
            }
        }
    }
}
```

4. 实验分析

实验后，主要进行以下几方面的分析：比较静态页面和动态页面的结构差异；分析静态页面和动态页面实现用户交互功能的方法；探究Web工作原理在静态页面和动态页面中的具体体现；理解ASP.NET引擎的工作原理。