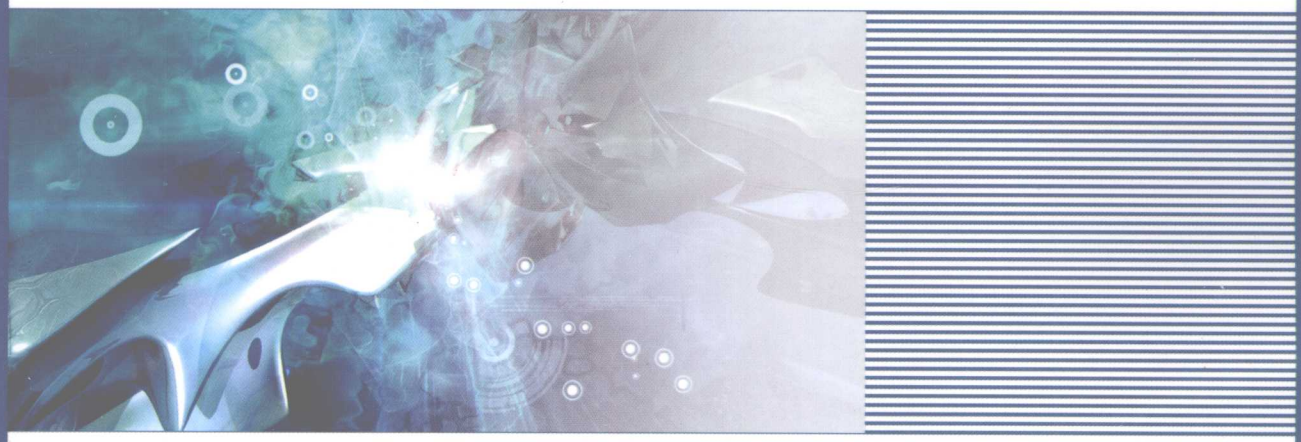
21世纪高等院校网络工程规划教材

21st Century University Planned Textbooks of Network Engineering



ASP.NET 网络程序设计教程

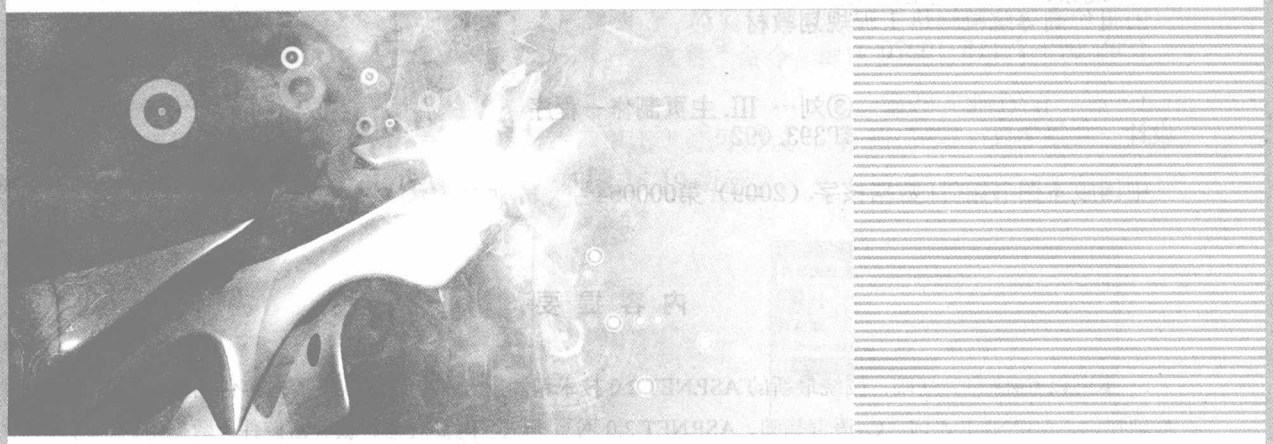
ASP.NET Web Programming

张恒 廖志芳 刘艳丽 编著

- 零基础入门，由浅入深，循序渐进
- 理论与实践相结合，突出思想、方法和技巧
- 面向应用，实例丰富，切实提高实际动手能力

人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等院校网络工程规划教材
21st Century University Planned Textbooks of Network Engineering



ASP.NET 网络程序设计教程

ASP.NET Web Programming

张恒 廖志芳 刘艳丽 编著

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

ASP.NET网络程序设计教程 / 张恒，廖志芳，刘艳丽编著. —北京：人民邮电出版社，2009.2
21世纪高等院校网络工程规划教材
ISBN 978-7-115-19270-7

Ⅰ. A… Ⅱ. ①张…②廖…③刘… Ⅲ. 主页制作－程序设计－高等学校－教材 Ⅳ. TP393.092

中国版本图书馆CIP数据核字（2009）第000054号

内 容 提 要

本书以C#语言为基础，围绕最新的ASP.NET 2.0技术精髓展开深入讲解，主要内容包括网络程序设计基础知识、ASP.NET入门、C#语言基础、ASP.NET 2.0网页语法、内置对象、服务器控件、数据访问、用户控件与自定义控件、样式和主题、安全技术、站点导航、使用XML以及网站发布及安装等。

本书结构合理、条理清晰、实用性强，从第3章开始，每一个技术的讲解都附有具体的实例，可供读者实际操作使用。此外，每章都附有习题，供课后练习和上机实验。

本书可以作为高等院校计算机科学与技术、网络工程、电子信息等相关专业“ASP.NET网络程序设计”课程的教材，也可供从事Web程序设计相关工作的技术人员自学参考。

21世纪高等院校网络工程规划教材

ASP.NET网络程序设计教程

◆ 编　　著　张　恒　廖志芳　刘艳丽
责任编辑　滑　玉
执行编辑　张　鑫

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街14号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市海波印务有限公司印刷

◆ 开本：787×1092　1/16
印张：18.75
字数：466千字　　2009年2月第1版
印数：1－3 000册　　2009年2月河北第1次印刷

ISBN 978-7-115-19270-7/TP

定价：31.00元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

随着计算机网络的飞速发展和日益普及，网络应用越来越多，面向网络的开发技术已经成为IT技术发展的重要分支之一。网络应用程序的设计和开发已经成为各类应用软件中最主要的组成部分，其需求也相应增多，因此计算机网络程序设计作为一项知识技能受到越来越多的重视。在众多提供动态内容的Web编程技术中，ASP.NET独立于浏览器，采用效率较高的、面向对象的方法来创建动态Web应用程序，而且支持VB.NET、C#.NET、VC++.NET等多种编程语言。

本书以C#语言为基础，紧紧围绕最新的ASP.NET 2.0技术精髓展开深入讲解，以清晰的思路、精炼的实例使读者快速入门，并逐步掌握网络编程的知识。本书注重基础理论与实用开发相结合，突出应用编程思想与开发方法的介绍，所选实例都具有较强的概括性和实际应用价值。

本书是作者根据多年从事网络程序设计工作和讲授计算机专业相关课程的教学实践，在已编多部讲义和教材的基础上编写而成的；内容充实，循序渐进，选材上注重系统性、先进性和实用性；注重实践性，精选大量例题，且增加了汉字注释，所有例题已在Visual Studio 2005上调试通过，可直接引用，读者也可按照书中提示步骤自己动手完成。

本书分为4部分。第1部分（第1、2、3章）为基础知识篇，包括网络程序设计概述、ASP.NET 2.0入门和C#程序设计语言基础；第2部分（第4、5、6、7、8章）为ASP.NET 2.0核心技术篇，包括ASP.NET 2.0网页语法、ASP.NET 2.0内置对象、ASP.NET 2.0常用服务器控件、ADO.NET数据访问、数据绑定；第3部分（第9、10、11章）为ASP.NET 2.0技术提高篇，包括用户控件与自定义控件、样式和主题、ASP.NET应用程序安全技术；第4部分（第12、13、14、15章）为系统集成开发篇，包括站点导航控件、ASP.NET中使用XML、网站实例及网站发布。

本书第1章由廖志芳编写，第2、7、8、12、13由张恒编写，第3、9、10和11章由刘艳丽编写，第4、5、6章由王长征编写，第14、15章由周洁编写，全书由张恒统稿。在此特别感谢中南大学樊晓平教授和华东交通大学刘觉夫教授对本书内容提出的宝贵意见。

由于编者水平有限，加之编写时间仓促，书中难免存在错误和疏漏之处，希望广大读者批评指正。

编　者

2008年11月

目　录

第 1 章　网络程序设计概述 ……1
1.1　Internet 概述 ……1
1.2　三类网络程序设计 ……2
1.2.1　基于 TCP/IP 协议栈的网络编程 ……2
1.2.2　基于 WWW 应用的网络编程 ……2
1.2.3　基于.NET 框架的 Web Services 网络编程……2
1.3　Web 编程概述 ……3
1.3.1　Web 的工作原理 ……3
1.3.2　动态 Web 开发技术概述 ……6
1.4　HTML 基础 ……9
1.4.1　HTML 标记 ……10
1.4.2　HTML 文档的基本结构 ……11
1.4.3　一些常用的 HTML 标记 ……12
1.4.4　用 Visual Studio 2005 编辑 HTML 文档 ……13
1.4.5　XHTML 文件 ……14
1.5　XML 基础 ……15
1.5.1　XML 文档的基本结构 ……15
1.5.2　使用 XML 的特点 ……18
1.5.3　HTML 与 XML 的比较 ……19
1.6　小结 ……19
习题 ……19
第 2 章　ASP.NET 2.0 入门 ……20
2.1　ASP.NET 2.0 简介 ……20
2.1.1　NET 2.0 框架体系结构概述 ……20
2.1.2　ASP.NET 2.0 的功能介绍 ……22
2.1.3　ASP.NET 与 ASP 的区别 ……23
2.2　ASP.NET 2.0 开发环境的安装与配置 ……23
2.2.1　IIS 的安装与配置 ……23
2.2.2　ASP.NET 2.0 开发环境 ……27
2.3　ASP.NET 2.0 应用程序的框架 ……28
2.3.1　ASP.NET Web 窗体代码模型 ……28
2.3.2　ASP.NET Web 窗体事件模型 ……31
2.3.3　ASP.NET 2.0 编译模型 ……33
2.4　ASP.NET 2.0 网站设计步骤 ……34
2.4.1　创建 ASP.NET 网站 ……34
2.4.2　创建 Web 窗体 ……35
2.4.3　设计 Web 窗体界面 ……35
2.4.4　编写代码 ……35
2.4.5　运行程序 ……36
2.5　小结 ……36
习题 ……36
第 3 章　C#语言基础 ……37
3.1　创建一个简单的 C#程序 ……37
3.2　C#数据类型 ……38
3.2.1　值类型 ……38
3.2.2　引用类型 ……40
3.2.3　装箱与拆箱 ……42
3.3　变量与常量 ……43
3.3.1　变量 ……43
3.3.2　常量 ……44
3.4　流程控制 ……44
3.4.1　分支语句 ……44
3.4.2　循环结构 ……45
3.5　运算符 ……47
3.5.1　算术运算符 ……47
3.5.2　赋值运算符 ……48
3.5.3　关系运算符 ……49
3.5.4　逻辑运算符 ……49
3.5.5　条件运算符 ……50
3.5.6　位运算符 ……50
3.5.7　运算符的优先级 ……51
3.6　字符串处理 ……51

3.6.1 使用 string 和 StringBuilder……51
3.6.2 格式化字符串……52
3.6.3 对字符串进行编码……53
3.7 类和结构……53
3.7.1 定义类和结构……53
3.7.2 定义属性……55
3.7.3 定义索引器……55
3.7.4 重载方法……56
3.7.5 使用 Ref 和 Out 类型参数……56
3.7.6 定义接口和抽象类……57
3.8 使用集合编程……58
3.8.1 使用枚举……58
3.8.2 使用数组……59
3.8.3 使用 ArrayList……60
3.8.4 使用哈希表……60
3.8.5 使用字典……61
3.8.6 使用堆栈……62
3.8.7 使用队列……62
3.9 小结……63
习题……63
第 4 章 ASP.NET 2.0 网页语法……64
4.1 页面指令……64
4.1.1 什么是页面指令……64
4.1.2 页面指令种类与作用……64
4.2 ASPX 文件内容注释……70
4.3 包含服务器端文件……71
4.4 HTML 服务器控件语法……72
4.5 ASP.NET 服务器（控件）语法……73
4.6 代码块语法<% %>……73
4.7 数据绑定语法……74
4.8 对象标记语法……75
4.9 表达式语法……76
4.10 小结……76
习题……77
第 5 章 ASP.NET 2.0 内置对象……78
5.1 Response 对象……78
5.1.1 Response 对象概述……78
5.1.2 Response 对象常用属性和方法……78
5.1.3 Response 对象在实际开发中的应用……80
5.2 Request 对象……81
5.2.1 Request 对象概述……81
5.2.2 Request 对象常用属性和方法……81
5.2.3 Request 对象在实际开发中的应用……83
5.3 Application 对象……83
5.3.1 Application 对象概述……83
5.3.2 Application 对象常用集合、属性和方法……84
5.3.3 Application 对象在实际开发中的应用……87
5.4 Session 对象……88
5.4.1 Session 对象概述……88
5.4.2 Session 对象常用集合、属性和方法……88
5.4.3 Session 对象在实际开发中的应用……90
5.5 Cookie 对象……91
5.5.1 Cookie 对象概述……91
5.5.2 Cookie 对象常用属性和方法……92
5.5.3 Cookie 对象在实际开发中的应用……93
5.6 Server 对象……94
5.6.1 Server 对象概述……94
5.6.2 Server 对象常用属性和方法……94
5.6.3 Server 对象在实际开发中的应用……96
5.7 Cache 对象……97
5.7.1 Cache 对象概述……97
5.7.2 Cache 对象常用属性和方法……97
5.7.3 Cache 对象在实际开发中的应用……99

5.8 小结 ……100
习题……100
第 6 章 ASP.NET 2.0 常用服务器控件……101
6.1 服务器控件概述 ……101
6.1.1 服务器控件的概念与作用……101
6.1.2 服务器控件与对象的关系……102
6.1.3 服务器控件的属性、方法和事件 ……102
6.2 常用服务器控件 ……104
6.2.1 Label 控件 ……104
6.2.2 TextBox 控件……105
6.2.3 Image 控件 ……107
6.2.4 Button 控件 ……109
6.2.5 LinkButton 控件……111
6.2.6 ImageButton 控件 ……112
6.2.7 HyperLink 控件……113
6.2.8 RequiredFieldValidator 控件……114
6.2.9 RangeValidator 控件 ……115
6.2.10 RegularExpression Validator 控件……116
6.2.11 CompareValidator 控件 ……118
6.2.12 CustomValidator 控件…120
6.2.13 ValidationSummary 控件 ……121
6.2.14 Login 控件……122
6.3 常用服务器控件综合实例 ……124
6.3.1 实例页面设计 ……124
6.3.2 实例界面设计 ……124
6.3.3 主要模块编码 ……125
6.4 小结 ……126
习题……126
第 7 章 ADO.NET 数据访问……127
7.1 ADO.NET 概述……127
7.1.1 ADO.NET 简介……127
7.1.2 ADO.NET 的体系结构 …128
7.1.3 数据库应用程序的开发流程……129
7.2 Connection 对象 ……129
7.2.1 Connection 对象概述 ……130
7.2.2 连接字符串……131
7.2.3 使用 Connection 对象连接数据库……132
7.2.4 连接池……137
7.3 Command 对象……138
7.3.1 Command 对象概述 ……138
7.3.2 创建和使用 Command 对象……140
7.4 DataReader 对象……143
7.4.1 DataReader 对象概述……143
7.4.2 创建和使用 DataReader 对象……144
7.5 DataSet 和 DataAdapter 对象 …145
7.5.1 DataSet 对象概述 ……145
7.5.2 DataSet 对象的基本结构……146
7.5.3 DataAdapter 对象 ……146
7.5.4 使用 DataAdapter 填充数据集……148
7.5.5 DataSet 数据更新 ……148
7.6 小结……149
习题……149
第 8 章 数据控件与数据绑定技术……150
8.1 数据源控件……150
8.1.1 SqlDataSource 控件……151
8.1.2 AccessDataSource 控件…158
8.1.3 ObjectDataSource 控件……158
8.1.4 XmlDataSource 控件……159
8.1.5 SiteMapDataSource 控件……160
8.2 数据绑定……160
8.2.1 简单数据绑定和复杂数据绑定……160
8.2.2 绑定到简单的数据源……161
8.3 GridView 控件……162

8.3.1 GridView 控件概述 ……162
8.3.2 GridView 控件常用的属性、方法和事件 ……166
8.3.3 使用 GridView 控件分页显示数据 ……170
8.3.4 使用 GridView 控件实现数据排序 ……172
8.3.5 使用 GridView 控件实现主/详细页 ……174
8.3.6 使用 GridView 控件更新数据……176
8.3.7 使用 GridView 控件删除数据……180
8.4 DetailsView 控件……181
8.4.1 DetailsView 控件概述 …181
8.4.2 DetailsView 控件常用的属性、方法和事件 ……183
8.4.3 使用 DetailsView 控件分页显示数据 ……185
8.4.4 使用 DetailsView 控件更新数据 ……186
8.5 FormView 控件 ……187
8.5.1 FormView 控件概述 ……187
8.5.2 为 FormView 控件创建模板……188
8.5.3 使用 FormView 控件中分页显示数据 ……189
8.5.4 使用 FormView 控件修改数据……191
8.6 小结 ……193
习题……193
第 9 章 用户控件与自定义控件 ……194
9.1 ASP.NET 用户控件和自定义控件概述……194
9.1.1 用户控件和普通的 Web 页比较 ……194
9.1.2 用户控件与自定义控件的比较 ……194
9.1.3 用户控件的优点 ……195
9.2 创建及使用用户控件 ……195
9.2.1 创建用户控件…… 195
9.2.2 将用户控件添加至网页 …… 196
9.2.3 在用户控件中添加用户控件…… 196
9.3 设置用户控件…… 197
9.3.1 访问用户控件的属性…… 197
9.3.2 访问用户控件中的服务器控件…… 197
9.3.3 将 Web 网页转化为用户控件 …… 199
9.4 用户控件编程…… 200
9.4.1 用户控件设计…… 200
9.4.2 界面设计…… 200
9.4.3 事件设计…… 201
9.4.4 运行情况…… 202
9.5 自定义控件…… 202
9.6 小结…… 204
习题…… 204
第 10 章 样式和主题…… 205
10.1 母版页和内容页…… 205
10.1.1 创建母版页…… 205
10.1.2 创建内容页…… 207
10.1.3 以编程方式访问母版页 …… 208
10.2 样式表 CSS…… 208
10.2.1 什么是 CSS…… 208
10.2.2 CSS 的基本语法…… 209
10.2.3 将 CSS 应用在 Web 控件上…… 210
10.3 主题和外观……211
10.3.1 主题和外观概述……211
10.3.2 创建主题和外观…… 212
10.3.3 应用主题和外观…… 212
10.4 小结…… 215
习题…… 215
第 11 章 ASP.NET 应用程序安全技术…… 216
11.1 ASP.NET 安全结构 …… 216
11.2 基于 Windows 的身份验证…… 217

11.2.1 IIS 和 ASP.NET 中的安全和访问控制 ……217
11.2.2 配置 Windows 身份验证与授权 ……218
11.3 ASP.NET 2.0 的成员资格和角色资格管理器 ……219
11.3.1 ASP.NET 2.0 成员资格概述 ……219
11.3.2 ASP.NET 2.0 成员资格的配置 ……220
11.3.3 ASP.NET 2.0 角色管理器配置 ……222
11.4 ASP.NET 网站管理工具……223
11.4.1 ASP.NET 网站管理工具概述 ……223
11.4.2 用户管理 ……224
11.4.3 角色管理 ……225
11.4.4 访问规则管理 ……226
11.5 ASP.NET 安全服务器控件……227
11.5.1 登录控件 ……227
11.5.2 创建用户向导控件 ……227
11.5.3 密码恢复控件 ……228
11.5.4 修改密码控件 ……228
11.5.5 其他控件 ……228
11.6 成员资格和角色特性 ……229
11.6.1 Membership 类 ……229
11.6.2 MembershipUser 类……230
11.6.3 Role 类介绍 ……230
11.7 小结……231
习题……231
第 12 章 站点导航控件 ……232
12.1 站点地图概述 ……232
12.2 TreeView 控件……233
12.2.1 TreeView 控件概述……233
12.2.2 TreeView 控件常用的属性和事件 ……234
12.2.3 TreeView 控件的基本应用 ……236
12.2.4 TreeView 控件绑定数据库 ……238
12.2.5 TreeView 控件绑定 XML 文件……240
12.3 Menu 控件 ……242
12.3.1 Menu 控件概述 ……242
12.3.2 Menu 控件常用的属性和事件……242
12.3.3 Menu 控件的基本应用 ……244
12.3.4 Menu 控件绑定 XML 文件……245
12.4 SiteMapPath 控件 ……246
12.4.1 SiteMapPath 控件概述……246
12.4.2 SiteMapPath 控件常用的属性和事件……247
12.4.3 应用 SiteMapPath 控件实现站点导航……248
12.5 小结……250
习题……250
第 13 章 ASP.NET 中使用 XML ……251
13.1 .NET Framework 2.0 中 XML 命名空间 ……251
13.2 读取 XML 文档……251
13.2.1 使用 XmlReader 读取 XML 文件……252
13.2.2 使用 XmlDocument 读取 XML 文件……256
13.3 生成和修改 XML 文档……259
13.3.1 使用 XmlWriter 生成 XML……259
13.3.2 使用 XmlDocument 创建 XML……262
13.3.3 使用 XmlDocument 修改 XML……263
13.4 使用 XSLT 转换 XML ……267
13.4.1 System.Xml.Xsl 命名空间下的类……267
13.4.2 直接使用 XSLT 转换 XML 文件……268
13.4.3 传递参数至 XSL

样式表 ……269
13.5 XML 与 DataSet 的交互……269
13.6 小结 ……270
习题……270
第 14 章 综合应用实例——留言板 ……271
14.1 系统功能概述 ……271
14.2 数据库设计 ……271
14.3 公共类编写 ……273
14.3.1 配置 Web.Config……273
14.3.2 SqlData 类 ……273
14.4 母版页的设计 ……275
14.4.1 母版页的创建 ……275
14.4.2 母版页的运行 ……276
14.4.3 母版页和内容页路径 ……276
14.5 首页设计 ……276
14.5.1 技术难点 ……276
14.5.2 功能实现 ……276
14.6 发表留言模块的设计 ……277
14.6.1 技术难点 ……277
14.6.2 功能实现 ……278
14.7 留言信息查看模块的设计……279
14.7.1 技术难点……279
14.7.2 功能实现……279
14.8 留言管理模块设计……281
14.8.1 技术难点……281
14.8.2 功能实现……282
14.9 回复留言功能设计……282
14.9.1 技术难点……282
14.9.2 功能实现……283
14.10 小结……283
习题……283
第 15 章 网站发布、打包与安装 ……284
15.1 发布网站……284
15.2 打包和安装……286
15.2.1 打包和安装网站……286
15.2.2 Web 安装项目细节问题……287
15.3 小结……289
习题……289
参考文献……290

第 1 章　网络程序设计概述

随着计算机网络的飞速发展和日益普及，网络应用越来越多，对计算机网络程序设计的需求也相应增多，因此，计算机网络程序设计作为一种重要的知识与技能越来越受到重视。本书以 ASP.NET 2.0 为框架讲解计算机网络程序设计，在深入介绍之前，有必要了解一些网络程序设计的基本内容、概念和方法等基础知识。

1.1　Internet 概述

Internet 的前身是美国国防部高级研究计划局（ARPA）于 1968 年主持研制的用于支持军事研究的计算机试验网络——阿帕网（ARPAnet）。建设该网的初衷旨在帮助美国军方的研究人员利用计算机进行信息交换。ARPAnet 的设计与实现的主导思想是网络应能够经受住故障的考验并能维持正常工作，当网络的某个部分遭受敌方攻击、摧毁而失去作用时，也能保证网络其他部分运行并仍能维持正常通信。

随着 Internet 技术的不断发展和成熟，Internet 不再仅局限在主干网上，大量现有的通信设施逐步成为 Internet 的运行载体。普通用户和小型网络之间可以通过电话线进行通信，小型网络之间，区域网或大型网络之间则可以利用光纤、通信卫星等实现通信。

20 世纪 80 年代后期以来 Internet 获得了长足的发展。许多大公司发现 Internet 是与遍及全球的员工保持联系、与其他公司合作、与用户进行交互的极好方式。Internet 服务供应商（ISP）开始为个人访问 Internet 提供各种服务，而随着计算机逐渐进入家庭，Internet 的成员数也呈指数增长，人们开始在网络上工作、学习和享受各种服务。

与 Internet 相关的常用术语简单解释如下。

（1）因特网（Internet），专指全球最大的、开放的、由众多网络相互连接而成的计算机网络。它由美国 ARPAnet 发展而来，主要采用 TCP/IP 协议。

（2）万维网（World Wide Web，WWW），也称环球网，是基于超文本的、方便用户在 Internet 上搜索和浏览信息的信息服务系统。

（3）超文本（Hypertext），一种全局性的信息结构，它将文档中的不同部分通过关键字建立链接，使信息得以用交互方式搜索。它是超级文本的简称。

（4）超媒体（Hypermedia），是超文本和多媒体在信息浏览环境下的结合，是超级媒体的简称。

（5）主页（HomePage），通过万维网进行信息查询时的起始信息页。

（6）浏览器（Brower），这里专指 Web 浏览器，如微软的 IE（Internet Explorer），以及可以跨平台的 Netscape Navigator、Opera 等。它可以向万维网服务器发送各种请求，并对服务

器发来的、由 HTML 定义的超文本信息和各种多媒体数据格式进行解释、显示和播放。

（7）目录服务（Directory Service），Internet 上根据用户的某些信息反查找另一些信息的一种公共查询服务。

（8）防火墙（Firewall），用于将 Internet 的子网和 Internet 的其他部分相隔离，以达到网络安全和信息安全效果的软件或硬件设施。

（9）Internet 服务商（Internet Service Provider，ISP），向用户提供 Internet 服务的公司或机构。其中，大公司在许多城市都设有访问站点，小公司则只提供本地或地区性的 Internet 服务。一些 ISP 在提供 Internet 的 TCP/IP 连接的同时，也提供他们自己各具特色的信息资源。

1.2 三类网络程序设计

网络程序设计，或称网络编程，是一个很大的范畴，可以大致上分为 3 类：基于 TCP/IP 协议栈的网络编程、基于 WWW 应用的网络编程、基于.NET 框架的 Web Services 网络编程。本书涉及的内容主要集中在第二类中，同时也会少量涉及第三类。

1.2.1 基于 TCP/IP 协议栈的网络编程

基于 TCP/IP 协议栈的网络编程是最基本的网络编程方式，主要是使用各种编程语言，利用操作系统提供的套接字网络编程接口，直接开发各种网络应用程序。

这种编程方式由于直接利用网络协议栈提供的服务来实现网络应用，所以编程者有较大的自由度，在利用套接字实现网络进程通信以后，可以随心所欲地编写各种网络应用程序。采用这种编程方式用户首先要深入了解 TCP/IP 的相关知识，掌握套接字网络编程接口，更重要的是要深入了解网络应用层协议。例如，要编写电子邮件程序，就必须深入了解 SMTP 和 POP3 协议，有时甚至需要用户自己开发合适的应用层协议。

1.2.2 基于 WWW 应用的网络编程

WWW 应用是 Internet 上最广泛的应用。它用超文本标记语言（Hyper Text Markup Language，HTML）来表达信息，用超链接将全世界的网站连成一个整体，用 Web 浏览器这种统一的形式来浏览，为人们提供了一个图文并茂的多媒体信息世界。WWW 已经深入应用到各行各业。无论是电子商务、电子政务、数字企业、数字校园，还是各种基于 WWW 的信息处理系统，信息发布系统和远程教育系统，都采用了网站的形式。这种巨大的需求催生了各种基于 WWW 应用的网络编程技术，首先出现了一大批所见即所得的网页制作工具，如 Frontpage、Dreamweaver、Flash 和 Firework 等，然后是各种动态服务器页面的制作技术不断发展和完善，如 ASP、JSP、PHP 及本书将要介绍的 ASP.NET 等。这类网络编程也称 Web 编程。

1.2.3 基于.NET 框架的 Web Services 网络编程

21 世纪是网络的世纪，诸如电子商务、电子政务、数字校园和数字企业等 Internet 上的

各种应用层出不穷。巨大的网络编程需求，急需建立一个更高效、更可靠、更安全的软件平台。微软公司的可扩展标记语言（eXtensible Markup Language，XML），Web 服务架构以及.NET 平台就是在这种背景下推出的。IT 业界领先的公司都已认识到它的重要性，表现出极大的兴趣，正在共同努力开发行业标准。

2000 年～2010 年，被比尔·盖茨称为是全新的“数字时代”。数字智能设备将无处不在，并被网络连接起来，新的应用会不断推出，将深刻改变人类的工作和生活方式。微软公司的.NET 技术是“数字时代”的全新技术，它把整个 Internet 当作计算的舞台，为人们提供统一、有序、有结构的 XML Web 服务。

1.3　Web 编程概述

Web 是一种典型的分布式应用框架。Web 应用中的每一次信息交换都要涉及客户端和服务端两个层面。因此，Web 编程技术大体上也可以被分为客户端技术和服务端技术两大类。

1.3.1　Web 的工作原理

Web 的信息源保存在 Web 站点中，用户通过 Web 浏览器来访问。因此，Web 是一种基于客户机/服务器（Client/Server，C/S）的体系结构。用户使用浏览器从网上查阅 Web 信息，把需要的信息从网上下载到本机。由于信息的分布点不同，用户需求信息的不同，表现在 Web 上是链接地址的不断变化。

浏览器的主要功能是解释并显示由 Web 服务器传送来的、由 HTML 写成的文档，包括嵌入在 HTML 文档中的 GIF 和 JPEG 格式的图像。此外，浏览器还可以根据用户的需要配置某些辅助应用程序，用来处理嵌入在 HTML 文档中的声音、视频等外部多媒体信息。通常将 Web 浏览器中显示的 HTML 文本称为 Web 页面（Page）。

Web 服务器是一个软件，用于管理 Web 页面，并使这些页面通过本地网络或 Internet 供客户机浏览器使用。在使用 Internet 的情况下，Web 服务器和浏览器通常位于两台不同的计算机上，也许它们之间相隔很远，甚至不在一个国家。然而，在本地情况下，也许是用一台计算机运行 Web 服务器软件，然后在同一台计算机上通过浏览器浏览它的 Web 页面。访问远程 Web 服务器与本地服务器之间没有什么差别，因为不论处于何种情况，Web 服务器的功能（即生成可用的 Web 页面）保持不变。如果用户是唯一在自己的计算机上访问 Web 服务器的人，那么其操作与用户在自己的计算机上运行 Web 服务器的操作是相同的。无论是何种情况，其工作原理都是不变的。

最常用的 Web 服务器有 Apache、IIS 和 iPlanet 的 Enterprise 服务器等，本书将只介绍微软的 IIS。IIS 是能够运行 ASP.NET 的唯一服务器。安装 Windows 2000 或 Windows XP 时，IIS 服务器是作为安装内容的一部分被安装。IIS 5.0 可以通过安装 Windows 2000 得到，而 IIS 5.1 则可以通过安装 Windows XP 得到，2.2.1 小节将介绍 IIS 的安装。

在 Web 系统中，Web 服务器向浏览器提供服务的工作方式如下。

（1）用户启动客户机（即本机）的浏览器程序，并在浏览器中指定一个统一资源定位器（Uniform Resource Locator，URL），它是浏览器用来访问 Internet 信息的地址，即通常所说的

网址。它准确地描述了信息所在的地址，浏览器可以通过向该 URL 所指向的 Web 服务器发出请求。

（2）Web 服务器接到浏览器的请求后，把 URL 转换成页面所在服务器上的文件路径名。

（3）若 URL 指向的是普通的 HTML 文档，Web 服务器直接送给浏览器，浏览器负责将 HTML 格式文档解释转换成用户能接受的文本格式。HTML 文档中可能包含用 Java、JavaScript、ActiveX 或 VBScript 等编写的小应用程序，服务器也将它们随 HTML 一起传到浏览器，在浏览器所在的计算机上执行。

1. 静态 Web 页面的工作原理

在 Internet 上浏览网页的时候，会发现许多 Web 页面的内容和外观总是保持不变的，并且这些页面的文件名后缀都是.htm 或者.html，这就是静态 Web 页面。下面，编写一个简单的名为“Welcome.htm”的静态页面并对其进行访问，步骤如下。

（1）新建一个文本文件，并输入如下代码。

```
<html>
<!-- 标题-->
<head>
    <title>我的页面标题</title>
</head>
<body>
<!-- 页面主体-->
Welcome! 这是最简单的网页。
</body>
</html>
```

（2）将此文件命名为“Welcome”，并修改扩展名“txt”为“htm”。

（3）将此文件保存到“E:\wwwroot”目录下。将文件保存到“E:\wwwroot”目录下是因为本书例程运行所用的 Web 服务器的主目录设为“E:\wwwroot”。具体的内容将在本书后续章节介绍。

（4）启动 IE，在地址栏中输入地址“http://localhost/Welcome.htm”，并按【Enter】键，运行结果如图 1-1 所示。其中“http://localhost/”代表本机的 Web 服务器。

图 1-1 静态页面

实际上，在用户访问这个页面之前，页面的内容已经确定，不管用户何时访问，以怎样的方式访问，页面的内容都不会再改变。现在对静态页面如何显示在客户机浏览器中进行详细地介绍，具体有以下 5 个步骤，过程如图 1-2 所示。

（1）Web 作者编写由纯 HTML 代码组成的 Web 页面，并将其以.htm 文件格式保存到 Web 服务器上（在 Web 服务器上发布）。

（2）过一段时间后，用户在其浏览器中输入了页面请求（URL），该请求从浏览器传送到 Web 服务器。

（3）Web 服务器确定.htm 页面的位置，并将它转换成 HTML 流。

（4）Web 服务器将 HTML 流通过网络传回到浏览器。

（5）浏览器处理 HTML 并显示该页面。

类似 Welcome.htm 这样静态的、纯 HTML 文件能够呈现出完全可用的 Web 页面，甚至可以给这样的页面加入更多的 HTML 来修改字体和颜色，提高页面的显示效果和可用性。然而，编写纯 HTML 也只能做这些工作，因为页面的内容在用户请求页面之前已经完全确定，没有任何用户交互或动态响应功能。

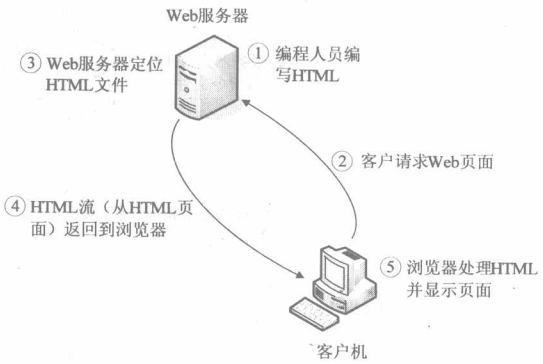


图 1-2　静态 Web 页面的工作原理

2. 动态 Web 页面的工作原理

动态 Web 页面不能在用户请求页面之前通过将硬编码的代码保存到文件这一方法来创建，而是在得到页面请求之后再生成 HTML 文件。主要有两种方法可以实现此功能，下面就介绍这两种方法。

（1）客户端动态 Web 页面。

在客户端模型中，附加到浏览器上的模块（即插件）完成创建动态页面的全部工作。通常包含有一套指令的单独文件随 HTML 代码传送到浏览器，HTML 页面对该文件进行引用。但是，常见的另一种情况是这些指令与 HTML 代码混合在一起。当用户请求 Web 页面时，浏览器利用这些指令生成纯 HTML 页面。也就是说，页面根据用户请求动态生成。这样就生成了一个在浏览器中显示的 HTML 页面。

在客户端模型中，前面介绍生成 Web 页面的 5 个步骤变成了以下 6 个步骤。

① Web 作者编写一套用于创建 HTML 的指令，并将它保存到.htm 文件中。作者也可以用其他语言编写一套指令，这些指令可以包含在.htm 文件中，或放在单独的文件中。

② 过一段时间后，用户在其浏览器中输入了请求 Web 页面，该请求就从浏览器传送到 Web 服务器。

③ Web 服务器确定.htm 页面的位置，也许还需要确定包含指令的其他文件的位置。

④ Web 服务器将新创建的 HTML 流与指令通过网络传回浏览器。

⑤ 位于浏览器的模块会处理指令，并将.htm 页面的指令以 HTML 形式返回（只返回一个页面，即使有两个请求也是如此）。

⑥ 浏览器处理 HTML，并显示该页面。

上述过程如图 1-3 所示。

客户端技术近来已不再受欢迎，因为此技术需要较长的下载时间，特别是当需要下载多个文件时，下载时间就更长。客户端技术的第二个缺点是每一个浏览器以不同的方式解释客户端脚本代码，因此无法保证所有的浏览器都能够理解它们。客户端技术的第三个缺点是当编写使用服务器端资源（如数据库）的客户端代码时会出现问题，因为代码是在客户端解释的，而客户端脚本代码是不安全的，很容易在浏览器中查看源代码。

（2）服务器端动态 Web 页面。

利用服务器端模型，HTML 源代码与混合在其中的一套指令被传回到 Web 服务器。当用户请求页面时，这套指令用于生成 HTML 页面，页面会根据请求动态生成。在服务器端模型中，前面曾介绍的 5 个步骤也变成了 6 个，但由于处理指令的位置不同，这 6 个步骤与客户端模型中的 6 个步骤略有不同。具体步骤如下。

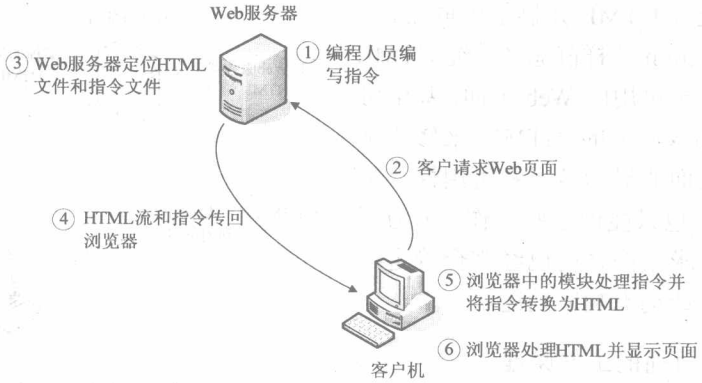


图 1-3　客户端动态 Web 页面工作原理

① Web 作者编写一套创建 HTML 的指令，并将这些指令保存到文件中。

② 一段时间后，用户在其浏览器中输入 Web 页面请求，该请求就从浏览器传递到 Web 服务器。

③ Web 服务器确定指令文件的位置。

④ Web 服务器根据指令创建 HTML 流。

⑤ Web 服务将新创建的 HTML 流通过网络传回浏览器。

⑥ 浏览器处理 HTML，并显示 Web 页面。

上述过程如图 1-4 所示。

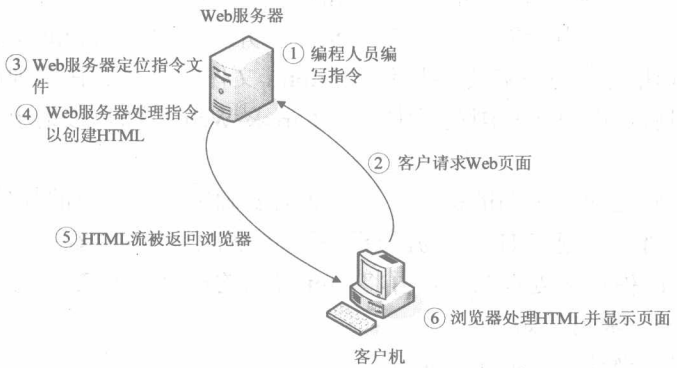


图 1-4　服务器端的动态 Web 页面工作原理

该方法与前面介绍的方法的不同之处是在页面返回到浏览器之前，所有的处理过程都在服务器上完成。与客户端模型相比，此方法的主要优点之一是只有 HTML 代码传回浏览器，这意味着页面的初始代码隐藏在服务器中，而且可以保证大多数浏览器能够显示生成的 HTML 页面。ASP.NET 就属于服务器端模型。

1.3.2　动态 Web 开发技术概述

1. 提供动态内容的客户端技术

每一项提供动态内容的客户端技术都依赖于内置在浏览器中的模块（即插件）来处理指

令。客户端技术是脚本语言、控件以及功能完善的编程语言的综合。

（1）JavaScript

JavaScript 是最早的浏览器脚本语言。JavaScript 语言的前身称为 Livescript，自从 Sun 公司推出著名的 Java 语言之后，Netscape 公司引进了 Sun 公司有关 Java 的程序概念，将原有的 Livescript 重新进行设计，并改名为 JavaScript。JavaScript 是一种基于对象和事件驱动并具有安全性能的脚本语言，JavaScript 可使网页变得更加生动。使用它的目的是与 HTML 超文本标识语言、Java 脚本语言一起实现在一个网页中链接多个对象，与网络客户交互作用，从而可以开发客户端的应用程序。它是通过嵌入或在标准的 HTML 语言中调入实现的。

JavaScript 编写容易，不需要有丰富的编程经验。现在，JavaScript 已经成为制作动态网页必不可少的元素，经常在网页上看到的动态按钮和滚动字幕等，大多数都是使用 JavaScript 技术制作的。

微软公司在 Internet Explorer 3.0 中推出了自己的 JavaScript 版本，其名称为 Jscript，且到目前为止 Internet Explorer 一直在支持它。虽然老版本 Internet Explorer 和 Netscape 浏览器支持的语言有较大的差别，但现在这两个版本之间的差别很小。

（2）VBScript

在 Internet Explorer 3.0 中，微软公司也推出了自己的脚本语言——VBScript。VBScript 是基于 Visual Basic 的编程语言，直接与 JavaScript 竞争。就功能而言，VBScript 与 JavaScript 之间并没有太大的区别，具体使用哪一种语言由个人的爱好决定，特别是 VBScript 具有类似 VB 的简化功能。Visual Basic 开发人员有时会喜欢使用 VBScript，因为 VBScript 的大部分内容是 Microsoft Visual Basic 语言（VB.NET 以前的版本）的子集。不过，对于初级编程者来说，VBScript 更具有吸引力的一点是不区分大小写（与 JavaScript 不一样），而且也不过分注重代码方面的细节。但正是由于这些“优点”，VBScript 运行速度慢、效率也较低。

VBScript 的最大缺点是没有一个微软公司以外的浏览器支持由 VBScript 编写的客户端脚本。虽然曾经有一些用于 Netscape 的插件可提供对 VBScript 支持，但一直没有流行起来，相比之下，JavaScript 有更为广泛的应用与支持。如果希望在客户端编写 Internet 上的 Web 页面，JavaScript 则是唯一可选择的语言。当编写内部网页面，且已知所有客户端都是 Windows 系统上的 IE 时，才考虑使用 VBScript。

JavaScript 和 VBScript 都依赖于称为脚本引擎的模块，该模块被内置到浏览器中，以动态方式处理指令，在这种情况下，这些指令也称为脚本。

（3）Java 小应用程序

Java 是用于开发应用程序的跨平台语言。当 Java 在 20 世纪 90 年代中期首次应用于 Web 时，曾引起了巨大的轰动。Java 代码以小应用程序的形式使用，这些小应用程序基本上是可以借助于<applet>标记方便地插入到 Web 页面的 Java 组件。

Java 比脚本语言的功能更为强大且不牺牲安全性，它在诸如图形功能、文件处理方面提供了更好的支持，Java 也通过 JDBC 提供了强有力的数据库支持。

各种浏览器均通过 Java 虚拟机（JVM）得到内置的 Java 支持，而且有上千个标准的<object>标记和非标准的<applet>标记用于给 Web 页面添加 Java 小应用程序。这些标记告诉浏览器从服务器下载 Java 文件，并利用内置于浏览器的 JVM 执行它。当然，构造 Web 页面的这一额外步骤意味着 Java 小应用程序需要花一些时间进行下载，且下载到浏览器上后，还需要用更长的时间进行处理。虽然在 Web 上非常流行较小的 Java 小应用程序，但较大的小