

ASP.NET动态网站 开发教程(第三版)

- ◆ ASP.NET的概述和开发平台
- ◆ ASP网页框架语言
- ◆ ASP.NET Web页面
- ◆ 常用的内置对象
- ◆ 常用服务器控件
- ◆ 样式、主题和母版页
- ◆ 数据访问和数据库高级操作
- ◆ LINQ技术
- ◆ ASP.NET AJAX控件
- ◆ 企业电子商务网站



韩颖 卫琳 邵玉梅 编著



清华大学出版社

高等学校计算机应用规划教材

ASP.NET 动态网站开发教程

(第三版)

韩颖 卫琳 邵玉梅



清华大学出版社

北 京

内 容 简 介

本书从初学者的角度出发,以通俗易懂的语言、丰富多彩的实例,详细介绍了 ASP.NET 4.0 Web 程序开发技术。全书共分 12 章,主要内容包括 ASP.NET 4.0 概述, Visual Studio 2010 集成开发环境, XHTML 网页设计基础和 C# 4.0 新增功能,使用 ASP.NET 编写网页的基础知识,常用内置对象,相关的服务器控件,数据源和数据绑定控件及 LINQ 技术,以及 ASP.NET 4.0 中的 AJAX 控件和一个综合开发实例。

本书注重基础、讲究实用、内容丰富、结构合理、思路清晰、示例翔实,可作为高等院校计算机及相关专业的教材,还可供工程师和想利用 Visual Studio 2010 开发平台开发 Web 应用程序的人员参考阅读,以及可作为各类培训班的培训教材。

本书的电子教案、实例源代码和习题答案可以到 <http://www.tupwk.com.cn/downpage> 网站下载。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

ASP.NET 动态网站开发教程/韩颖,卫琳,邵玉梅 编著. —3 版. —北京:清华大学出版社,2013. 2
(高等学校计算机应用规划教材)

ISBN 978-7-302-31101-0

I. ①A… II. ①韩… ②卫… ③邵… III. ①主页制作—程序设计—高等学校—教材 IV. ①TP393.092

中国版本图书馆 CIP 数据核字(2012)第 309141 号

责任编辑:胡辰浩 袁建华

封面设计:牛静敏

版式设计:康 博

责任校对:成凤进

责任印制:何 芊

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 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

课 件 下 载: <http://www.tup.com.cn>, 010-62796045

印 刷 者:北京富博印刷有限公司

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185mm×260mm 印 张:23.5 字 数:587 千字

版 次:2004 年 11 月第 1 版 2013 年 2 月第 3 版 印 次:2013 年 2 月第 1 次印刷

印 数:1~5000

定 价:38.00 元

前 言

随着网络技术的飞速发展，人类的信息资源实现了高度共享，从根本上改变了人类进行信息交流的方式，展开了一场史无前例的信息革命。越来越多的人习惯从网上搜索自己需要的资料，越来越多的企业将应用系统发布成网站，以实现快捷、方便的业务处理。浏览器/服务器(B/S)结构的应用程序随着用户的这种需求而被提升到更高的地位。

在实现 B/S 结构的技术中，最具代表性的就是 .NET 框架下的 ASP.NET 技术和 J2EE 框架下的 JSP 技术。如今，随着 ASP.NET 技术的方便性逐渐提高，已经有越来越多的开发人员转入 .NET 开发阵营，从而在这个技术领域内的初学者和急需提高的人员数量不断增加。2010 年，微软公司发布 Visual Studio 2010 正式版本，ASP.NET 升级到 4.0 版本。为了使初学者和 ASP.NET 技术人员快速、完整地了解 ASP.NET 4.0 的技术特性，本书从基础到提高，由浅入深地介绍了相关知识，使读者能够全面、轻松、深刻地了解书中介绍的技术。

目前市面上有不少介绍 ASP.NET 的图书，但是要找一本适合初学者的图书也不容易。有些图书起点太高，初学者难以理解基本概念，学习起来困难重重，容易产生厌倦心理而放弃学习；有的图书又过于简单，读者在学完之后还是不会做任何实际的事情，不能达到一定的高度。而本书恰好解决了这些问题。

概括起来，本书具有以下主要特点：

- 注重基础，讲究实用，力求从入门到精通。
- 充分体现案例教学。本书以易学易用为重点，精选大量实用的示例、知识丰富、步骤详细、学习效率高，特别适合入门者。
- 配有源代码，方便上机实践。本书的所有示例均在 Visual Studio 2010 开发环境下调试通过，读者可以直接下载所有例子的源程序，并通过书中介绍的步骤学习开发要点。

本书共分 12 章，各章的主要内容如下：

第 1 章简要介绍了 HTTP 协议、静态网页和动态网页等 Web 基础知识，以及 ASP.NET 4.0 的发展历史和主要特点，并且讲解了 Visual Studio 2010 的安装方法和开发 ASP.NET 应用程序的一般步骤，最后介绍了 Visual Studio 2010 平台新增功能。通过这些介绍使读者对 ASP.NET 有一个整体的了解，为以后章节的学习打下基础。

第 2 章主要介绍了 ASP.NET 网页框架语言 XHTML 的语法规则以及常用标记，这是进行页面设计的基础，并分析了 HTML、XML 和 XHTML 的不同。

第 3 章主要讲述了 C# 4.0 新增的特性，主要有隐式类型的局部变量、对象和集合初始值设定项、扩展方法、匿名类型、Lambda 表达式、自动实现的属性、dynamic、命名参数和可选参数、协变性和逆变性。

第4章介绍了 ASP.NET 程序结构,如何利用 ASP.NET 建立 Web 页面和创建 ASP.NET Web 页面所需的基础知识,包括 ASP.NET 网页代码模型和生命周期。了解网页代码模型和生命周期能够帮助读者高效地创建 ASP.NET 应用页面,最后,详细地讲述配置文件 web.config 的配置方法。对读者理解 ASP.NET 的工作模式非常重要。

第5章介绍了 ASP.NET 中常用的内置对象,包括 Request、Response、Session、Application 和 Server 的主要方法和属性,并讲解了 Cookie 对象的使用方法。熟练掌握这些内置对象,可以开发出功能强大的应用程序。

第6章介绍了 Web 控件的种类和属性,包括标准控件、验证控件、登录控件、导航控件的使用方法,控件为开发人员提供了高效的应用程序开发方法,开发人员无须具有专业知识就能够实现复杂的应用操作,是开发 ASP.NET 应用程序的基础。

第7章介绍了 CSS 和母版页对 ASP.NET 应用程序进行样式控制的方法和技巧。包括 CSS 的用法、CSS 和 Div 布局的方法、主题的创建和引用以及创建母版页和内容页的方法。

第8章介绍了使用 ADO.NET 进行数据库访问的方法,主要包括 ADO.NET 的数据提供者(Data Provider)和数据集(DataSet)的基础知识等。

第9章介绍了数据绑定技术、ASP.NET 4.0 提供的各种数据源控件和使用数据源控件连接到各种数据源的方法以及复杂数据绑定控件的功能和使用方法。

第10章介绍了 LINQ 的基本知识和如何使用 LINQ 进行数据库操作,包括如何将表生成实体类,了解 DataContext 类,如何使用 LINQ to SQL,并利用 LINQ 技术完成数据的基本查询、添加、删除和修改。最后讲解了一个数据源控件 LinqDataSource 控件。

第11章介绍了 Ajax 的基础知识以及 ASP.NET AJAX 控件,这是微软的客户端异步无刷新页面技术,在 ASP.NET 4.0 以前的版本中,已经包含了此技术框架。

第12章通过一个综合实例将所学知识贯穿在一起。让读者有开发实际项目的体会,从而能够深刻地了解本书前面的知识并达到实战的能力。

本书由韩颖、卫琳、邵玉梅编著,曹仰杰、陶永才参与了部分章节的编写工作,在本书编写过程中,石磊教授给予了大力支持和帮助,同时张宁宁、景京、周梦雪、代琳娜、任国明、张龙涛、贾伟伟、王冬、谢涛、李文洁、何宗真、王晓慧、吴保东、高宇飞、张丹丹、王慧娟等付出了辛勤的劳动,在此一并向他们表示诚挚的感谢。

在编写本书的过程中参考了许多相关文献,在此向这些文献的作者深表感谢。由于时间较紧,书中难免有错误与不足之处,恳请专家和广大读者批评指正。我们的信箱是 huchenhao@263.net, 电话是 010-62796045。

作 者

2012 年 10 月

目 录

第 1 章 ASP.NET 4.0 概述与

开发平台1

1.1 Web 基础知识1

1.1.1 HTTP 协议1

1.1.2 Web 服务器和浏览器2

1.1.3 C/S 模式与 B/S 模式2

1.1.4 Web 的访问原理2

1.2 ASP.NET 简介4

1.2.1 ASP.NET 的历史5

1.2.2 ASP 简介、ASP 与 ASP.NET 的区别6

1.2.3 ASP.NET 的优点7

1.2.4 其他常见的网络程序 设计技术8

1.3 ASP.NET 的开发环境8

1.3.1 安装 Visual Studio 20109

1.3.2 主窗口12

1.3.3 文档窗口12

1.3.4 工具箱13

1.3.5 错误列表窗口14

1.3.6 解决方案资源管理器15

1.3.7 属性窗口16

1.4 ASP.NET 应用程序基础17

1.4.1 创建 ASP.NET 应用程序17

1.4.2 运行 ASP.NET 应用程序18

1.5 Visual Studio 2010 新增功能19

1.6 本章小结22

1.7 练习23

第 2 章 ASP.NET 网页框架语言24

2.1 Web 基本技术24

2.1.1 HTML24

2.1.2 XML25

2.1.3 XHTML25

2.2 XHTML 的基本格式26

2.2.1 ASP.NET 的文档结构26

2.2.2 XHTML 的语法规则28

2.3 XHTML 标记、标记属性30

2.3.1 主体标记<body>... </body>30

2.3.2 注释标记<!--注释内容-->31

2.3.3 分层标记<div>...</div>31

2.3.4 文本和格式标记31

2.3.5 表格标记34

2.3.6 超链接标记<a>...37

2.3.7 图像标记40

2.3.8 表单<form>...</form>42

2.4 本章小结46

2.5 练习47

第 3 章 C# 4.0 新增功能 48

3.1 C#语言简介48

3.2 C#新增功能48

3.2.1 隐式类型的局部变量49

3.2.2 对象和集合初始值设定项52

3.2.3 扩展方法55

3.2.4 匿名类型56

3.2.5 Lambda 表达式59

3.2.6 自动实现的属性63

3.2.7 dynamic 关键字64

3.2.8 命名参数和可选参数65

3.2.9 协变性和逆变性65

3.3 本章小结66

3.4 练习66

第4章 ASP.NET Web 技术简介	68	5.2.4 使用 UserHostAddress 属性	98
4.1 ASP.NET 程序结构	68	5.2.5 使用 Browser 属性	98
4.1.1 ASP.NET 文件类型介绍	68	5.2.6 ServerVariables 属性	100
4.1.2 ASP.NET 文件夹	70	5.2.7 ClientCertificate 属性	101
4.1.3 其他文件夹介绍	72	5.3 Response 对象	101
4.2 页面管理	73	5.3.1 Response 对象简介	101
4.2.1 ASP.NET 页面代码模式	73	5.3.2 利用 Write 方法输出 信息	102
4.2.2 页面的往返与处理机制	74	5.3.3 使用 Redirect 方法引导 客户至另一个 URL 位置	102
4.2.3 页面的生命周期	75	5.3.4 关于 BufferOutput 属性	103
4.2.4 ASP.NET 页生命 周期事件	76	5.3.5 输出缓存资料	104
4.2.5 ASP.NET 页面指令	79	5.3.6 WriteFile 方法	104
4.3 ASP.NET 的网页代码模型	80	5.4 Cookie 对象	105
4.3.1 创建 ASP.NET 网站	80	5.4.1 Cookie 对象简介	105
4.3.2 单文件页模型	81	5.4.2 Cookie 对象的属性和 方法	106
4.3.3 代码隐藏页模型	82	5.4.3 Cookie 对象的使用	106
4.3.4 创建 ASP.NET Web 应用程序	86	5.4.4 检测用户是否启用了 Cookie	108
4.3.5 ASP.NET 网站和 ASP.NET 应用程序的区别	86	5.5 Session 对象	109
4.4 状态管理	87	5.5.1 Session 对象简介	109
4.4.1 页面状态概述	87	5.5.2 Session 对象的使用	110
4.4.2 视图状态	88	5.5.3 Session_Start 和 Session_End 事件	111
4.4.3 控件状态	89	5.5.4 Timeout 属性	111
4.4.4 隐藏域	89	5.5.5 Abandon 方法	111
4.5 ASP.NET 配置管理	90	5.5.6 Session 对象的注意事项	112
4.5.1 web.config 文件介绍	90	5.6 Application 对象	112
4.5.2 配置文件的语法规则	91	5.6.1 Application 对象简介	112
4.6 本章小结	93	5.6.2 利用 Application 对象 存储信息	113
4.7 练习	93	5.6.3 Global.asax 文件	114
第5章 ASP.NET 内置对象	94	5.7 Server 对象	115
5.1 ASP.NET 对象的概况及 属性方法事件	94	5.7.1 Server 对象简介	116
5.2 Request 对象	95	5.7.2 MachineName 属性	116
5.2.1 Request 对象简介	95		
5.2.2 使用 QueryString 属性	96		
5.2.3 使用 Path 属性	98		

5.7.3 ScriptTimeout 属性	117	6.2.10 广告控件(AdRotator)	147
5.7.4 CreateObject 方法	117	6.2.11 表格控件(Table)	149
5.7.5 Execute 方法	117	6.2.12 Literal 控件和 Panel 控件	151
5.7.6 Transfer 方法	117	6.3 验证控件	155
5.7.7 HtmlDecode 方法和 HtmlEncode 方法	118	6.3.1 验证控件及其作用	155
5.7.8 MapPath 方法	119	6.3.2 验证控件的属性和方法	156
5.7.9 URLEncode 方法	119	6.3.3 表单验证控件 (RequiredFieldValidator)	157
5.8 本章小结	121	6.3.4 比较验证控件 (CompareValidator)	158
5.9 练习	121	6.3.5 范围验证控件 (RangeValidator)	159
第 6 章 ASP.NET 常用服务器 控件	123	6.3.6 自定义验证控件 (CustomValidator)	160
6.1 服务器控件概述	123	6.3.7 正则验证控件 (RegularExpression- Validator)	161
6.1.1 控件的种类	124	6.3.8 验证组控件 (ValidationSummary)	163
6.1.2 在页面中添加 HTML 服务器控件	124	6.3.9 禁用数据验证	163
6.1.3 在页面中添加 Web 服务器控件	125	6.4 登录控件	163
6.1.4 以编程方式添加 服务器控件	125	6.4.1 登录控件(Login)	163
6.1.5 设置服务器控件属性	125	6.4.2 登录名称控件 (LoginName)	164
6.2 标准服务器控件	127	6.4.3 登录视图控件 (LoginView)	165
6.2.1 标签控件(Label)	127	6.4.4 登录状态控件 (LoginStatus)	166
6.2.2 TextBox(文本框)控件	128	6.4.5 密码更改控件 (ChangePassword)	167
6.2.3 按钮控件(Button、 LinkButton、 ImageButton)	130	6.4.6 生成用户控件 (CreateUserWizard)	168
6.2.4 HyperLink(超链接)控件	134	6.5 导航控件	169
6.2.5 图像控件(Image)	134	6.5.1 SiteMapPath 导航控件	170
6.2.6 CheckBox(复选框)和 CheckBoxList(复选框列表) 控件	135	6.5.2 Menu 导航控件	172
6.2.7 RadioButton 和 RadioButtonList 控件	138	6.5.3 TreeView 导航控件	176
6.2.8 列表控件(DropDownList 和 ListBox)	140		
6.2.9 MultiView 和 View 控件	143		

6.6	用户控件	178	8.3.4	使用 Command 对象修改数据库的数据	218
6.7	本章小结	182	8.4	使用 DataReader 对象执行数据库命令	220
6.8	练习	182	8.5	使用 DataAdapter 对象和 DataSet 对象	223
第 7 章	样式、主题和母版页	183	8.5.1	DataAdapter 对象简介	223
7.1	CSS 概述	183	8.5.2	DataSet 对象简介	223
7.1.1	CSS 的简介	183	8.5.3	查询数据库的数据	226
7.1.2	CSS 的基础	184	8.5.4	修改数据库的数据	227
7.1.3	创建 CSS 文件	186	8.5.5	增加数据库的数据	229
7.1.4	CSS 常用属性	187	8.5.6	删除数据库的数据	230
7.1.5	DIV 和 CSS 布局	189	8.6	使用 ODBC.NET Data Provider	232
7.2	主题	194	8.6.1	ODBC .NET Data Provider 简介	232
7.2.1	主题的概念	194	8.6.2	连接 ODBC 数据源	232
7.2.2	在主题中定义外观	196	8.7	连接池技术	234
7.2.3	在主题中同时定义外观和样式表	197	8.8	本章小结	238
7.3	母版页	199	8.9	练习	238
7.3.1	母版页和内容页的概念	199	第 9 章	ADO.NET 数据库高级操作	240
7.3.2	创建母版页	200	9.1	数据源控件	240
7.4	本章小结	202	9.1.1	SqlDataSource 控件	241
7.5	练习	202	9.1.2	AccessDataSource 控件	241
第 8 章	ADO.NET 数据访问	204	9.1.3	ObjectDataSource 控件	242
8.1	ADO.NET 概述	204	9.1.4	SiteMapDataSource 控件	243
8.1.1	ADO.NET 简介	204	9.1.5	XmlDataSource 控件	245
8.1.2	与数据有关的命名空间	206	9.2	数据绑定技术	250
8.1.3	ADO.NET 数据提供者	206	9.2.1	简单数据绑定技术	250
8.2	使用 Connection 连接数据库	207	9.2.2	复杂的数据绑定技术	251
8.3	使用 Command 对象执行数据库命令	211	9.2.3	Eval()和 Bind()方法	253
8.3.1	使用 Command 对象查询数据库的数据	212	9.3	数据绑定控件	253
8.3.2	使用 Command 对象增加数据库的数据	214	9.3.1	GridView 控件	254
8.3.3	使用 Command 对象删除数据库的数据	216	9.3.2	Repeater 控件	260
			9.3.3	DataList 控件	263

9.3.4	DetailsView 控件	265	11.1.4	Ajax 使用的技术	318
9.3.5	FormView 控件	271	11.1.5	ASP.NET AJAX	318
9.3.6	DataPager 控件	276	11.1.6	ASP.NET 4.0 AJAX 简单示例	319
9.3.7	ListView 控件	278	11.2	ASP.NET 4.0 AJAX 控件	321
9.4	本章小结	284	11.2.1	ScriptManger (脚本管理员)控件	321
9.5	练习	284	11.2.2	Timer(时间)控件	323
第 10 章	LINQ 技术	286	11.2.3	UpdatePanel (更新区域)控件	324
10.1	LINQ 基本概念	286	11.2.4	UpdateProgress (更新进度)控件	326
10.1.1	LINQ to Objects	286	11.3	本章小结	328
10.1.2	LINQ to ADO.NET	287	11.4	练习	328
10.1.3	LINQ to XML	287	第 12 章	企业电子商务网站	330
10.1.4	LINQ 相关的命名空间	288	12.1	系统设计	330
10.2	LINQ to SQL	288	12.1.1	需求分析	330
10.2.1	IEnumerable 和 IEnumerable<T>接口	289	12.1.2	概念结构设计	330
10.2.2	IQueryProvider 和 IQueryable<T>接口	293	12.1.3	数据库设计	331
10.2.3	DataContext 类	295	12.1.4	功能设计	332
10.2.4	基本查询操作	296	12.2	系统实现	333
10.2.5	基本更改操作	303	12.2.1	设置数据库连接信息	333
10.2.6	LinqDataSource 控件	308	12.2.2	访问数据库公共类	333
10.3	本章小结	314	12.2.3	母版页	337
10.4	练习	314	12.2.4	前台模块功能	338
第 11 章	ASP.NET AJAX	315	12.2.5	后台管理模块	352
11.1	Ajax 简介	315	12.3	本章小结	362
11.1.1	什么是 Ajax	315	12.4	练习	362
11.1.2	Ajax 与传统 Web 技术的 区别	316	参考文献		364
11.1.3	Ajax 的优点	317			

第1章 ASP.NET 4.0概述与开发平台

本章将介绍网站建设的基本原理、流程和创建网站的工具，以及 ASP.NET 的基本概况。作为一种新的 Web 开发技术，ASP.NET 基于 Microsoft 公司的 .NET 框架，支持 C# 和 VB.NET 语言，是主流的网站开发平台之一。通过本章的学习，读者将了解如何安装、使用 ASP.NET 的集成开发环境——Visual Studio 2010(以下简称 VS 2010)，并能够建立简单的动态网站和页面。

本章的学习目标：

- 理解静态网页与动态网页的概念及其工作原理；
- 了解 ASP.NET 的发展历史、特点以及其他常见的网站开发技术；
- 掌握安装 ASP.NET 的集成开发环境 VS 2010 的方法；
- 了解动态网站开发的一般流程并能够创建简单的动态网站；
- 了解 VS 2010 的新增功能。

1.1 Web 基础知识

1.1.1 HTTP 协议

WWW(World Wide Web)又称万维网，起源于 1989 年欧洲粒子物理研究所(CERN)，当时是研究人员为了互相传递文献资料用的。在 WWW 出现之前，Internet 主要用于科学研究和军事方面。自从 WWW 问世以后，Internet 迅速进入千家万户，成为人们学习、工作、交流、娱乐的一个非常重要的手段。

HTTP(Hyper Text Transfer Protocol)，即超文本传输协议，是在 Internet 中进行信息传送的协议，浏览器默认使用该协议。

从浏览器向 Web 服务器发出的访问某个 Web 网页的请求叫做 HTTP 请求。Web 服务器收到 HTTP 请求后，就会按照请求的要求，寻找相应的网页。如果找到，就把网页以 HTML(Hypertext Markup Language，超文本标记语言)代码形式通过 Internet 传回浏览器；如果没有找到，就发送一个错误信息给浏览器。后面的这些操作就叫做 HTTP 响应。

HTTP 协议是一个无状态协议，也就是说，使用该协议时，不同的请求之间不会保存任何信息。每个请求都是独立的，它不知道现在的请求是第一次发出还是第二次或第三次发出，也不知道这个请求的发送来源。当用户请求到所要的网页后，就会断开与 Web 服务器的连接。

从程序设计角度来看,无状态的特点对于 HTTP 来说是一个缺点,因为这使得某些功能很难实现。但是,由于网络本身的特点,这也是没有办法改变的。可以假设一下,如果 HTTP 协议是一个有状态的协议,那么,就需要在 Web 服务器上保存用户的每一个连接,这样可能会导致服务器瘫痪。

1.1.2 Web 服务器和浏览器

Web 服务器就是一台安装了 Web 服务器软件的计算机,它可以为提出 HTTP 请求的浏览器提供 HTTP 响应。常见的 Web 服务器软件有 Apache 和 IIS。Apache 是一个开放源码、采用模块化设计的 Web 服务器软件,具有很强的安全性和稳定性。IIS 是微软公司的产品,最大的特点是图形化的管理界面,使用方便,易于维护。

浏览器是运行在客户机上的程序,用户可以通过它来浏览服务器上的可用资源,因此称为浏览器。当客户进行网页浏览时,由客户的浏览器执行来自服务器的 HTML 代码,并将其内容显示给客户。最初的浏览器是基于文本的,不能显示任何图形信息。1993 年早期,随着 Mosaic 的出现,这一情况发生了改变,Mosaic 是第一个具有图形用户界面的浏览器。目前,最常用的浏览器是 Microsoft Internet Explorer(IE)和 Firefox 浏览器。

1.1.3 C/S 模式与 B/S 模式

C/S 和 B/S 是目前开发模式技术架构的两大主流技术。C/S 模式最早是由美国 Borland 公司研发的,而 B/S 模式则是由美国微软公司研发的。

1. C/S 模式

C/S(Client/Server, 客户机/服务器)模式是一种软件系统体系结构。这种结构是建立在局域网基础之上的,它需要针对不同的操作系统开发不同版本的软件。同时,它不依赖于外网环境,即无论是否能够上网都不会影响应用。

2. B/S 模式

B/S(Browser/Server, 浏览器/服务器)模式是随着 Internet 技术的兴起,对 C/S 模式的一种变化或改进。在这种模式下,用户工作界面是通过 Web 浏览器来实现的。B/S 模式的最大好处是能够实现不同人员、从不同地点、以不同的接入方式访问和操作共同的数据,这就大大减轻了系统维护与升级的成本和工作量,降低了用户的总体成本;其最大的缺点是对外网依赖性太强。

1.1.4 Web 的访问原理

Web 应用程序是基于 B/S 结构的。下面首先介绍客户端和服务器端的概念,然后详述静态网页和动态网页的工作原理。

1. 客户端和服务器端

一般来说,提供服务的一方称为服务器端,而接受服务的一方称为客户端。例如,当用户浏览搜狐主页的时候,搜狐网站所在的服务器就称为服务器端,而用户自己的计算机就称

为客户端，如图 1-1 所示。

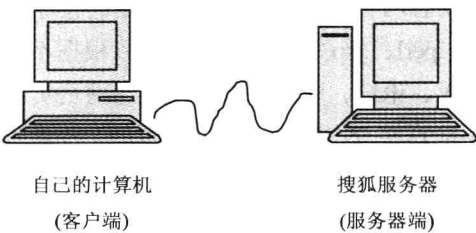


图 1-1 服务器端和客户端示例图

如果在自己的计算机上安装了 Web 服务器软件，其他浏览者通过网络就可以访问该计算机，那么它就是服务器端。很多初学者在调试程序时，往往把自己的计算机既作为服务器端，又作为客户端。

2. 静态网页的工作原理

静态网页也称为普通网页，是相对动态网页而言的。静态并不是指网页中的元素都是静止不动的，而是指网页文件中没有程序代码，只有 HTML(超文本标记语言)标记，一般后缀为 .htm、.html、.shtml 或 .xml 等。在静态网页中，可以包括 GIF 动画，鼠标经过 Flash 按钮时，按钮可能会发生变化。静态网页一经制成，内容就不会再变化，不管何人何时访问，显示的内容都是一样的。如果要修改网页的内容，就必须修改其源代码，然后重新上传到服务器上。

对于静态网页，用户可以直接双击打开，看到的效果与访问服务器是相同的。这是因为在用户访问该网页之前，网页的内容就已经确定，无论用户何时访问，以怎样的方式访问，网页的内容都不会再改变。静态网页的工作流程可以分为以下 4 个步骤：

- (1) 编写一个静态网页文件，并在 Web 服务器上发布。
- (2) 用户在浏览器的地址栏中输入该静态网页的 URL(统一资源定位符)并按回车键，浏览器发送访问请求到 Web 服务器。
- (3) Web 服务器找到此静态网页文件的位置，并将它转换为 HTML 流传送到用户的浏览器。
- (4) 浏览器收到 HTML 流，显示此网页的内容。

在步骤(2)~(4)中，静态网页的内容不会发生任何变化，其原理如图 1-2 所示。

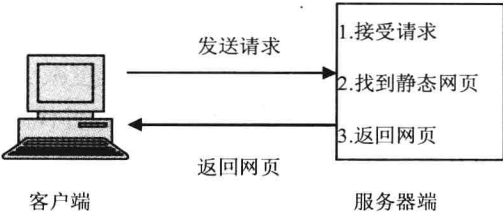


图 1-2 静态网页的工作原理

3. 动态网页的工作原理

动态网页是指在网页文件中除了 HTML 标记以外，还包括一些实现特定功能的程序代

码, 这些程序代码使得浏览器与服务器之间可以进行交互, 即服务器端可以根据客户端的不同请求动态产生网页内容。动态网页的后缀通常根据所用的程序设计语言的不同而不同, 一般为 .asp、.aspx、.cgi、.php、.perl、.jsp 等。动态网页可以根据不同的时间、不同的浏览者显示不同的信息。常见的留言板、论坛、聊天室都是用动态网页实现的。

动态网页相对复杂, 不能直接双击打开, 动态网页的工作流程分为以下 4 个步骤:

(1) 编写一个动态网页文件, 其中包括程序代码, 并在 Web 服务器上发布。

(2) 用户在浏览器的地址栏中输入该动态网页的 URL 并按 Enter 键, 浏览器发送访问请求到 Web 服务器。

(3) Web 服务器找到此动态网页的位置, 并根据其中的程序代码动态创建 HTML 流传送到用户的浏览器。

(4) 浏览器收到 HTML 流, 显示此网页的内容。

从整个工作流程中可以看出, 用户浏览动态网页时, 需要在服务器上动态执行该网页文件, 将含有程序代码的动态网页转化为标准的静态网页, 最后把静态网页发送给用户, 其原理如图 1-3 所示。

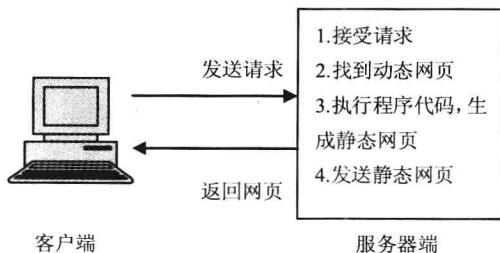


图 1-3 动态网页的工作原理

1.2 ASP.NET 简介

ASP.NET 是 Microsoft 的 Active Server Pages 的新版本, 是建立在微软新一代 .NET 平台架构上、建立在公共语言运行库上, 在服务器后端为用户提供建立强大的企业级 Web 应用服务的编程框架。ASP.NET 为开发能够面向任何浏览器或设备的更安全的、更强的可升级性、更稳定的应用程序提供了新的编程模型和基础结构。使用 ASP.NET 提供的内置服务器控件或者第三方控件, 可以创建既复杂又灵活的用户界面, 大幅度减少了生成动态网页所需的代码, 同时, ASP.NET 能够在服务器上动态编译和执行这些控件代码。

微软在发布 ASP.NET 1.0 时, 根本没有期望这项技术能被广泛采用。但随着该技术的发展和完善, ASP.NET 很快变成了用微软技术开发 Web 应用的标准, 沉重打击了其他 Web 开发平台的竞争者。后来, ASP.NET 有了一个修正版(ASP.NET 1.1)和之后逐步升级的版本(ASP.NET 2.0、ASP.NET 3.5、ASP.NET 4.0)。目前, ASP.NET 作为 Windows 平台上流行的网站开发工具, 能够提供各种方便的 Web 开发模型, 利用这些模型用户可以快速地开发出动态网站所需的各种复杂功能。

1.2.1 ASP.NET 的历史

早期的 Web 程序开发是一件非常繁琐的事, 一个简单的动态页面就需要编写大量的代码(一般用 C 语言)才能完成。

1996 年, Microsoft 推出了 ASP(Active Server Page)1.0 版。它允许使用 VBScript/JavaScript 这些简单的脚本语言编写代码, 并允许将代码直接嵌入 HTML 中, 从而使得设计动态 Web 页面变得简单。在进行程序设计时, ASP 能够通过内置的组件, 实现了强大的功能(如 Cookie)。ASP 最显著的贡献就是推出了 ActiveX Data Objects(ADO), 它使得程序对数据库的操作变得十分简单。

1998 年, 微软发布了 ASP 2.0 和 IIS 4.0。与前一版本相比, 2.0 版最大的改进是外部的组件需要初始化。用户能够利用 ASP 2.0 和 IIS 4.0 建立各种 ASP 应用, 而且每个组件都有了自己单独的内存空间, 可以进行事务处理。

随后, 微软开发了 Windows 2000 操作系统, 其 Server 版系统提供了 IIS 5.0 和 ASP 3.0。此次升级, 最主要的改变就是把很多事情交给 COM+来做, 效率比以前的版本有很大提高, 而且更稳定。

ASP.NET 是微软公司于 2002 年推出的新一代体系结构——Microsoft .NET 的一部分, 用于在服务器端构建功能强大的 Web 应用, 包括 Web 窗体(Web Form)和 Web 服务(Web Services)两部分。随着 .NET 技术的出现, ASP.NET 1.0 也应运而生。ASP.NET 1.0 在结构上与前面的 ASP 截然不同, 几乎完全是基于组件和模块化的。ASP.NET 1.0 允许开发者以一种非常灵活的方式创建 Web 应用程序, 并把常用的代码封装到面向对象的组件中, 这些组件可以由客户端用户通过事件来触发。同时, ASP.NET 提出了代码隐藏类(CodeBehind)的概念, 把逻辑代码(.aspx.cs)和表现页面(.aspx)分离开来, 使用户可以使用后台代码来控制页面的逻辑功能。

2003 年, Microsoft 公司发布了 Visual Studio.NET 2003(简称 VS 2003), 提供了在 Windows 操作系统下开发各类基于 .NET 框架的全新应用程序的开发平台。

2005 年, .NET 框架从 1.0 升级到 2.0 版, Microsoft 公司发布了 Visual Studio.NET 2005(简称 VS 2005)。相应的 ASP.NET 1.0 也升级为 ASP.NET 2.0, 新版本修正了以前版本中的一些 Bug 并在移动应用程序开发、代码安全以及对 Oracle 数据库和 ODBC 的支持等方面都做了很多改进。

2008 年, Visual Studio.NET 2008(简称 VS 2008)问世了, ASP.NET 相应的从 2.0 版升级到 3.5 版。ASP.NET 3.5 版最重要的新功能在于: 支持 AJAX 的网站, 改进了对语言集成查询(LINQ)的支持。这些改进提供了新的服务器控件和新的面向对象的客户端类型库等功能。

2010 年, 微软公司发布 Visual Studio 2010 正式版本, 微软大中华区开发工具及平台事业部总经理谢恩伟总结了 Visual Studio 2010 的五大新特性和功能如下:

- (1) 云计算架构;
- (2) Agile/Scrum 开发方法;
- (3) 搭配 Windows 7 与 Silverlight 4;
- (4) 发挥多核并行运算威力;

(5) 更好支持 C++。

1.2.2 ASP 简介、ASP 与 ASP.NET 的区别

ASP(Active Server Pages)是 Microsoft 公司于 1996 年 11 月推出的 Web 应用程序开发技术,它既不是一种程序语言,也不是一种开发工具,而是一种技术框架,无需使用微软的产品就能编写代码,能产生和执行动态、交互式、高效率的服务器端应用程序。运用 ASP 可以将 VBscript、javascript 等脚本语言嵌入到 HTML 中,可以快速完成网站的应用程序,无需编译即可在服务器端直接执行。容易编写,使用普通的文本编辑器如记事本就可以完成。由于脚本是在服务器端而不是客户端运行,所以用户端的浏览器不需要提供任何别的支持,这样大大提高了用户与服务器之间的交互速度。此外,它可以通过内置的组件实现更强大的功能,如使用 ADO 可以轻松地访问数据库。

ASP 使用 VBS/JS 这样的脚本语言混合 HTML 来编程,而这些脚本语言属于弱类型、面向结构的编程语言,而非面向对象的,这就明显产生了以下几个问题。

- 代码逻辑混乱,难于管理:由于 ASP 是脚本语言混合 HTML 编程,所以很难看清代码的逻辑关系,并且随着程序的复杂性增加,使得代码的管理变得十分困难,甚至超出一个程序员所能达到的管理能力,从而造成这样或那样的问题。
- 代码的可重用性差:由于是面向结构的编程方式,并且混合 HTML,所以,页面原型修改一点,可能整个程序都需要修改,更别提代码重用了。
- 弱数据类型造成潜在的出错可能:尽管弱数据类型的编程语言使用起来会方便一些,但相对于它所造成的出错几率是远远得不偿失的。

以上是语言方面的弱点,在功能方面 ASP 同样存在问题:第一是功能太弱,一些底层操作只能通过组件来完成,在这点上远远比不上 PHP 和 JSP;其次是缺乏完善的纠错/调试功能,这方面 ASP、PHP、JSP 差不多。

那么,ASP.NET 有哪些改进呢?ASP.NET 摆脱了 ASP 使用脚本语言来编程的缺点,理论上可以使用任何编程语言,包括 C++、VB、JS 等,当然,最合适的编程语言还是 Microsoft 为 .NET Framework 专门推出的 C#。首先,C#是面向对象的编程语言,而不是一种脚本,所以它具有面向对象编程语言的一切特性,比如封装性、继承性、多态性等,这就解决了 ASP 的那些弱点。封装性使得代码逻辑清晰,易于管理,并且应用到 ASP.NET 上就可以使业务逻辑和 HTML 页面分离,这样,无论页面原型如何改变,业务逻辑代码都不必做任何改动;继承性和多态性使得代码的可重用性大大提高,可以通过继承已有的对象最大限度地保护以前的开发,并且,C#和 C++、Java 一样提供了完善的调试/纠错体系。

ASP 与 ASP.NET 的区别如下:

- 开发语言不同

ASP.NET 和 ASP 的最大区别在于编程思维的转换,而不仅仅是功能的增强。ASP 仅仅局限于使用 non-type 脚本语言来开发,用户给 Web 页中添加 ASP 代码的方法与客户端脚本中添加代码的方法相同,导致代码杂乱。ASP.NET 则允许用户选择并使用功能完善的 strongly-type 编程语言,也允许使用功能巨大的 .NET Framework。

- 运行机制不同

ASP 是解释运行的编程框架，所以执行效率比较低。ASP.NET 是编译性的编程框架，运行的是服务器上编译好的公共语言运行时库代码，可以利用早期绑定，实时编译来提高效率。

- 开发方式

ASP 把界面设计和程序设计混在一起，维护和重用困难。ASP.NET 则把界面设计和程序设计以不同的文件存放分离开，重用性和维护性得到了提高。

1.2.3 ASP.NET 的优点

ASP.NET 是一种建立在通用语言上的程序构架，能被用于一台 Web 服务器来建立强大的 Web 应用程序。ASP.NET 提供了许多比现在的 Web 开发模式强大的优势。

ASP.NET 可完全利用 .NET 框架的强大、安全、高效的平台特性。ASP.NET 是运行在服务器后端的，编译后的普通语言运行时代码，运行时早绑定、即时编译、本地优化、缓存服务、零安装配置、基于运行时代码受管与验证的安全机制等都为 ASP.NET 带来卓越的性能。对 XML、SOAP、WSDL 等 Internet 标准的支持更是为 ASP.NET 在异构网络里提供了强大的扩展性。

- 威力和灵活性

由于 ASP.NET 基于公共语言运行库，因此，Web 应用程序开发人员可以利用整个平台的威力和灵活性。 .NET 框架类库、消息处理和数据访问解决方案都可以从 Web 无缝访问。ASP.NET 与语言无关，所以可以选择最适合应用程序的语言，或跨多种语言分割应用程序。另外，公共语言运行库的交互性保证在迁移到 ASP.NET 时保留基于 COM 的开发中的现有投资。

- 简易性

ASP.NET 使执行常见任务变得容易，从简单的窗体提交和客户端身份验证，到部署和站点配置。例如，使用 ASP.NET 页框架可以生成将应用程序逻辑与表示代码清楚分开的用户界面，和在类似 Visual Basic 的简单窗体处理模型中处理事件。另外，公共语言运行库利用托管代码服务(如自动引用计数和垃圾回收)简化了程序开发。

- 可管理性

ASP.NET 采用基于文本的分层配置系统，简化了应用服务器环境和 Web 应用程序的配置。由于配置信息是以纯文本形式存储的，因此可以在没有本地管理工具帮助的情况下应用新设置。这种“零本地管理”思想也扩展到了 ASP.NET 框架应用程序的部署。只需将必要的文件复制到服务器，即可将 ASP.NET 框架应用程序部署到服务器上。即使是在部署或替换运行的编译代码时也不需要重启服务器。

- 可伸缩性

ASP.NET 在设计时考虑了可缩放性，增加了专门用于在聚集环境和多处理器环境中提高性能的功能。另外，进程受到 ASP.NET 运行库的密切监视和管理，以便当进程行为不正常(泄漏、死锁)时，可以就地创建新进程，以帮助保持应用程序始终可用于处理请求。