

ASP.NET 2.0 (C#)

基础教程

李勇平 编著



清华大学出版社



ASP.NET 2.0 (C#)

基础教程

李勇平 编著



清华大学出版社
北京



内 容 简 介

本书主要介绍如何使用 ASP.NET 进行服务器端网页的编程。ASP.NET 是微软推出的 Web 开发技术, 开发者可以使用 C#、VB.NET、JavaScript 等 .NET 支持的语言进行开发。本书将使用 C# 作为 ASP.NET 开发语言。

本书将介绍 C# 基本语法 (包括变量、数据类型、表达式、运算符等)、面向对象 C# 编程技术 (包括自定义类和对象、对象的封装性、对象的继承性和对象的多态性)、ASP.NET Web 页面技术 (包括服务器控件的使用、验证控件的使用等)、ASP.NET Web 数据访问技术 (ADO.NET 数据集、ADO.NET 数据访问对象等)、ASP.NET 数据控件技术 (ADO.NET 数据绑定技术), 最后本书还将介绍应用程序状态管理技术。

本书既适合作为软件开发人员的自学教材, 也适合作为大中专院校学生的教材。

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 13501256678 13801310933

图书在版编目(CIP)数据

ASP.NET 2.0 (C#) 基础教程 / 李勇平编著. —北京: 清华大学出版社, 2008.1
ISBN 978-7-302-16291-9

I. A… II. 李… III. ①主页制作—程序设计—教材 ②C 语言—程序设计—教材
IV. TP393.092 TP312

中国版本图书馆 CIP 数据核字 (2007) 第 159554 号

责任编辑: 冯志强 夏兆彦

责任校对: 张 剑

责任印制: 何 芊

出版发行: 清华大学出版社 地 址: 北京清华大学学研大厦 A 座

<http://www.tup.com.cn> 邮 编: 100084

c-service@tup.tsinghua.edu.cn

社 总 机: 010-62770175 邮购热线: 010-62786544

投稿咨询: 010-62772015 客户服务: 010-62776969

印 刷 者: 清华大学印刷厂

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: 185×260 印 张: 25.75 字 数: 611 千字

附光盘 1 张

版 次: 2008 年 1 月第 1 版 印 次: 2008 年 1 月第 1 次印刷

印 数: 1~5000

定 价: 39.80 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题, 请与清华大学出版社出版部联系调换。联系电话: (010)62770177 转 3103 产品编号: 026842-01

学习软件开发并不难，关键是要有好的学习方法。那么学习软件开发高效的学习方法有哪些呢？

第一，多写代码。很多读者喜欢深究概念、模式、架构等，而忽略了最重要的代码编写，结果实际工作时发现自己眼高手低。任何系统分析师、软件架构师都是经过大量的代码编写出来的，而不是只看或者只学习理论学出来的。所以一定要牢记：多写代码，别无选择。

第二，多读代码。读代码不是泛泛地看有多少个类，也不是一行一行研究都是怎么写出来的，读代码要有目的地读，读完后要总结，要有收获。现在网络上有很多开源项目，这些项目都是由很多高手编写出来的，他们不仅有很好的设计和架构，设置运行效率也非常高。所以在学习过程中我们要多读代码，有目的地读，读完后要总结，谈感受，讲收获。只有这样，我们才能站在巨人的肩膀上，也只有这样，我们才能成为软件开发的高手。

本书主要介绍如何使用 ASP.NET 2.0 进行 Web 应用系统的编程。开发者可以使用 C#、VB.NET、JavaScript 等 .NET 支持的语言进行开发。本书将使用 C# 作为 ASP.NET 开发语言。

本书主要介绍 C# 基本语法、面向对象 C# 编程技术、ASP.NET Web 页面技术、ASP.NET Web 数据访问技术、ASP.NET 数据控件技术、ASP.NET 安全机制以及 ASP.NET 配置系统。

1. 学习对象

本书适合作为 ASP.NET 网站开发人员、ASP.NET Web 开发人员、.NET 开发人员入门学习，以及大中专院校学生正式教材、辅导教材、培训教材。

2. 本书使用的软件环境

操作系统：Windows 2000/2003 Server

开发工具：Visual Studio.NET 2005

数据库系统：SQL Server 2000/2005

3. 本书学习方法

学习软件开发最重要的就是能够学以致用，同样，学习 ASP.NET Web 开发最好的效果也是能够学以致用，因此在全书的学习过程中，读者应该带着需求来学习，例如学习完全书后，我们如何完成一个中小型电子商务系统的开发，比如网上书店系统、网上手机系统、网上论坛等。读者也可以从网上下载一些经典的案例，边学习边研究这些案例

的开发技巧和关键点。

ASP.NET 各个知识点是分布在各个章节中的, 因此学习好每一章节非常重要, 读者可以按照以下步骤来完成本书每一章节的学习:

(1) 学习书中的理论知识, 理解 ASP.NET 中的各个关键概念。在学习理论知识时, 读者应该打开 Visual Studio.NET 2005, 并且边学习边练习, 动手编写书中提供的各个实例, 理解实例中涉及到的各个关键知识点和关键代码, 并且举一反三。

(2) 完成每一章后的课后作业题, 检验学习效果。

(3) 完成每一章后的相应上机练习, 进一步熟悉上机步骤以及关键代码的编写。

最后, 在学习过程中应该注重阶段性学习效果的检验。本书作为 ASP.NET 的基础教程, 使用 C# 作为 ASP.NET 的开发语言, 全书分成三个主要部分, 分别为:

(1) ASP.NET 基础部分, 基础部分主要讲解 Web 应用程序的概念以及 C# 语言的基础知识。本书在讲解 C# 语法知识时, 直接在 ASP.NET Web 应用程序中进行演示, 这样让读者能够更加直观地理解 C# 和 ASP.NET 的结合以及应用。基础部分列举了很多小程序, 并且大部分程序列出了详细步骤, 读者应该按照书中列出的步骤练习书中的各个小程序。

(2) ASP.NET 页面设计部分, 页面设计部分主要介绍 ASP.NET Web 服务器控件、验证控件以及用户控件的使用方法, 并且介绍如何使用 Web 服务器控件、验证控件设计页面。读者在学习该部分内容时, 重点需要理解 Web 服务器控件、验证控件的工作机制, 并且理解如何使用控件设计页面。

(3) ASP.NET 数据访问部分, 数据访问是开发 ASP.NET 应用程序的核心, 本书介绍了 ADO.NET 中各种数据访问的方法和技巧, 并且介绍了数据绑定控件的使用。读者在学习数据访问部分时, 应该多练习, 并且结合实际开发一些综合性比较强的实例。另外, Visual Studio.NET 2005 提供了强有力的帮助系统, 读者可以查看帮助系统以了解相关知识点的细则。

由于编著时间仓促, 书中疏漏之处在所难免, 恳请广大读者批评指正。

编 者

2007 年 9 月

理 论 部 分

第 1 章 动态网页编程与 ASP.NET

简介	1
简介	1
1.1 静态网页技术	1
1.2 动态网页技术	2
1.2.1 客户端动态网页技术	2
1.2.2 服务器端动态网页技术	3
1.2.3 动态网页技术总结	3
1.2.4 几种流行的服务器端动态 网页技术简介	4
1.3 什么是 ASP.NET	6
1.3.1 ASP.NET 与 ASP 的区别	7
1.3.2 使用 C# 和 ASP.NET	7
1.3.3 ASP、ASP.NET 以及 C# 之间的区别	8
1.3.4 ASP.NET 2.0	8
1.4 ASP.NET 开发环境	11
1.5 什么是 .NET	12
1.5.1 公共语言运行库 CLR 与微 软中间语言 MSIL	13
1.5.2 使用对象	15
1.5.3 .NET 基类	16
1.5.4 类浏览器	17
总结	18
作业	18

第 2 章 ASP.NET 页面剖析

简介	19
2.1 ASP.NET 页面概述	19
2.1.1 将 ASP.NET 文件以 .aspx 扩展名保存	19

2.1.2 在 Web 页面中插入 ASP.NET 代码	20
2.2 ASP.NET 2.0 页面代码模型和 编译机制	26
2.2.1 ASP.NET 2.0 代码模型	26
2.2.2 ASP.NET 编译机制	27
2.3 ASP.NET 页面请求和响应机制	28
2.4 ASP.NET 页面请求和响应内置 对象	31
2.4.1 Request 对象	31
2.4.2 Response 对象	33
2.5 ASP.NET 应用示例	34
2.5.1 绑定到数据库	35
2.5.2 绑定到简单的 XML 文档	37
总结	40
作业	40

第 3 章 使用 ASP.NET Web 服务器
控件构建 Web 窗体

简介	41
3.1 Web 窗体与 ASP.NET Web 服务器控件	41
3.1.1 HTML 窗体与 ASP.NET Web 窗体	42
3.1.2 将 CSS 应用于 ASP.NET Web 服务器控件	43
3.2 使用标准 ASP.NET Web 服务器控件	44
3.2.1 <asp:Label> 控件	44
3.2.2 <asp:DropDownList>	46
3.2.3 <asp:ListBox>	49
3.2.4 <asp:TextBox>	51

3.2.5	<asp:RadioButton>和 <asp:RadioButtonList>	53
3.2.6	<asp:CheckBox>和 <asp:CheckBoxList>	54
3.2.7	<asp:FileUpload>文件 上传控件	56
3.3	母版页	58
3.4	导航控件	64
3.4.1	TreeView 控件	64
3.4.2	Menu 控件	65
3.4.3	SiteMapPath 控件	66
	总结	66
	作业	67

第4章 在ASP.NET对象中存储信息

	简介	68
4.1	ASP.NET 对象和类	68
4.2	.NET 命名空间	69
4.3	在对象中存储信息	70
4.3.1	变量的声明	70
4.3.2	变量的命名规则	73
4.3.3	变量的作用范围	74
4.3.4	常量	78
4.4	变量的数据类型	79
4.4.1	整数	79
4.4.2	浮点数	80
4.4.3	小数	80
4.4.4	char (字符)	80
4.4.5	boolean (布尔型)	81
4.4.6	枚举类型	81
4.4.7	结构类型	82
4.5	变量的常见运算	83
4.5.1	赋值运算	83
4.5.2	算术运算	84
4.5.3	数值比较运算	84
4.5.4	逻辑运算	85
4.5.5	类型转换运算	85
4.6	数组	87

4.6.1	一维数组	87
4.6.2	多维数组	89
4.6.3	交错数组	91
4.7	数据集合	93
4.7.1	ArrayList	93
4.7.2	Hashtable	95
4.7.3	SortedList	98
4.7.4	四种存储数据集合的 方式的比较	99
	总结	99
	作业	99

第5章 实现和使用ASP.NET

	对象的方法	100
	简介	100
5.1	方法中代码执行的顺序	100
5.1.1	选择执行	100
5.1.2	循环执行	103
5.2	在 Page 类中自定义方法	104
5.2.1	模块化	104
5.2.2	定义和使用方法	105
5.2.3	方法参数传递	107
5.2.4	方法返回值	110
5.2.5	方法参数传递方式	114
5.2.6	模块化最佳实践	116
5.3	调用.NET 对象方法调用	116
5.3.1	日期和时间对象	116
5.3.2	字符串对象	118
	总结	124
	作业	124

第6章 ASP.NET 对象的事件

	与 ASP.NET 服务器对象	125
	简介	125
6.1	什么是事件驱动编程	125
6.2	客户端 HTML 事件	126
6.3	ASP.NET Web 服务器控件事件 处理	127

6.3.1	回发事件与非回发事件	127	7.3.2	base 关键字	162
6.3.2	事件处理程序	130	7.3.3	密封类和密封方法	163
6.3.3	将多个事件连接到一个 事件处理程序	131	7.3.4	虚函数	163
6.4	ASP.NET 服务器控件与客户 端脚本	132	7.3.5	继承总结	164
6.5	ASP.NET 的页面生命周期以及 事件处理	135	7.4	抽象类和接口	165
6.5.1	常规页生命周期阶段	136	7.4.1	抽象的含义	165
6.5.2	生命周期事件	136	7.4.2	抽象类	166
6.5.3	IsPostBack 测试	137	7.4.3	接口	168
6.5.4	ASP.NET 的跟踪功能	138	总结		172
6.6	服务器对象	141	作业		173
6.6.1	Execute 方法和 Transfer 方法	142	第 8 章 访问数据库		174
6.6.2	HtmlEncode 方法和 HtmlDecode 方法	144	简介		174
6.6.3	UrlEncode 方法和 UrlDecode 方法	145	8.1	ADO.NET 概述	174
6.6.4	MapPath 方法	147	8.1.1	ADO.NET 命名空间	175
总结		147	8.1.2	ADO.NET 体系结构	176
作业		147	8.2	连接到数据源	177
第 7 章 自定义 ASP.NET 类		148	8.2.1	选择一个数据提供程序 (Data Provider)	177
简介		148	8.2.2	定义数据库连接	179
7.1	ASP.NET 代码部署单元: 程序集	148	8.2.3	使用数据库连接	179
7.2	ASP.NET 类的定义	150	8.2.4	管理数据库连接	180
7.2.1	对象构造和析构	150	8.3	通过数据提供程序向数据库 执行命令	183
7.2.2	对象的特性: 类的成员 变量	153	8.3.1	查询数据	183
7.2.3	类的成员变量访问作 用域	154	8.3.2	执行数据库操作	185
7.2.4	对象的行为: 类的方法、 属性	154	总结		190
7.2.5	索引器	156	作业		190
7.2.6	方法重载	157	第 9 章 数据集		191
7.3	类的继承	159	简介		191
7.3.1	在 C# 中实现继承	160	9.1	使用数据适配器和数据集处理 数据方式	191
			9.2	数据集	192
			9.2.1	DataTable 对象	193
			9.2.2	表间关系 DataRelation	194
			9.3	数据适配器	196
			9.3.1	使用数据适配器和数据 集添加数据	197

9.3.2 使用数据适配器和数据集修改数据	199	11.2.2 使用数据绑定表达式实现数据绑定	236
9.3.3 使用数据适配器和数据集删除数据	201	11.2.3 使用数据源控件实现数据绑定	238
9.4 数据访问技术总结	202	11.3 数据绑定控件	240
9.4.1 数据访问方式总结	202	11.3.1 使用 GridView 控件显示数据	242
9.4.2 数据集总结	203	11.3.2 DetailsView 控件和 FormView 控件	246
总结	204	11.3.3 DataList 控件和 Repeater 控件	252
作业	204	总结	255
第 10 章 数据访问技巧	205	作业	256
简介	205	第 12 章 用户和应用程序状态管理	257
10.1 异常处理技术	205	简介	257
10.1.1 异常处理结构	205	12.1 Web 上状态管理的意义	257
10.1.2 System.Exception 类	208	12.2 ASP.NET 中状态管理技术	258
10.1.3 识别和使用 SQL Server 异常和错误	210	12.3 使用 Cookie	258
10.1.4 使用 SqlException 类	211	12.3.1 Cookie 的工作原理	259
10.2 处理 BLOB 大数据	213	12.3.2 使用 Cookie 实现状态管理	260
10.2.1 访问 BLOB 数据	214	12.3.3 使用 Cookie 的一般规则	262
10.2.2 存储 BLOB 数据	215	12.4 使用 Session 技术	262
10.3 分页技术	216	12.4.1 会话的工作原理	263
10.3.1 将所有页面的页号显示在页面上的分页方法	217	12.4.2 使用 Session 实现状态管理	264
10.3.2 在页面上显示导航链接的分页方法	222	12.5 使用 Application 技术	268
总结	226	12.5.1 应用程序状态的工作原理	268
作业	226	12.5.2 应用程序状态同步	269
第 11 章 Web 数据验证和数据绑定	227	12.6 响应应用程序和会话事件	270
简介	227	12.7 高速缓存	271
11.1 Web 数据验证	227	12.7.1 页输出缓存	271
11.1.1 ASP.NET 数据验证概述	228	12.7.2 页片断缓存	273
11.1.2 数据验证控件	233	12.7.3 数据缓存	276
11.2 Web 数据绑定概述	234	12.8 有关状态管理的建议	277
11.2.1 使用 DataBind 方法实现数据绑定	235		

12.8.1 使用 Cookie 的场合	277
12.8.2 使用会话的场合	278
12.8.3 使用应用程序状态的 场合	279
总结	279
作业	280

上机部分

第1阶段 动态网页编程与 ASP.NET

简介	281
简介	281
练习 1.1 (估计实验时间 30 分钟)	281
在 IIS 中创建虚拟目录来组织 Web 应用程序	281
练习 1.2 (估计实验时间 30 分钟)	283
使用 Visual Studio .NET 创建一个 新的 Web 应用程序	283
练习 1.3 (估计实验时间 30 分钟)	289
上机练习	289

第2阶段 ASP.NET 页面剖析

简介	290
练习 2.1 (估计实验时间 15 分钟)	290
使用流模式实现 ASP.NET 页面	290
练习 2.2 (估计实验时间 15 分钟)	292
使用页面和代码分离模式实现 ASP.NET 页面	292
练习 2.3 (估计实验时间 15 分钟)	294
使用代码隐藏模式实现 ASP.NET 页面	294
练习 2.4 (估计实验时间 15 分钟)	297
使用 Visual Studio.NET 创建 ASP.NET 页面	297
练习 2.5 (估计时间 30 分钟)	299
使用 Request 和 Response 对象	299
练习 2.6 (估计实验时间 30 分钟)	303
上机练习	303

第3阶段 使用 ASP.NET Web 服务器

控件构建 Web 窗体	304
简介	304
练习 3.1 (估计实验时间 30 分钟)	304
使用 TreeView 控件	304
使用 Menu 控件	307
练习 3.2 (估计实验时间 30 分钟)	308
制作母版页	308
由母版页制作普通页	313
练习 3.3 (估计实验时间 30 分钟)	315
上机练习	315

第4阶段 在 ASP.NET 对象中存储

信息	316
简介	316
练习 4.1 (估计实验时间 60 分钟)	316
使用 C# 创建一个小型考试 成绩管理程序	316
练习 4.2 (估计实验时间 30 分钟)	320
上机练习	320

第5阶段 实现和使用 ASP.NET

对象的方法	321
简介	321
练习 5.1 (估计实验时间 60 分钟)	321
使用 Visual Studio .NET 创建一个 注册页面	321
练习 5.2 (估计实验时间 30 分钟)	328
上机练习	328

第6阶段 ASP.NET 对象的事件与

ASP.NET 服务器对象	329
简介	329
练习 6.1 (估计实验时间 60 分钟)	329
使用 Visual Studio .NET 创建 网上调查程序	329
练习 6.2 (估计实验时间 30 分钟)	336
上机练习	336

第 7 阶段 自定义 ASP.NET 类	337
简介.....	337
练习 7.1 (估计实验时间 60 分钟)	337
使用 ASP.NET 服务器控件创建一个	
简单的网上购书 Web 应用程序.....	337
练习 7.2 (估计实验时间 30 分钟)	341
上机练习	341

第 8 阶段 访问数据库	342
简介.....	342
练习 8.1 (估计实验时间 60 分钟)	342
在 ADO.NET 中直接使用 SQL	
语句访问数据	342
练习 8.2 (估计实验时间 30 分钟)	354
上机练习	354

第 9 阶段 数据集	355
简介.....	355
练习 9.1 (估计实验时间 30 分钟)	355
编写类型化数据集类	355
练习 9.2 (估计实验时间 30 分钟)	358
编写数据访问类 (使用数据适配器和	
类型化数据集实现数据访问)	358
练习 9.3 (估计实验时间 30 分钟)	364
上机练习	364

第 10 阶段 数据访问技巧	365
简介.....	365

练习 10.1 (估计实验时间 30 分钟)	365
Web 应用程序异常处理	365
练习 10.2 (估计实验时间 30 分钟)	367
创建账户管理系统	367
练习 10.3 (估计实验时间 30 分钟)	378
上机练习	378

第 11 阶段 Web 数据验证和数据	
绑定	379
简介.....	379
练习 11.1 (估计实验时间 10 分钟)	379
创建 BookShop 数据库	379
练习 11.2 (估计实验时间 50 分钟)	380
使用 ObjectDataSource 实现 Books	
表的基本操作 (GridView	
和 DetailsView)	380
练习 11.3 (估计实验时间 30 分钟)	388
DataList 控件实例	388
练习 11.4	395
上机练习	395

第 12 阶段 用户和应用程序	
状态管理	396
简介.....	396
练习 12.1 (估计实验时间 50 分钟)	396
访问量统计实例	396
练习 12.2 (估计实验时间 40 分钟)	400
上机练习	400

理论部分

第1章 动态网页编程与 ASP.NET 简介

本章内容:

- 区分静态网页技术与动态网页技术
- 了解客户端动态网页技术和服务器端动态网页技术的工作机制
- 了解常见的服务器端动态网页技术
- 掌握 ASP.NET 和 .NET Framework 的基本概念

简介

随着 Internet 技术的发展,基于 Internet 的 Web 应用程序开发已经成为当今软件技术发展最为快速的领域之一, B/S 结构的应用程序已经成为企业应用软件开发的主流。

本章首先简要介绍各种 Web 应用程序开发技术,然后介绍 ASP.NET 以及 .NET Framework 的基本概念。

1.1 静态网页技术

Web 最初发明的时候只是一种简单的标记语言,主要用于链接文档和各种多媒体数据。当用户在 Web 上浏览信息时,看到的基本上都是静态的网页,这些网页以 .html 或者 .htm 文件保存,在用户浏览前,网页制作者用 HTML 语言编写网页的内容。静态网页制作完成并发布后,页面的内容(文本、图像、超链接等)和外观总是保持不变——它并不考虑谁在访问页面、何时访问页面、如何进入页面以及其他因素。

静态网页的工作过程如下(如图 1.1 所示):

(1) 网页开发者编写由纯 HTML 代码组成的网页,将其以 .htm 文件形式保存到 Web 服务器上。

(2) 用户在浏览器中输入一个 HTTP 网页请求(URL),该请求通过网络从浏览器传到 Web 服务器。

(3) Web 服务器在本地文件系统中定位 .htm 文件,将其转化为 HTML 流。

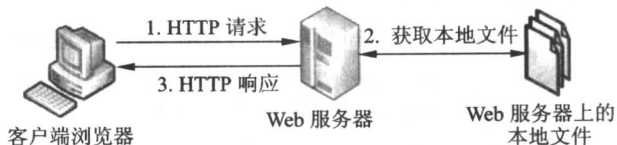


图 1.1 静态 HTML 网页技术的工作过程

(4) Web 服务器将 HTML 流通过网络传到客户端的浏览器。

(5) 浏览器解析 HTML，并显示该网页。

静态的网页可能包含图像、声音等多媒体信息，但是不包含任何客户交互的动态内容，这种简单的静态技术目前仍然在很多 Internet 站点使用，其优点在于信息访问的效率很高，网站的架设与开发相当容易。

1.2 动态网页技术

随着 Internet 的迅速普及，静态的 HTML 内容已经不能满足人们信息查询的需求，开发人员开始给静态的 HTML 网页添加动态内容，以满足不同应用领域的需要。

动态网页技术主要有两个发展方向：客户端动态网页技术与服务器端动态网页技术。

1.2.1 客户端动态网页技术

客户端动态网页技术是指 Web 服务器把原始的 HTML 页面，以及一组包含了页面逻辑的脚本、组件等一起发送到客户端，这些脚本和组件包含了如何与用户交互并产生动态内容的指令，由客户端的浏览器及其附带的插件解析 HTML 页面并执行这些指令。典型的客户端动态网页技术包括 JavaScript、VBScript、ActiveX 控件、Java Applet 等。

客户端动态网页技术的工作过程如下（如图 1.2 所示）：

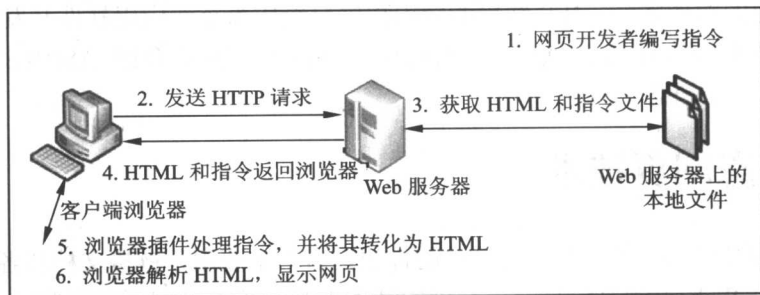


图 1.2 客户端动态网页技术的工作过程

(1) Web 开发者用 HTML 语言编写网页，并将其保存为.htm 或者.html 文件。同时，用其他语言编写指令，这些指令嵌入 HTML 语言中，或者以单独的文件保存。

(2) 用户在客户端浏览器中输入一个 HTTP 网页请求，通过网络传到 Web 服务器上。

(3) Web 服务器在本地文件系统中定位.htm 或者.html 文件，以及 HTML 文件指令中包含的其他文件。

(4) Web 服务器将创建的 HTML 流和其他指令通过网络传递到客户端浏览器中。

(5) 浏览器插件解析指令，并将其返回到.htm 文件中。

(6) 浏览器解析 HTML，显示网页。

客户端动态网页技术的主要优点是充分利用了客户端的计算机资源，减轻了服务器和网络上的计算机压力，同时可以很方便地实现基于图形的用户交互界面。

然而, 因为客户端动态网页技术需要把脚本或者组件 (例如 JavaScript 的 js 脚本文件、ActiveX 控件等) 下载到客户端计算机中, 如果脚本或者组件比较大, 那么下载速度就会很慢。其次, 现存的每种客户端浏览器可能按照不同的方法解析客户端脚本代码, 用户几乎没有办法保证所有的浏览器按照同样的方法解析代码。最后, 组件或者脚本下载带来的一对很难调和的矛盾就是: 对于开发者来说, 如何保证源代码的保密; 对于客户端用户来说, 如何确保下载到自己计算机上的脚本或者组件中的代码不包含恶意攻击的代码。正因为如此, 客户端动态网页技术在 Web 应用程序上的应用一般只局限于显示动画、验证用户输入等方面。

1.2.2 服务器端动态网页技术

服务器端动态网页技术是指在 Web 服务器端, 根据客户端浏览器的不同请求, 动态地生成相应的内容, 然后发送给客户端浏览器。

服务器端动态网页技术的工作过程如下 (如图 1.3 所示):

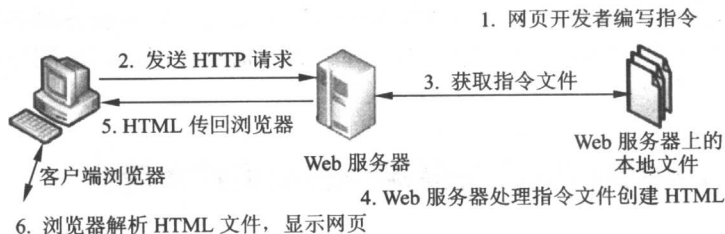


图 1.3 服务器端动态网页技术的工作过程

- (1) Web 开发者编写、创建 HTML 的指令, 将这些指令保存在一个文件中。
- (2) 用户在客户端浏览器中输入一个 HTTP 网页请求, 通过网络传到 Web 服务器上。
- (3) Web 服务器定位指令文件。
- (4) Web 服务器根据指令创建 HTML 流。
- (5) Web 服务器将新创建的 HTML 流通过网络传到客户端浏览器中。
- (6) 浏览器解析 HTML, 显示网页。

使用服务器端动态网页技术, 所有指令在传回浏览器之前在服务器上处理, 用户不再需要考虑指令在哪里处理, 这种技术的一个最关键优点就是只有 HTML 被传回浏览器。这意味着原始页面代码始终被隐藏在服务器中, 开发者不需要去适应各种复杂的客户端计算机环境, 只需要集中在服务器端处理请求就可以了。因此, 它便于采用各种复杂的、结合特定服务器平台的技术, 并形成成熟可靠的标准, 为应用程序开发提供了良好的环境。服务器端动态网页技术的缺点在于对 Web 服务器端的硬件要求较高, 容易产生性能上的瓶颈。

1.2.3 动态网页技术总结

动态网页技术在比静态网页技术的处理过程并没有增加更多复杂性的同时, 增加的

处理步骤（客户端动态网页技术的第（5）步、服务器端动态网页技术的第（4）步）是至关重要的。这里，定义网页的 HTML 直到该网页被请求后才产生。例如，可以使用两种动态网页技术之一编写一个显示当前时间的指令，如下所示：

```
<html>
<head><title>显示当前时间</title></head>
<body>
  <h1>欢迎使用本程序</h1>当前时间是：
  <指令：编写 HTML 显示当前时间>
</body>
</html>
```

在这种情况下，除了不能使用纯 HTML 硬代码显示当前时间外，可以使用纯 HTML 组成大部分的网页内容。同时，可以编写特殊代码（替换上面的黑体代码），在网页被请求时，指示 Web 服务器在使用客户端动态网页技术的第（5）步，或者在使用服务器端动态网页技术的第（4）步，产生 HTML 片段。

服务器端技术安装在 Web 服务器上，所有网页运行在服务器上。而使用客户端技术，网页运行在浏览器中。因此，在服务器脚本传回浏览器前，Web 服务器必须首先将其转换为 HTML。如果没有转换，浏览器将不能理解服务器端代码，那么浏览器中也就不会显示信息。

1.2.4 几种流行的服务器端动态网页技术简介

服务器端网页技术早期有 CGI、ISAPI 等，后期有 JSP、Servlet、ASP、PHP 等，以及微软公司推出的最新服务器端网页技术 ASP.NET。下面简要讲述一下具有代表性的 CGI、ISAPI、ASP 技术，最新的 ASP.NET 技术是本书的主要内容，将在后续章节详细介绍。

1. CGI (Common Gateway Interface)

早期的动态 Web 内容的解决方案之一是通用网关接口 (CGI) 规范，它现在在 UNIX 领域中还十分流行。CGI 本身并不是一种编程语言，而是一种传递客户端请求的标准。Web 服务器接收到客户端浏览器的请求后，将客户端传来的数据按照 CGI 规定的格式进行编码，然后传送给后台的 CGI 应用程序。CGI 应用程序是可以在 Web 服务器上运行的可执行程序，它接收这些数据，然后进行处理，按照 HTML 的格式产生相应的输出。输出被 Web 服务器接收，然后传递给客户端浏览器。

CGI 应用程序可以用很多种语言书写，例如 C、C++、Shell、Perl 等。以下为一个用 C 语言开发的 CGI 控制台程序的源代码：

```
#include "stdafx.h"
#include <stdio.h>
int main(int argc, char* argv[])
{
    printf("HTTP/1.0 200 OK \nContent-Type:text/html \n\n");
```

```
printf("<HTML>\r\n <HEAD>");
printf("<TITLE>测试</TITLE></HEAD>\r\n");
printf("<body>\r\n <h1> 使用 CGI 开发动态 Web 页面</h1>");
printf("<br>\r\n </body>\r\n");
printf("</html>");
return 0;
}
```

上述代码经过编译器编译成为一个可执行的 EXE 文件，然后只需要将 EXE 文件放在虚拟目录下，就可以在浏览器中浏览该 CGI 程序生成的动态网页了。

CGI 应用程序最大的优势在于其简单性，Web 服务器不需要做很多复杂的扩展，只需要启动一个 CGI 外部程序，把参数传递进去，然后负责接收输出就可以了。同时，CGI 程序容易测试、调试。如果程序员具备编写 C/C++/Perl/HTML 的经验，就能创建 CGI 程序。在部署的时候，只需要将可执行文件保存在虚拟目录下，即可在浏览器中测试程序。

然而，由于 CGI 应用程序缺少 Web 服务器的协助，加上 HTTP 协议的无状态的特性，因此状态管理、跟踪等方面的功能较难实现，应用程序的开发也需要很多技巧。同时，CGI 程序在程序运行完毕并且退出之后，可以像其他程序那样修改或者删除 CGI 程序。CGI 的这种特性将导致一些问题的出现。CGI 程序在运行时加载进内存，在完成时将完全从内存中删除。创建和删除进程需要做大量的工作。与阅读 HTML 文件相比，创建和删除进程是一个非常昂贵的操作。Web 服务器如果为每一个请求创建和删除进程，将导致性能问题，也会涉及到资源消耗问题。如果有很多客户端访问同一个 CGI 程序，在内存中就会有该程序的多个实例，这将很快消耗掉 Web 服务器的资源并且导致可伸缩性的问题。当 Web 站点从纯粹的文件共享服务器发展到大型的电子商务或者电子政务机构时，使用 CGI 解决方案将面临大量的问题。

2. ISAPI (Internet Server Application Programming Interface)

为了解决 CGI 带来的性能和可伸缩性问题，微软公司为开发人员提供了另一种动态 Web 页开发方法。通过这种方法可以建立可伸缩性的应用程序，这种方法为 Internet 服务器应用程序编程接口，简称 ISAPI。ISAPI 的功能依赖于 DLL 而不是可执行文件。毫无疑问，使用 DLL 而不使用可执行程序，在性能和可伸缩性上都有一定的优势。

ISAPI 扩展对象在服务器的生命周期内通常仅装载一次，而且 ISAPI 程序通常运行在 IIS 的进程空间中，这样 ISAPI 扩展对象就可以同 IIS 更好地交互，这种运行模式将提高服务器的性能。

ISAPI 相对于 CGI 来说，性能上有很大的提高，但是欲从事 ISAPI 开发的程序员却发现 ISAPI 是一门比较繁杂的技术，开发者不仅需要了解 MFC 的相关知识，而且需要对 HTML 有相当的了解。因此，ISAPI 虽然在性能上有所改进，但是对于开发者来说，ISAPI 是一门很难掌握的技术。

3. ASP (Active Server Pages)

ISAPI 最大的缺点就是编写 ISAPI 程序对开发者的要求很高，许多开发者很难掌握

ISAPI 开发技术。随着 ISAPI 的开发效率和网络的普及与发展之间的矛盾越来越突出, 微软以及许多厂家开发了各种服务器端的脚本技术以用于开发动态 Web 页面, 使得开发效率大大提高。

对于服务器端的脚本, 需要用某种类型的中间应用程序或插件程序来连接。它必须能够接收用户请求, 读取并解释合适的基于服务器的脚本文件, 接着创建输出页, 并传送给 Web 服务器, 在那里作为响应发送给客户端。

在某些情况下, 这个任务划分为两个部分:

- 一个应用程序或插件程序处理与 Web 服务器的往来通信 (一般通过 CGI)。
- 另一个处理解释和执行脚本。

这就是 ASP 中的情况, 脚本引擎的使用与在其他环境下相同。在 ASP 中, 这个脚本引擎就是 ISAPI。通过 ISAPI, ASP 可以将脚本转换为 HTML。

ASP 本身包含了一个 DLL 文件, 名字为 asp.dll, 默认安装在 WINNT\System32\inetsrv 目录下。这个 DLL 文件负责得到一个 ASP 页面 (文件扩展名为 .asp), 然后对它进行分析, 寻找服务器端脚本内容。这个脚本传送给相应的脚本引擎, 脚本的执行结果与 ASP 页中的 HTML 和模板文本结合在一起。完整的页面会送到 Web 服务器, 从那里送往原先提出请求的客户端。

ASP 代码通常用 Microsoft VBScript 编写, 下面的代码是一个用 VBScript 书写的 ASP 文件。可以将以下代码保存为扩展名为 .asp 的文件, 然后将其保存在 IIS 的虚拟目录下, 即可通过 IIS 浏览该 ASP 页面。代码如下:

```
<%@language=vbscript %>
<html><head><title>asp 测试</title></head>
<body>
    <% response.write "<center>asp 测试</center>" %>
</body>
</html>
```

ASP 代码运行的时候是由上而下依次解释执行的。ASP 将一些困难的事情 (使用 CGI 和 ISAPI 创建动态 Web 内容) 简单化了, 使用 ASP 创建动态 Web 页面将非常容易。ASP 开发模式允许开发者编写并且运行代码。

ASP 是 Web 开发者用来创建大型的、可扩展的 Web 应用程序的强大工具。但是由于 ASP 使用了脚本语言, 而脚本语言是解释执行的, 这使得 ASP 创建大型的 Web 应用存在不少问题: 首先, 脚本语言是一种弱类型的语言, 这种语言在处理字符串等其他复杂数据类型的时候, 性能受到一定的限制。其次, ASP 将标准 HTML 和脚本混合, 这种代码编写方式大大限制了开发者实现代码重用和代码维护。

1.3 什么是 ASP.NET

为克服 ASP 的限制, 微软公司开发了一套新的技术, 叫做 ASP.NET。首先, ASP.NET 是一项功能强大的、非常灵活的服务器端技术, 用于创建动态 Web 页面。其次, ASP.NET 是构成 .NET Framework 的一系列技术中的一个。在此可以把 .NET Framework 看成是用