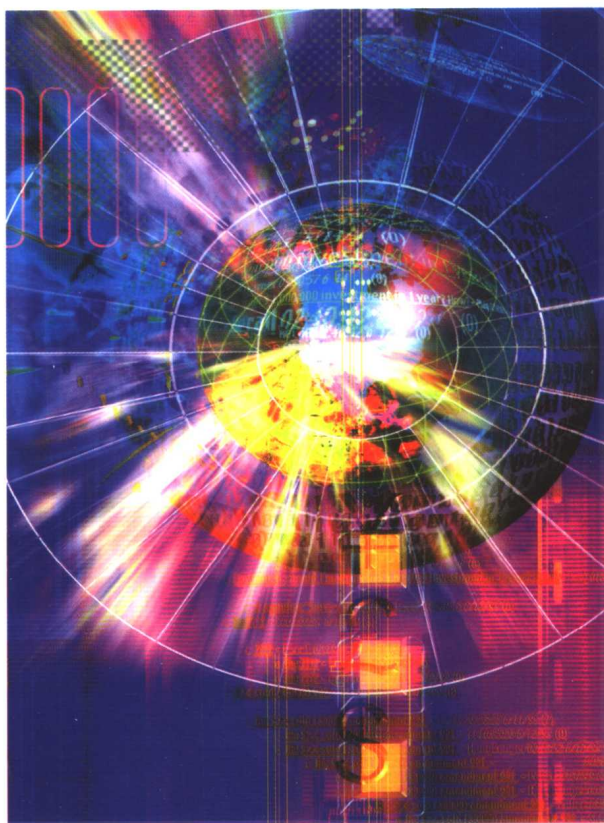


ASP.NET 简明教程

- ◆ 配置 ASP.NET 运行环境
- ◆ VB.NET 语法基础
- ◆ ASP.NET Web 控件
- ◆ ASP.NET 内置对象
- ◆ 事件驱动的程序设计
- ◆ 编写自定义控件
- ◆ 使用 ADO.NET 访问数据库
- ◆ 在 ASP.NET 中应用 XML
- ◆ 性能优化与安全编程



金雪云 编著



清华大学出版社

ASP.NET 简明教程

金雪云 编著

清华大学出版社
北 京

内 容 简 介

本书主要介绍了如何在 Windows 2000 上使用 ASP.NET 创建动态 Web 网页的相关技术, 内容涵盖了什么是 ASP.NET、运行 ASP.NET 程序的环境需求、VB.NET 语法基础、如何使用由 ASP.NET 扩展的 HTML 控件和 Web 内部控件、ASP.NET 的常用内置对象、.NET 的命名空间、如何在 ASP.NET 中实现事件驱动、如何在 ASP.NET 程序中使用 Web 增强控件和自定义控件、ADO.NET 和使用 ADO.NET 进行数据库访问、如何在 ASP.NET 中应用 XML、如何对 ASP.NET 进行配置和优化以及如何在 ASP.NET 中实现安全。最后介绍了一个网站的文章发布系统实例。

本书对于了解 ASP2.0 或者 ASP3.0 并希望在 .NET 框架下开发 Web 应用程序的 Web 程序设计人员来说可以作为参考书, 对于希望从基本概念开始学习的 Web 程序爱好者来说有详细的例子可以边学习边实践, 也可供大专院校师生参考使用。本书所有代码可从 <http://www.tupwk.com.cn> 免费下载。

图书在版编目(CIP)数据

ASP.NET 简明教程/金雪云编著. —北京: 清华大学出版社, 2003

ISBN 7-302-06787-2

I. A... II. 金... 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(2003)第 047343 号

版权所有, 翻印必究。

本书封面贴有清华大学出版社激光防伪标签, 无标签者不得销售。

出 版 者: 清华大学出版社

<http://www.tup.com.cn>

社 总 机: 010-62770175

组稿编辑: 孟毅新

文稿编辑: 孟毅新

封面设计: 王 伟

版式设计: 康 博

印 刷 者: 北京鑫海金澳胶印有限公司

发 行 者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 20.5 字数: 486 千字

版 次: 2003 年 7 月第 1 版 2003 年 7 月第 1 次印刷

书 号: ISBN 7-302-06787-2/TP·5048

印 数: 0001~6000

定 价: 29.00 元

地 址: 北京清华大学学研大厦

邮 编: 100084

客户服务: 010-62776969

前言

微软的 ASP 2.0 和 ASP 3.0 虽然相对而言都是十分年轻的产品,但是很多网站在实现的时候都把它们作为主要的选择。由于 ASP 使得网站的各种代码难于管理,因此面对大量的 HTML 代码和 VBScript、JavaScript 代码混合在一起的程序,程序员往往是一愁莫展,当需要改动程序的时候,他们宁愿写新的代码,也不愿意去改原来的程序,因为原来程序的模块化和可重用性都太低了;另外,由于 ASP 只能使用 VBScript 这样的脚本语言,使得很多功能都不可能轻松地实现;第三,ASP 程序是解释执行的,而 ASP.NET 程序是编译执行的,这导致 ASP 程序在执行效率上要大大低于实现同样功能的 ASP.NET 程序。为了改善网络应用程序的效率,随着微软.NET 的发布,学习如何在.NET 框架中使用 ASP.NET 成为很多 Web 程序设计人员的首选。

ASP.NET 作为 .NET Framework 的一部分,具有 .NET Framework 所拥有的一切优势。首先,在 .NET Framework 中使用高级语言(例如 VB.NET、C#)编写程序的时候,并不是把这些语言所写的代码直接编译成机器代码,而是把程序编译成为中间语言(例如 MS 中间语言)。需要注意的是,中间语言并不是一种可以直接执行的机器代码。与高级语言编写的代码相比,它的可读性很差,但是进行了一系列的优化。为了执行中间语言,就需要一个执行环境 CLR。CLR 在 .NET Framework 中的位置十分重要,可以说是 .NET Framework 的基础。CLR 用 JIT(Just-In-Time)编译器把中间语言代码编译成可以执行的代码,并对程序进行最后的、与机器相匹配的优化,使得程序可以在宿主计算机上尽可能高效率地运行。这些经过优化的二进制代码会保存在缓存中,直到源代码改变为止。

虽然从名字上看起来,ASP.NET 好像是 ASP 的一个新版本,但是从本质上讲,ASP.NET 革命性地改变了 Web 程序设计的设计方式。

1. 本书的内容

在第 1 章中,简要介绍了 HTTP 协议、静态网页和动态网页等 Web 基础知识,并简单介绍了什么是 ASP.NET、ASP.NET 的发展历史、ASP.NET 与 ASP 的比较,以及 ASP.NET 的工作原理;然后讲解了运行 ASP.NET 的配置需求,包括 IIS 和 NGWS;最后,编写了本书的第一个 ASP.NET 程序,这是一个常见的 HelloWorld 程序,让读者对 ASP.NET 程序有一个直观的认识,了解如何加入 ASP.NET 代码到一个 HTML 网页中去。通过这些讲解,读者可以尽快地掌握一些关键的基础知识,对 ASP.NET 有一个概念性的了解,为后面进行 ASP.NET 程序设计打下基础。

在第 2 章中主要介绍了 VB.NET 的语法基础。在进行 ASP.NET 程序设计之前,首先要选择一种 ASP.NET 编程语言。ASP.NET 的默认语言就是 Visual Basic.NET。Visual Basic.NET 是最新的 VB 版本,从功能和性能上讲可以说有了很大的飞跃。首先,早期的 VB 版本是不完全支持面向对象的程序设计的。由于它不支持对象的继承和重载,所

以产生了很多的问题。另外，早期的 VB 版本无法创建多线程的应用程序，这样就不能充分利用机器的资源，也不能很好地提高应用程序的执行效率。但是现在，由于 VB.NET 使用了 CLR，所有的问题都很好地得到了解决。本章介绍了 Visual Basic.NET 的初步知识，包括为什么选择 VB.NET，常量、变量和表达式，如何实现分支和循环，以及过程和函数等基本语法。如果读者是一个精通 Visual Basic.NET 的程序员，那么这一章的内容大致看一下就可以了，但是如果读者对 Visual Basic.NET 一无所知的话，这一部分内容就十分重要。

在第 3 章中讲解了由 ASP.NET 扩展了的 HTML 控件和 Web 内部控件，并着重区分 Web 表单和一般的表单。在这一章中讲述的控件在 ASP.NET 程序设计过程中发挥了重要作用。针对早期的 ASP 版本，ASP.NET 所使用的控件都是在服务器端运行的，并且在实现上比 ASP 要方便得多。

在第 4 章中介绍了 ASP.NET 中常用的内置对象，包括 Response 对象、Request 对象、Application 对象、Session 对象、Server 对象，并讲解了环境变量的使用和 Cookie 的使用。所有的这些对象在 ASP 中都存在，ASP.NET 也保留了这些内置对象。Cookie 是在用户机器的硬盘中保存的小文本，用于存储一些需要保存的数据，这样可以节省服务器端的资源。在这一章中介绍了 Cookie 的基础知识、设置和检索 Cookie、检查用户的浏览器是否禁用了 Cookie。读者通过第 3 章和第 4 章的学习，可以开始进行基本的 ASP.NET 程序设计，并可以根据本书配备的例子来深入了解 ASP.NET 程序设计方法。

在第 5 章中介绍了 ASP.NET 的高级编程方法，重点讲解了命名空间的概念、什么是类库以及 .NET 中常用类库的使用方法；如何使用 Page 类，以及如何使用事件驱动的方式进行程序设计，同时讲解了 ASP.NET 中常用的 Page 事件。

在第 6 章中介绍了其他的 Web 控件。这些控件包括 Web 增强控件和自定义控件。增强的 Web 控件有：日历控件、广告控件以及 6 个验证控件，这些控件是 ASP.NET 新增加的控件，为程序设计提供了极大的方便。另外，使用自定义控件可以获得最大的程序设计自由，并且可以把代码隐藏在 dll 文件中。

在第 7 章中介绍了使用 ADO.NET 进行数据库访问的方法。主要讲解了 ADO.NET 相对于 ADO 的优势、ADO.NET 的使用方法以及在数据库应用中常用的 Web 控件 (DataGrid、DataList、Repeater)。与 ADO 相比，它更容易实现数据共享，提高了标准化程度并使可编程性大大增强。同时，从效率上讲，使用 ADO.NET 将大大提高程序性能。

在第 8 章中介绍了如何在 ASP.NET 中使用 XML。XML 是 Extensible Markup Language(可扩展标记语言)的缩写，它提供了一种独立于应用程序的格式来保存数据，并可以通过这种格式很容易地在不同应用程序之间实现数据共享。现在，.NET 也把 XML 作为 .NET 应用程序传递数据的一种主要的方法。本章主要讲解了什么是 XML、XML 的格式以及如何在 ASP.NET 中使用 XML 存储和管理数据。

在第 9 章中介绍了 ASP.NET 的配置和优化方法。ASP.NET 的所有配置文件都是 XML 格式的文件，因此采用这种方式进行 ASP.NET 配置有方便和灵活的优点。另外，所有的 ASP.NET 配置都是可以随时更改的。也就是说，在一个应用程序的运行期间，可以随时增加和删除 ASP.NET 配置文件中的项目，在修改成功后，可以立即激活使用，

并不会影响服务器的效率。这与有些系统当配置发生变化时需要服务器重新启动才能使新配置生效相比较有非常大的优势。本章主要讲解了两个 config 文件(machine.config 和 web.config)的编写方法以及作用,并讲解了如何使用 global.asax。另外,为了能够让 ASP.NET 应用程序以更快更优的方式运行,除了使用一定的程序设计技巧,还可以采用一些选项来提高应用程序的执行速度。本章重点讲解了使用缓存、跟踪和监视 ASP.NET 进程的方法。

在第 10 章中介绍了 ASP.NET 中如何实现安全性。安全性是对用户的身份进行验证并对通过验证的用户按照对其授予的访问权限来确定此用户是否可以访问某种资源的一个过程。本章重点讲解了实现身份验证和授权的几种方法,尤其是基于 Forms 的身份验证方法具有界面美观、实现方便的特点。除此之外,还简单介绍了如何使用 SSL 进行加密。

在第 11 章介绍了一个网站的文章发布系统作为本书的一个大型综合性实例。栏目管理员可以使用后台管理系统进行网站栏目的管理(增加、修改、删除);文章管理员可以使用后台的管理系统进行栏目中文章的管理(增加、修改、删除、共享其他栏目的文章)。最后在栏目中使用代码实现文章列表和文章内容的显示。为了使这个系统更加实用,还可以确定一个栏目中的文章的显示形式。

2. 本书适用读者

本书适用于希望在 .NET 框架下开发 Web 程序的 Web 程序设计人员,对于希望从基本概念开始学习的 Web 程序爱好者来说也有详细的例子可以边学习边实践,也可供大专院校师生参考使用。

3. 使用本书的要求

运行本书中的所有例子需要建立下面的环境:

- (1) Windows NT/2000/XP 操作系统。
- (2) IIS 5.0 或者 5.0 以上的版本。
- (3) Internet Explorer 6.0 或者更高的版本。
- (4) 本书使用的数据库是 Microsoft SQL Server 2000。
- (5) 在服务器中创建一个文件夹,原样建立例子程序。
- (6) 设置 IIS 中默认 Web 站点的主目录属性页,将它的本地路径设置为刚刚建立的保存了所有例子程序的目录。
- (7) 启动 Web 浏览器,在“地址”栏中输入 `http://localhost/`后面加入要运行的例子程序的路径和文件名。

本书由金雪云编写。在本书的编写过程中,笔者受到来自各方面人士的帮助和支持,在这里表示衷心的感谢。由于时间仓促,本书中难免会有不足之处,敬请各位读者批评指正。

目 录

第 1 章 ASP.NET 概述	1
1.1 Web 基础知识	1
1.1.1 HTTP 协议	1
1.1.2 Web 服务器	2
1.1.3 静态网页	2
1.1.4 动态网页	3
1.2 ASP.NET 简介	4
1.2.1 基本概念	4
1.2.2 ASP 发展历史	7
1.2.3 ASP.NET 与 ASP 的区别	8
1.2.4 ASP.NET 的工作原理	8
1.3 建立 ASP.NET 的运行环境	10
1.3.1 IIS	10
1.3.2 MDAC	13
1.3.3 .NET Framework 和 ASP.NET	13
1.4 开始编写 ASP.NET 程序	14
1.4.1 使用<% %>包含代码块	14
1.4.2 使用<script>标记	15
1.4.3 注释	15
1.5 习题	16
第 2 章 VB.NET 语法基础	17
2.1 编程语言的选择	17
2.1.1 .NET 支持的语言	17
2.1.2 为什么选择 VB.NET	18
2.2 变量、常量和表达式	18
2.2.1 数据类型	18
2.2.2 变量	19
2.2.3 运算符	21
2.2.4 常量	23
2.2.5 数组	24
2.2.6 表达式	25
2.3 分支	25

2.4	循环	27
2.5	过程和函数	30
2.5.1	模块化的优点	30
2.5.2	过程和函数	31
2.5.3	变量的作用域	33
2.6	习题	34
第 3 章	HTML 控件和 Web 控件	35
3.1	使用 ASP.NET 扩展的 HTML 控件	35
3.1.1	Web 表单控件	35
3.1.2	Anchor 控件	38
3.1.3	Button 控件	39
3.1.4	InputButton 控件	41
3.1.5	InputCheckBox 控件	41
3.1.6	InputRadioButton 控件	43
3.1.7	InputImage 控件	44
3.1.8	InputFile 控件	45
3.2	Web 控件	46
3.2.1	用于文本输入和显示的内部控件	46
3.2.2	用于控制传送的内部控件	50
3.2.3	用于选择的内部控件	57
3.3	习题	72
第 4 章	ASP.NET 的常用内置对象	73
4.1	Response 对象	73
4.1.1	输出字符串到网页上	74
4.1.2	重定向	75
4.1.3	缓存 HTML	75
4.1.4	输出文本文件的内容	77
4.2	Request 对象	78
4.2.1	使用 get 方法	79
4.2.2	使用 post 方法	80
4.2.3	使用环境变量	81
4.2.4	获取用户浏览器信息	84
4.3	Application 对象	85
4.3.1	如何使用 Application 对象	85
4.3.2	Contents 集合和 StaticObjects 集合	88
4.3.3	Lock 和 Unlock 方法的使用	89
4.3.4	Application 事件	90

4.4	Session 对象	90
4.4.1	Session 对象的属性	92
4.4.2	Session 对象的方法	93
4.4.3	Session 对象的事件	93
4.4.4	使用 Session 对象的注意事项	93
4.5	Server 对象	95
4.5.1	Server 对象的属性	95
4.5.2	Server 对象的方法	95
4.6	Cookie	97
4.6.1	什么是 Cookie	98
4.6.2	设置 Cookie	100
4.6.3	检索 Cookie	104
4.6.4	检测用户是否启用了 Cookie	106
4.7	习题	106
第 5 章	ASP.NET 高级编程	108
5.1	命名空间	108
5.2	类库	110
5.2.1	类库和基类	110
5.2.2	集合	111
5.2.3	目录和文件的处理	119
5.3	Page 类	126
5.3.1	页面状态	126
5.3.2	页面指令	127
5.4	事件驱动	130
5.4.1	什么是事件驱动	131
5.4.2	ASP.NET 中的事件	131
5.4.3	ASP.NET 中的服务器控件事件	135
5.5	习题	136
第 6 章	ASP.NET 的其他 Web 控件	137
6.1	Web 增强控件	137
6.1.1	日历控件	137
6.1.2	广告控件	143
6.1.3	验证控件	145
6.2	自定义控件	157
6.2.1	代码分离技术	157
6.2.2	编写自定义控件	157
6.2.3	用户控件	161

6.3 习题	162
第 7 章 访问数据库	163
7.1 数据库基础知识	163
7.1.1 关系型数据库基础	163
7.1.2 SQL Server2000	164
7.2 ADO.NET 的优势	165
7.3 ADO.NET 的使用	166
7.3.1 Managed Providers	167
7.3.2 建立数据库连接	167
7.3.3 使用 Command 对象执行数据库命令	169
7.3.4 使用 DataAdapter 对象执行数据库命令	173
7.4 数据绑定	179
7.4.1 什么是数据绑定	179
7.4.2 DataGrid 控件	179
7.4.3 DataList 控件	192
7.4.4 Repeater 控件	197
7.5 习题	199
第 8 章 在 ASP.NET 中应用 XML	200
8.1 什么是 XML	200
8.2 标记、元素以及元素的属性	201
8.3 创建 XML 文档	202
8.4 使用样式表显示 XML	204
8.5 在 ASP.NET 中使用 XML	206
8.5.1 写入 XML 数据	206
8.5.2 读取 XML 数据	209
8.5.3 编辑 XML 数据	210
8.5.4 将 XML 转化为字符串	212
8.6 习题	213
第 9 章 ASP.NET 的配置和优化	214
9.1 ASP.NET 的配置	214
9.1.1 machine.config 文件和 web.config 文件	214
9.1.2 global.asax 文件	222
9.1.3 创建 Application 事件代码	223
9.2 ASP.NET 的优化	227
9.2.1 使用缓存	227
9.2.2 跟踪	232
9.2.3 监视 ASP.NET 进程	234

9.3 习题.....	236
第 10 章 ASP.NET 的安全性	237
10.1 什么是安全性.....	237
10.2 身份验证和授权.....	238
10.2.1 Windows 提供的身份验证和授权	238
10.2.2 IIS 提供的身份验证和授权	240
10.2.3 ASP.NET 提供的身份验证	242
10.3 SSL 加密方法简介	255
10.4 习题.....	255
第 11 章 ASP.NET 应用实例：文章发布系统	256
11.1 文章发布系统简介.....	256
11.2 文章发布系统需求分析.....	262
11.2.1 功能说明	262
11.2.2 功能模块	262
11.3 数据库设计.....	263
11.4 文章发布系统的实现.....	264

第1章 ASP.NET 概述

对于现代人来说, Internet 已经成为生活、学习和工作中不可缺少的一部分。Internet 是跨平台的, 它不需要用户在自己的机器上装载任何其他的软件, 只要有一个浏览器, 就可以浏览到各种各样的信息, 享受各种各样的服务。随着网络接入技术的不断发展, 与 Internet 相连也变得越来越简单, 越来越方便, 以至于很多传统的行业现在也开始大大地依赖于这个神奇的网络。这在 Internet 普及之前是很难想像的。

在这种情况下, 很多单位和个人都开始准备建立自己的网站。不论出于什么目的, 所有人都希望自己制作的网站信息量丰富, 功能尽可能强大。但是如果只使用 HTML, 就只能保证网页的美观, 却不能引入更多更强大的功能。我们看到过很多的小型网站, 由于所有的页面都是静态网页, 所以信息量和更新速度都没有办法得到提高, 时间一长, 用户就失去了兴趣。另外, 由于对每一个页面都需要重新设计, 所以发布新消息也变得很麻烦。既然计算机有强大的计算功能和存储功能, 为什么只用它来存储一些静态网页, 而不很好地使用这些功能呢? 于是, 采用动态网页设计成为现在网站设计的主流。ASP 就是在这种情况下诞生的。

随着时间的推移, 人们又发现, ASP 一方面为网站的设计者带来了简便, 一方面也使得网站的各种代码难于管理。对于程序员来说, 面对的是大量的 HTML 代码和 VBScript、JavaScript 代码混合在一起的程序。当需要改动程序的时候, 他们宁愿写新的代码, 也不愿意去改原来的程序, 因为原来程序的模块化和可重用性都太低。另外, 由于 VBScript 这样的脚本语言的局限, 使得很多功能都不能够轻松地实现, 即使实现了, 也需要写大量的代码。为了解决这些问题, ASP.NET 诞生了。

本节将介绍 ASP.NET。

通过本章的学习, 读者将能够:

- (1) 了解一些 Web 的基础知识。
- (2) 了解 ASP.NET 的发展过程。
- (3) 了解 ASP.NET 的优势。
- (4) 建立 ASP.NET 的运行环境。
- (5) 了解编写 ASP.NET 程序的方法。

1.1 Web 基础知识

1.1.1 HTTP 协议

HTTP 协议即超文本传输协议(Hypertext Transfer Protocol)。这个协议是在 Internet

中进行信息传送的协议、浏览器默认使用这个协议。当用户在浏览器的地址栏中输入 `www.mywebsite.com` 的时候, 浏览器会自动使用 HTTP 协议来搜索 `http://www.mywebsite.com` 网站的首页。

从浏览器向 Web 服务器发出的搜索某个 Web 网页的请求是 HTTP 请求。当 Web 服务器收到这个请求之后, 就会按照请求的要求, 找到相应的网页。如果可以找到这个网页, 那么就把网页的 HTML 代码通过网络传回浏览器; 如果没有找到这个网页, 就发送一个错误信息给发出 HTTP 请求的浏览器。后面的这些操作称为 HTTP 响应。

HTTP 协议是无状态协议。也就是说, 当使用这种协议的时候, 所有的请求都是为搜索某一个特定的 Web 网页而发出的。它不知道现在的请求是第一次发出还是已经多次发出, 也不知道这个请求的发送来源。当用户请求一个 Web 网页的时候, 浏览器会与相关的 Web 服务器相连接, 检索到这个页面之后, 就会把这个连接断开。

从程序设计的角度来看, 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 协议是有状态的协议, 那么就应该让一个连接长时间地存在下去, 这样就可以判断一个用户到底使用了多长时间, 在这段时间内都做了些什么事情。这样在 Internet 环境中, 一个 Web 服务器要保存太多的连接(因为在 Internet 环境中, 用户的数量是很难估计的), 会导致服务器瘫痪。正因如此, 对于所有的 HTTP 请求, Web 服务器都会以同样的方式来对待。

1.1.2 Web 服务器

当提到 Web 服务器的时候, 很多人都会认为这是一台物理的机器。但实际上, Web 服务器是一种软件, 可以管理各种 Web 文件, 并为提出 HTTP 请求的浏览器提供 HTTP 响应。大多数情况下, Web 服务器和浏览器处于不同的机器, 但是它们也可以并存在同一机器上。

比较常见的 Web 服务器有 Apache 和 IIS。由于 ASP.NET 只能在 IIS 上运行, 所以本书把介绍的重点放在 IIS 上。IIS 是微软公司的操作系统 Windows 2000/XP 所提供的。在后面的章节会详细介绍 IIS。

1.1.3 静态网页

在动态网页产生之前, 所有的网页都是静态的。静态网页就是用纯 HTML 代码编写的网页。这些网页的代码是用一些编辑器输入的, 或者是用一些网页设计程序生成的, 保存为 .html 或 .htm 文件的形式。由于这些网页中没有任何与用户相关的部分, 所以在设计完成之后, 无论是哪个用户访问这个网页, 在什么时候访问这个网页, 以何种方式进入这个网页, 它的样子都不会发生任何变化。

下面是一个静态网页的例子。

例 1.1(01-01.htm)一个简单的静态网页, 用于显示一个红色的“hello world.”字符串, 运行结果如图 1-1 所示。

```
1: <html>
2: <body>
3: <font color=red>hello world.<br></font>
4: </body>
5: </html>
```

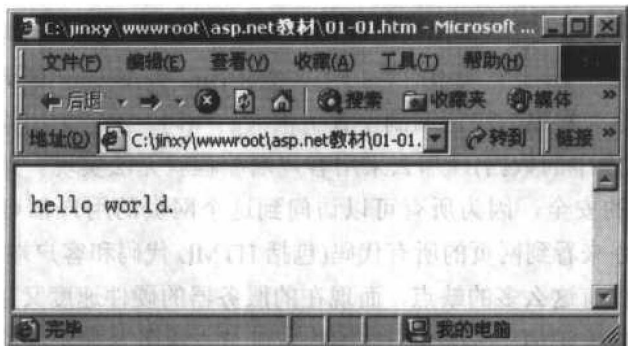


图 1-1 例 1.1 的运行结果

本书约定:

本书中所有的例子都用上面的方法给出。代码前面的数码和冒号并不是代码的内容而是行号,使用它们的目的是为了更清楚地进行代码分析。

上面的例子是一个最简单的 HTML 静态网页,它的目的是显示红色的“Hello World.”字符串。只要这个文件存在,不论什么用户要访问,在什么时候访问,以什么方式进入这个网页,都会显示同样的结果。

采用静态网页会导致很大的局限性。如果希望为用户显示一些个性化的信息,使用静态网页就无法达到目的。例如,如果当前的时间是新年的开始,就可以在网页的最上面显示一个“新年好!”的信息;如果当前的时间是圣诞节,就可以在同样的位置显示一个“Merry Christmas!”的信息。除此之外,静态网页无法防止用户复制 HTML 代码,因为每个用户都可以采用浏览器的“查看源文件”命令来看到网页的 HTML 代码。

1.1.4 动态网页

动态网页可以为不同的用户提供个性化的服务,从前面关于静态网页的介绍中可以看出,使用这种网页的优点是不言而喻的。而为了实现这种动态性,就需要进行程序设计。随着技术的不断发展,在动态网页的实现过程中,一般采用客户端编程和服务器端编程两种程序设计方法。

客户端编程是采用下载到浏览器上的程序来完成所有的有关动态服务的工作。通常的情况是程序员把客户端代码编写到 HTML 文件中,当用户提出对这个网页的请求时,这些客户端代码(即可以实现动态内容的程序)和 HTML 文件的代码一起以响应的方式返回给提出请求的浏览器。由于所有的代码(包括程序和 HTML 标记等)都被浏览器接收,所以这些程序的执行是由浏览器来实现的。常见的客户端编程技术有 JavaScript、VBScript 和 Java applet 等。

在动态网页刚刚出现的时候,多数是使用客户端编程的方法来实现网页的动态服务,因为这样做可以减少服务器的负担,充分利用客户端机器的资源。但是现在客户端编程技术已经越来越不受欢迎。首先,由于所有的代码都要下载到客户端来执行,所以相对而言下载的时间就会增加,尤其当程序的代码量很大的时候,下载时间的延长会十分明显。其次,由于所有的客户端代码都是由浏览器来执行的,所以,在程序编制的过程中,需要针对不同的浏览器进行测试,以保证代码的正确执行。因为现在流行的浏览器很多,一个程序能在 IE 上正确执行,但是在 Netscape Navigator 或者 Opera 上就不一定可以正确执行。这为程序的快速编制设置了很大的障碍。第三,如果需要使用服务器端的资源(例如数据库中的数据),那么采用客户端编程就无法实现。第四,采用客户端编程无法保证代码的安全,因为所有可以访问到这个网页的用户都可以采用浏览器的“查看源文件”命令来看到网页的所有代码(包括 HTML 代码和客户端程序)。

由于客户端编程有这么多的缺点,而现在的服务器的硬件速度又越来越快,相应可以使用的资源也就越来越多,使得客户端编程可以节省服务器端资源的优势已经大大丧失,所以服务器端编程已经渐渐成为动态网页编程的主流。服务器端编程的原理是:程序员编写的代码被保存在服务器上,当用户对某个动态网页提出 HTTP 请求的时候,这个请求所要访问的网页的代码都在服务器端执行完成,并把执行结果以 HTML 的形式传回浏览器。这样,由于浏览器接收到的只是程序执行的结果,所以上面提到的所有的问题都可以迎刃而解。常见的服务器端编程技术有:CGI、PHP、ASP、JSP 和 ASP.NET。

1.2 ASP.NET 简介

1.2.1 基本概念

ASP(Active Server Pages)是一种功能强大而且易于学习的服务器端的脚本编程环境。它是 Microsoft 公司的产品,从 NT Server 操作系统开始就附带这种脚本编程环境。并且,在 NT Workstation、Windows 98 和 Windows 2000 中也都附带这个脚本编程环境。在 2001 年,微软在前面三个版本的 ASP 基础上,推出了全新的 ASP.NET,它开始抛弃前面三个版本都在使用的脚本语言,而是使用 Visual Basic.NET 作为它的默认语言。但是,无论如何变化,下面的优势依然存在:使用这种环境,可以方便地创建动态、快速、交互性强的 Web 站点。

1. 请求和响应

在讲解请求和响应的过程之前,先来了解程序的编译和解释有什么不同。

由于机器能够执行的只是二进制代码,所以所有用助记符来编写的程序都需要使用一个程序把程序语句翻译成机器可以执行的二进制代码。如果这个翻译的过程是在程序执行之前预先进行的,那么就是编译;如果这个翻译的过程是在程序的执行过程中进行的,那么就是解释。由于编译是在程序执行之前进行的,因此可以对代码进行优化来保证编译的结果可以最好地利用机器硬件的各种性能。而解释是在程序的执行过程中进行

的, 所以没有办法对程序进行相关的优化。

早期的 ASP(ASP1.0、ASP2.0、ASP3.0)是 IIS 的一种开放式的无需进行编译的应用程序环境。也就是说, ASP 程序是解释执行的。IIS 是服务器上安装的 Internet 信息服务器(Internet Information Server), 它是 Microsoft 公司开发的一个网络文件和应用程序服务器(即 Web 服务器), 这个服务器包含在操作系统中。在 Windows 2000 中, 它的版本是 5.0。IIS 支持 HTTP、FTP、和 Gopher 协议。由于 ASP 是服务器端的脚本编程环境, 而所有的程序都是解释执行, 这意味着在这个环境中的所有程序在每次被访问的时候都需要 IIS 进行一次解释, 从而客户端会得到一个执行结果。

在 ASP.NET 中, 所有的程序执行都是经过服务器编译的。在这一点上, ASP.NET 与早期的 ASP 版本有很大的不同, 因此在程序执行的效率上也有很大的提高。具体的方法是: 在 ASP.NET 中, 所有的程序仍然是保存在服务器端的, 当一个程序第一次被执行的时候进行编译, 所以当这个程序被再次执行的时候会直接在服务器上执行它的已编译好的可执行二进制代码, 然后把执行结果通过网络返回给客户端。因此, 与 ASP 相比, ASP.NET 程序的执行速度会快很多。

ASP 请求的处理过程如图 1-2 所示。注意, 由于 ASP.NET 对 ASP 请求的处理过程与早期版本的 ASP 的处理有所不同, 所以在服务器端并没有详细表示出请求的处理过程。

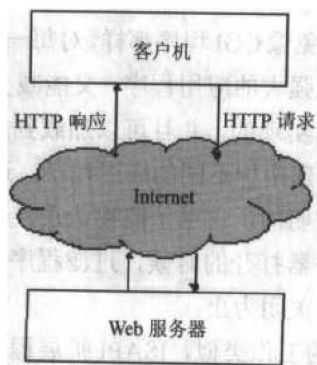


图 1-2 一个 ASP 请求的处理过程

具体的处理过程是这样的: 在客户机中, 有一个用于浏览网页的浏览器, 用户在这个浏览器中输入 HTTP 请求。HTTP 请求通过 Internet 找到相应的 Web 服务器, 并把这个请求传给这个服务器相应的处理模块。由这个处理模块负责找到相应的 ASP 程序或 ASP.NET 程序, 进行相应的执行(对于 ASP 程序来说, 是通过一个名叫 asp.dll 的 ISAPI DLL 进行程序的解释; 而对于 ASP.NET 程序来说, 则是通过 aspnet_isapi.dll 进行处理。执行结果通过 Internet 返回给客户端, 形成 HTTP 响应。

不论是 ASP 文件还是 ASP.NET 文件都是一个可以用任何文本编辑器编辑的纯文本文件, 只要将这个文件的扩展名保存为 asp(ASP 程序)或者 aspx(ASP.NET 程序)就可以了。现在也有很多的开发工具可以进行 ASP.NET 程序的开发, 比如说 Microsoft 的 Visual InterDev 等开发工具。不过也有很多程序员就选择使用记事本来开发 ASP 程序或 ASP.NET 程序。当然, 对于 ASP.NET 来说, 还可以使用最新的 Visual Studio.NET 进行

各种控件的开发。

2. 选择 ASP.NET 的原因

现在有很多流行的技术进行动态网页的设计,下面比较一下常用的几种动态网页设计技术: CGI、ISAPI、IDC、ASP 和 ASP.NET。

(1) CGI

几乎所有的 Web 服务器都支持 CGI。在 UNIX 计算机上,大部分的 CGI 程序都是用 Perl 语言和 C 语言编写的。在 IIS 上,用 CGI 程序可以调用操作系统所提供的所有功能,并使用 ODBC 建立数据库连接来使用数据库功能。

但是,从开发人员的角度来看,这些应用程序在维护和调试的时候都十分困难。并且,从 Web 服务器的性能上来看,使用 CGI 还有一个更大的缺点:对于每一个客户的请求,CGI 都要产生一个进程来进行处理。这样,当服务器的访问量很大的时候,系统内部用于处理用户响应的进程就十分多,这么多的进程会让服务器的资源很快地消耗掉,很容易导致机器的崩溃。

因此可以看出,使用 CGI 的优点是可以创建功能强大的应用程序;缺点是难以维护和调试,系统资源消耗大。

(2) ISAPI

使用 ISAPI 的优点是可以避免像 CGI 程序那样,对每一个客户请求都产生一个进程,所以它成为一种既可以产生功能强大的应用程序,又能减少性能损失的一个程序设计环境。ISAPI 应用程序是通过 DLL 实现的,并且可以加载到服务器的进程空间,以保证程序执行得更快。用 ISAPI 可以创建两种不同的应用程序:过滤程序和扩展程序。

过滤程序对于客户来说是透明的,它用于监视请求、自定义身份验证方案、随机数据转换以及更多的事情。在服务器打开的时候,过滤程序加载到 Web 的进程空间,并一直驻留在内存中,直到服务器关闭为止。

扩展程序与 CGI 应用程序的工作类似,ISAPI 扩展程序用于处理表单、从数据库中检索数据、执行业务逻辑等。与过滤程序一样的是,扩展程序也是加载到 Web 服务器的进程空间,但是只在第一个用户从扩展程序请求服务时才发生,而不是在服务器启动时发生。扩展程序也可以用来获得操作系统级别的访问。

由于 ISAPI 的 DLL 是被加载到 Web 服务器的进程空间的,所以此种方法的最大缺点是,如果编写的 ISAPI 应用程序对性能的考虑不是很好,就会导致服务器的崩溃。它还有另外一个缺点,就是它仍然要使用 C 语言这样的十分复杂的语言来进行编写。这对于程序员来说,调试和维护仍然不容易。

(3) IDC

IDC 是 IIS 从第一个版本就有的功能,用它可以通 ODBC 建立与数据库的连接,进行数据访问,并创建动态的页面把结果显示给用户。这样,使用 IDC 可以很容易地实现数据库的功能,不用创建 IGC 和 ISAPI 程序。

使用 IDC 要建立两个文件:一个查询文件和一个模板文件。查询文件用于保存对数据库访问的 SQL 语句,模板文件用于保存命令文件,它用 HTML 模板合并查询的结果。