

高等院校计算机教育系列教材

ASP.NET 2.0 动态网站开发教程

程不功 龙跃进 卓琳 主编



清华大学出版社

高等院校计算机教育系列教材

ASP.NET 2.0 动态网站开发教程

程不功 龙跃进 卓琳 主 编

清华大学出版社

北 京

内 容 简 介

ASP.NET 2.0 是微软公司推出的新一代企业级开发平台, 它不是 ASP.NET 1.1 版本的简单升级, 而是对原有版本的全面革新。ASP.NET 2.0 不仅功能强大, 而且易学易用。利用它能够在最短的时间内开发出具有高效性、高可靠性和高可扩展性的网站。

全书共 21 章分为四部分。第一部分介绍 ASP.NET 2.0 的基础知识, 包括 .NET 框架的体系结构、ASPX 网页的运行机制、验证控件的使用以及相关的浏览器技术。第二部分介绍了数据访问技术, 包括 ADO.NET 的系统结构, 对数据库的连接、显示、编辑与同步, 以及使用存储过程、数据缓存的方法等。第三部分站在网站的高度介绍了网页界面一致性的设计、网站导航以及网页保护等方面的问题。第四部分介绍了三层架构以及个性化设计等方面的问题。本书的内容以应用开发为主线, 叙述方法的主要特点是通过大量示例来说明问题。除此以外, 书中还提供了几个典型的综合应用, 如网上招聘网页的设计、以订单为中心的电子商务网站的设计等, 只需对它们稍加修改即可应用于实际工作之中。

本书是高等学校计算机专业及电子信息类等相关专业的教材, 亦可作为高等学校非计算机专业的教材或其他学习 ASP.NET 语言人员的自学教材。

版权所有, 翻印必究。举报电话: 010-62782989 13501256678 13801310933

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。

本书防伪标签采用特殊防伪技术, 用户可通过在图案表面涂抹清水, 图案消失, 水干后图案复现; 或将表面膜揭下, 放在白纸上用彩笔涂抹, 图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

ASP.NET 2.0 动态网站开发教程/程不功, 龙跃进, 卓琳主编; —北京: 清华大学出版社, 2006.4
(高等院校计算机教育系列教材)
ISBN 7-302-12718-2

I. A… II. ①程… ②龙… ③卓… III. 主页制作—程序设计—教材 IV. TP393.092

中国版本图书馆 CIP 数据核字(2006)第 022486 号

出 版 者: 清华大学出版社 地 址: 北京清华大学学研大厦
<http://www.tup.com.cn> 邮 编: 100084
社 总 机: 010-62770175 客户服务: 010-62776969

组稿编辑: 邹 杰

文稿编辑: 汤涌涛

排版人员: 房利萍

印 装 者: 三河市春园印刷有限公司

发 行 者: 新华书店总店北京发行所

开 本: 185 × 260 印张: 18.25 字数: 432 千字

版 次: 2006 年 4 月第 1 版 2006 年 4 月第 1 次印刷

书 号: ISBN 7-302-12718-2/TP · 8118

印 数: 1 ~ 4000

定 价: 27.00 元

前 言

近几年来,网络应用的广度和深度以人们预想不到的速度迅猛向前发展,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,网站开发平台的竞争也异常激烈。

进入 21 世纪以来,微软公司(Microsoft)提出了.NET 战略,确定了“创建下一代 Internet 开发平台”的目标。在创建 ASP.NET 2.0 新版本中采用了前所未有的方式:即首先发布试用版本,并在全世界范围征求意见,然后根据反映的意见改进后再次发布试用版本,多次反复以达到尽可能完善的目的。在近两年的时间里微软公司先后发表了 ASP.NET 2.0 的 Alfa、beta 1、beta 2 三个试用版本,从而获得了丰硕的成果。即使是在 2005 年年底正式发布以后还允许全世界的客户免费使用一年。

ASP.NET 2.0 是一个快速开发网站的平台,它是 2005 年 12 月 16 日微软公司在中国正式发布的三大产品之一。作为新一代企业级开发平台,ASP.NET 2.0 具有功能强大,运行可靠且易学、易用的特点。虽然 ASP.NET 2.0 是从 ASP.NET 1.1 版本发展而来,但是新版本已经对原版本做了全面的革新(新版本与老版本不兼容)。在扩大网站功能、增强可靠性、可扩展性以及提高运行效率等方面都取得了突破性的进展。因此新版本的诞生是一个重要的里程碑,它标志着.NET 时代已经到来,网络的应用开发将从此进入一个新的阶段。

本书的主要目的是使读者在尽可能短的时间内,掌握使用 ASP.NET 2.0 的方法,以创建功能强大的动态网站。“应用开发”是全书的主线,通过大量示例讲述方法,是本书的主要特点。除此以外,书中还提供了几个综合性的应用实例,如:“网上招聘”和“留言板”网页的设计以及“以订单为中心的电子商务网站”的设计等。只需要对这些实例稍作修改即可应用于实际工作中。

虽然本书的主要目的是教会读者使用系统,但本书并不是一本“傻瓜书”。因为从本书作者的角度看来,要想真正掌握一个系统,学会相关的知识和理论,了解系统的基本设计思想是完全必要的。只有这样才能正确使用系统,灵活地应用系统,并为进一步创造性地应用系统奠定基础。在这种思想的指导下,本书的内容做了以下安排:

- 第 1~2 章介绍了系统的全貌。
- 第 3~5 章介绍了浏览器端网页的开发技术。ASP.NET 是基于服务器端的技术,在这之前先介绍浏览器端技术,这是基于以下两个理由:一个完整的网站通常是服务器端技术与浏览器技术相结合的产物;了解传统的浏览器技术对掌握 ASP.NET 技术的实质很有帮助。
- 第 6~8 章从代码的存储模式、事件模型、状态管理三个方面介绍 ASPX 网页运行的基本机制。
- 第 9 章是校验控件的应用。
- 第 10~15 章围绕数据访问介绍了 ADO.NET 的系统结构,对数据库的连接、显

示、编辑与同步, 以及使用存储过程、数据缓存等方法。这一部分是全书的重点。

- 第 16~18 章是从网站的整体高度上讲述为实现网页界面一致性的设计、网站导航以及网页保护等方面的问题。
- 第 19 章讲述一个以订单为中心的小型电子商务网站的系统设计, 这是一个很好的综合示例, 有利于提高综合应用多方面知识的能力。
- 第 20 章介绍“创建三层架构”, 这对于开发一个比较复杂的系统或者有特殊要求的网站的设计很有帮助。
- 第 21 章是个性化服务的设计。个性化服务是当前网站发展的一个方向, 也是正在发展中的热门技术。本章介绍了 ASP.NET 2.0 对个性化服务所提供的支持, 并举例说明了它的应用。

在 ASP.NET 2.0 中允许使用 VB .NET、VJ# .NET 或者 C# .NET 等开发语言。本书采用的是 C# .NET, 这是微软创建的最适合于开发 .NET 系统的语言。其实采用其他语言开发时的功能也都相同, 只是语法有些区别。不熟悉 C# .NET 的读者并不会感到有多大困难。必要时可以参阅本书的附录。

本书特别适合于广大从事 IT 的工作人员即初、中级网站开发者以及动态网页的设计者或业余爱好者, 也可以用作大专院校计算机应用专业的教材或参考书。作者曾经利用这些内容给计算机专业的学生授课, 取得了很好的教学效果。

感谢微软提供了如此卓越的产品, 并且在网站上发表了一系列相关技术的文章, 本书就是在参阅了这些文章, 并加上自己的理解、经验和体会才编著出来的。在这里对产品的创建者, 文章的作者一并表示感谢。同时也欢迎读者对本书的缺点和错误提出批评、指正。

编者

2006 年 2 月

目 录

第 1 章 .NET 框架简介	1	3.4 创建移动的文本	32
1.1 网站发展历史的简要回顾	1	3.5 HTML 与 XML 的比较	33
1.1.1 从静态网页发展到动态网页	1	3.6 综合示例	33
1.1.2 动态网页发展的几个阶段	2	3.7 小结	34
1.1.3 ASP 的技术特点	3	3.8 习题	35
1.1.4 新一代 ASP.NET 技术	3	第 4 章 网页布局与插入图像	37
1.2 .NET 框架的体系结构	4	4.1 网页布局	37
1.3 XML: 可扩展标记语言	7	4.1.1 布局的基本概念	37
1.3.1 什么是 XML	7	4.1.2 表格和图层	37
1.3.2 使用 XML 的特点	10	4.2 插入图像	41
1.4 小结	11	4.2.1 图像的类型	41
1.5 习题	11	4.2.2 插入图像的方法	41
第 2 章 ASP.NET 2.0 的体系结构	13	4.3 编辑图像	42
2.1 ASP.NET 的逻辑结构	13	4.3.1 用 HTML 语言编辑图像	42
2.2 ASP.NET 应用程序的组成	14	4.3.2 用可视化方法编辑图像	43
2.2.1 虚拟目录	14	4.4 小结	44
2.2.2 网页文件	14	4.5 习题	44
2.2.3 网站配置文件(Web.config)	15	第 5 章 动态 HTML 技术	46
2.2.4 网站全局文件(Global.asax)	16	5.1 动态 HTML 的基本理论	46
2.2.5 两个共享的目录	16	5.1.1 DHTML 基本概念	46
2.3 创建新网站	16	5.1.2 DOM: 文档对象模型	47
2.4 小结	19	5.2 JavaScript 语言	51
2.5 习题	19	5.2.1 JavaScript 语言简介	51
第 3 章 HTML 语言与文本编辑	21	5.2.2 JavaScript 的基本用法	52
3.1 HTML 语言	21	5.3 级联样式表(CSS)的使用	61
3.1.1 HTML 语言概述	21	5.3.1 CSS 的基本概念	61
3.1.2 HTML 标记的基础	22	5.3.2 CSS 的定义与使用	62
3.1.3 HTML 文档的基本结构	23	5.4 多媒体的引用	64
3.1.4 HTML 文档的编辑工具	24	5.5 DHTML 的应用示例	64
3.2 HTML 文本编辑	25	5.6 小结	68
3.2.1 HTML 文本的格式化	25	5.7 习题	69
3.2.2 设置列表文本	28	第 6 章 ASPX 网页及代码存储模式	70
3.3 超链接	29	6.1 ASPX 网页的基类	70

6.2 ASPX 网页代码的存储模式.....	70	9.3 各验证控件的使用方法.....	111
6.2.1 代码分离模式.....	71	9.3.1 RequiredFieldValidator 控件.....	111
6.2.2 代码的单文件模式.....	74	9.3.2 CompareValidator 控件.....	112
6.3 代码模式的选择.....	75	9.3.3 RangeValidator 控件.....	112
6.4 小结.....	76	9.3.4 RegularExpressionValidator 控件.....	114
6.5 习题.....	76	9.3.5 ValidationSummary 控件.....	114
第 7 章 网页标准控件与事件模型.....	77	9.4 自定义控件.....	114
7.1 网页中的控件.....	77	9.5 分组校验技术.....	115
7.1.1 控件类型.....	77	9.6 综合示例.....	117
7.1.2 HTML 控件.....	77	9.7 小结.....	117
7.1.3 网页标准控件.....	78	9.8 习题.....	118
7.2 ASP.NET 2.0 的事件处理模型.....	85	第 10 章 ADO.NET 简介.....	120
7.2.1 基于服务器的处理模型.....	86	10.1 从 ODBC 到 ADO 数据库 的通用接口.....	120
7.2.2 尽量减少信息的往返次数.....	86	10.1.1 ODBC 通用接口.....	120
7.2.3 结合浏览器处理事件.....	87	10.1.2 ADO 通用接口.....	121
7.3 Web 表单.....	89	10.2 ADO.NET 的数据模型.....	122
7.3.1 对比 HTML 表单.....	89	10.2.1 数据访问的层次结构.....	122
7.3.2 ASP.NET 2.0 的 Web 表单.....	90	10.2.2 数据集与数据提供者.....	123
7.4 应用示例.....	91	10.3 数据源控件.....	126
7.5 小结.....	97	10.3.1 概述.....	126
7.6 习题.....	97	10.3.2 数据源控件的类型.....	127
第 8 章 状态管理.....	99	10.4 小结.....	128
8.1 状态的类型.....	99	10.5 习题.....	129
8.2 视图状态.....	99	第 11 章 利用 GridView 控件 显示数据.....	131
8.3 应用程序状态.....	101	11.1 数据绑定的基本概念.....	131
8.4 会话状态.....	102	11.2 显示数据表记录.....	132
8.4.1 概述.....	102	11.3 对数据表进行分页、 排序和选择.....	135
8.4.2 Session 对象中方法的调用.....	103	11.4 利用模板美化显示.....	136
8.5 Cookie 状态.....	104	11.4.1 模板.....	136
8.6 简单的应用示例.....	106	11.4.2 自动套用格式.....	137
8.7 Web 窗体页的生命周期.....	107	11.4.3 设置模板样式.....	137
8.8 小结.....	108	11.5 显示记录中的图像.....	138
8.9 习题.....	108		
第 9 章 数据验证.....	110		
9.1 概述.....	110		
9.2 验证控件的类型.....	110		

11.6 小结	140	14.4.1 Wizard 控件的用途	170
11.7 习题	140	14.4.2 Wizard 控件的结构	171
第 12 章 数据库查询与同步	141	14.4.3 Wizard 控件的使用方法	171
12.1 数据库查询	141	14.5 小结	172
12.1.1 数据库查询语句	141	14.6 习题	172
12.1.2 简单查询	142	第 15 章 存储过程与数据缓存	174
12.1.3 组合查询	144	15.1 概述	174
12.2 数据表同步	145	15.2 创建存储过程	175
12.2.1 概述	145	15.3 调用存储过程	177
12.2.2 同一窗体页中父、 子表同步	146	15.4 数据缓存	178
12.2.3 不同窗体页中父、 子表的同步	147	15.4.1 网页输出缓存	178
12.3 小结	149	15.4.2 数据库缓存	179
12.4 习题	149	15.4.3 缓存池	180
第 13 章 编辑数据表	151	15.5 小结	181
13.1 数据表编辑的 SQL 语句	151	15.6 习题	181
13.2 使用 GridView 控件更新数据表	152	第 16 章 主题、用户控件和主控页	183
13.3 使用 GridView 控件的列模板	154	16.1 主题	183
13.3.1 选择显示的字段	155	16.1.1 概述	183
13.3.2 增添按钮	155	16.1.2 创建主题的方法	183
13.3.3 使用模板列	155	16.1.3 同一控件多种定义 的方法	184
13.4 在 GridView 控件中增添记录	160	16.1.4 将主题文件应用于 整个应用程序	186
13.5 使用 DetailsView 控件	161	16.2 用户控件	186
13.6 小结	164	16.2.1 什么是用户控件	186
13.7 习题	164	16.2.2 创建用户控件的方法	186
第 14 章 网上招聘与留言板	166	16.2.3 使用用户控件	187
14.1 概述	166	16.2.4 代码分析	187
14.2 利用 FormView 控件 设计招聘网页	166	16.2.5 将 Web 窗体页转换 为用户控件	188
14.3 利用 FormView 控件 设计留言板	168	16.3 主控页	189
14.3.1 进入留言板	169	16.3.1 什么是主控页	189
14.3.2 留言板网页的界面设计	169	16.3.2 创建主控页的方法	189
14.3.3 对留言板的管理	170	16.3.3 在主控页中放入新网页 的方法	190
14.4 使用 Wizard 控件	170	16.3.4 将已建成的网页放入 主控页中	191

16.4 小结	192	18.5 直接调用 Membership API 方法	214
16.5 习题	192	18.6 小结	216
第 17 章 网站导航	194	18.7 习题	216
17.1 TreeView 控件	194	第 19 章 购货车与订单	218
17.1.1 概述	194	19.1 概述	218
17.1.2 选择 TreeView 控件 的视图	194	19.1.1 Northwind 样板库简介	218
17.1.3 编辑节点	195	19.1.2 本系统的功能	219
17.1.4 对节点事件的处理	196	19.1.3 主要数据表的结构及 其他准备工作	221
17.2 站点地图文件	197	19.2 设计用户控件及主控页	221
17.3 将 TreeView 结合站点 地图进行导航	198	19.3 设计主页	222
17.4 利用动态菜单进行导航	199	19.4 产品类型与产品目录之间同步	223
17.5 使用 SiteMapPath 控件	199	19.5 选择商品	224
17.6 小结	200	19.5.1 选择按钮触发的事件	224
17.7 习题	200	19.5.2 如何取出数据	224
第 18 章 基于角色的安全技术	202	19.5.3 如何创建数据表	225
18.1 基于角色的安全技术的特点	202	19.6 显示购货车	226
18.1.1 网站中可以包括多个 入口	202	19.6.1 输入客户标志	227
18.1.2 基于角色的安全 技术是有层次的	203	19.6.2 在 GridView 中增加按钮	227
18.2 ASP.NET 2.0 基于角色的 安全技术的特点	203	19.6.3 增加模板列	228
18.3 基于角色的安全技术 的准备工作	204	19.6.4 在窗体页的 Page_Load 事件中增加条件	228
18.3.1 组织好站点中的文件	204	19.6.5 在 GridView 的 RowCommand 事件中 编写代码	228
18.3.2 利用网站管理工具 进行安全配置	205	19.7 结账	229
18.4 利用控件创建安全页	208	19.7.1 结账网页的数据显示	229
18.4.1 用户登录控件	208	19.7.2 在 GridView 控件中 增加复选框	230
18.4.2 使用创建新用户控件	210	19.7.3 汇总的计算	230
18.4.3 登录状态与登录 姓名控件	211	19.8 保存订单	231
18.4.4 登录视图控件	212	19.9 查看订单	232
18.4.5 PasswordRecovery 控件和 ChangePassword 控件	213	19.10 小结	233
		19.11 习题	234
		第 20 章 创建三层架构	237
		20.1 从两层架构发展成三层架构	237

20.2 ASP.NET 2.0 中间层的特点.....	238	21.6 小结.....	259
20.3 创建中间层的步骤.....	239	21.7 习题.....	259
20.4 在网页中调用中间层对象.....	241	附录 A C#常用语法简介	261
20.4.1 直接调用中间层对象.....	241	A.1 数据类型.....	261
20.4.2 通过 ObjectDataSource 数据源控件调用中间层.....	242	A.1.1 常量与变量.....	261
20.5 小结.....	243	A.1.2 值类型.....	261
20.6 习题.....	244	A.1.3 引用类型.....	263
第 21 章 网站的个性化服务	245	A.1.4 装箱与拆箱.....	267
21.1 概述.....	245	A.2 运算符与表达式.....	268
21.2 ASP.NET 2.0 对个性化 设计的支持.....	246	A.2.1 算术运算符与 算术表达式.....	268
21.2.1 关于 Membership.....	246	A.2.2 赋值运算符与 赋值表达式.....	268
21.2.2 关于 Profile.....	246	A.2.3 关系运算符与 关系表达式.....	268
21.2.3 关于 WebParts.....	248	A.2.4 逻辑运算符与 逻辑表达式.....	268
21.3 保留用户关心的数据.....	248	A.2.5 位运算符.....	269
21.4 WebParts 介绍.....	250	A.2.6 对象创建运算符.....	269
21.4.1 定制网页时能够 执行的任务.....	250	A.2.7 其他运算符.....	269
21.4.2 WebParts 的分类.....	251	A.3 流程控制语句.....	269
21.5 定制主页.....	251	A.3.1 分支控制语句.....	270
21.5.1 创建简单的包含 WebPart 控件的网页.....	251	A.3.2 循环控制语句.....	271
21.5.2 创建可以编辑和 改变布局的网页.....	254	A.3.3 异常处理语句.....	273
21.5.3 运行中增添 WebParts 控件.....	257	附录 B 部分习题参考答案	275

第 1 章 .NET 框架简介

近几年来,因特网以超乎人们想象的速度向前发展,不仅网民大量增加,因特网的服务功能也得到成倍的扩展。现在的因特网除用于发布和收集信息以外,还能完成诸如远程教育、网上银行、电子政务、电子商务等项服务工作。这些服务工作常常涉及到大量的数据处理和计算,需要客户与服务器之间频繁地进行交互,实际上已经成为一种多层分布式的应用系统。

网站是以浏览器或其他数字设备作为客户端的网络应用程序。它是网络应用中最受欢迎的一部分,由于对网络应用的需要如此广泛,又由于网站与桌面系统相比有很大的不同,因此网站的开发目前已经成为所有软件开发中最重要的组成部分。

ASP.NET 2.0 是微软 2005 年年底推出的新一代企业级应用平台,是微软推出的三大重要产品之一。ASP.NET 2.0 虽然从 ASP.NET 1.1 版本发展而来,但是经过多年的研发,在很多方面都有突破性的进展。本书的目的就是帮助读者学会使用这个新一代平台快速开发功能强大、运行可靠、易于扩展的网站。作为本书的开头,本章将首先对 ASP.NET 的基本情况做一些简单介绍。本章的具体内容包括:

- 网站发展历史的简要回顾
- .NET 框架的体系结构
- XML 可扩展的标记语言

1.1 网站发展历史的简要回顾

1.1.1 从静态网页发展到动态网页

早期的 Web 服务器传送的是静态网页,虽然网页中包括有文字和图片,但是只要不改变设计,网页的内容是不会变化的。对静态网页的访问过程如下:

- (1) 客户通过 Web 服务器的 URL 申请页面。
- (2) 服务器向客户送回被申请的页面。
- (3) 在客户端下载并在浏览器上显示页面。
- (4) 断开客户与服务器之间的联系。

整个过程比较简单,到客户端下载完页面时为止,整个过程就结束了。用于发布静态网页的网站设计也比较简单。这种设计对于早期的网站来说也许已足够。因为早期使用网站的大多是一些科学工作者,他们关注的重点只是交流有关科学技术的内容。

随着因特网应用领域的扩展,各种不同类型的客户加入到网络中来,不少客户很快就提出了新的要求。例如,有的客户提出,能不能代我查阅一下我银行存款的变化情况?要满足类似这样的需求,服务器的工作就不那么简单了。它首先要查阅银行账户,进行必要的计算和统计,再将结果反馈给客户。这就是说,服务器在回答问题前必须先执行一些相

关的程序。这段程序不仅应能回答客户的问题，还要能够保障客户的信息安全，防止其他人进行查阅或破坏。

类似这种网页的输出内容将随程序执行的结果而有所不同。这样的网页被称之为“动态网页”。访问动态网页的过程如下：

- (1) 客户通过 Web 服务器的 URL 申请一个网页。
- (2) 服务器接收请求，并处理网页上的代码。
- (3) 将代码的处理结果转换成 HTML 代码后向客户送出。
- (4) 在客户端下载并在浏览器上显示网页。
- (5) 服务器断开与客户的联系并转向其他客户，以便提供新的服务。

和静态网页相比，动态网页的处理上多了一个处理代码的过程。用什么方式来处理代码，在不同的历史时期采用了不同的技术，大体上可以划分为 3 个阶段。

1.1.2 动态网页发展的几个阶段

1. CGI 阶段

CGI 是英文 Common Gateway Interface 的缩写，代表服务器端的一种通用(标准)接口。每当服务器接到客户更新数据的要求以后，利用这个接口去启动外部应用程序来完成各类计算、处理或访问数据库的工作，处理完后将结果返回 Web 服务器，再返回浏览器。外部应用程序是用 C、C++、Perl、Pascal、Java 或其他语言编写的程序，程序运行在独立的地址空间中。具体情况如图 1.1 所示。

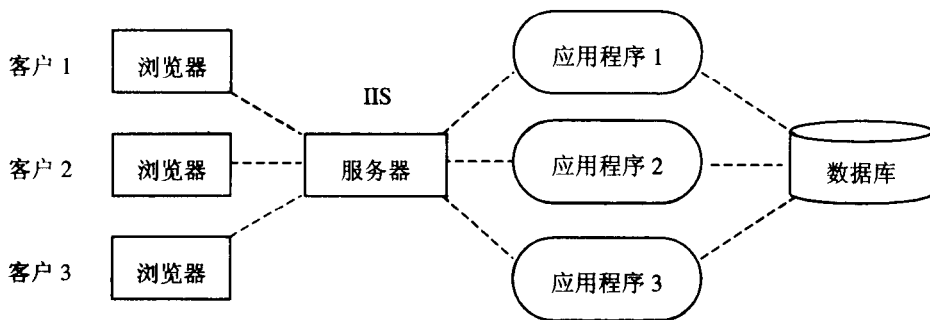


图 1.1 CGI 示意图

后来出现了“ISAPI”(用于 Internet Explorer 浏览器)或“NSAPI”(用于 Netscape 浏览器)技术，其功能与 CGI 相同，但技术方面有些改进。外部应用程序改用动态链接库(DLL)，被载入 Web 服务器的地址空间运行，并且用“线程”代替“进程”，因而显著地提高了运行效率。但不论是 CGI 还是 ISAPI 或 NSAPI，都需要编写外部应用程序，而编写外部应用程序并不是一件容易的事情。从开发人员的角度讲，这种开发方式并没有带来开发上的方便。

2. 脚本语言阶段

这个阶段出现了许多杰出的脚本语言，如 ASP、PHP、JSP 等。脚本语言的出现大大简化了动态网站开发的难度，特别是 ASP 和 PHP 学习简单、功能强大，成为许多网站开

发者的首选。

JSP 与 ASP 的程序结构非常相似。它的主要特点是在传统的 HTML 网页文件中加入 Java 程序片段(Scriptlet)和使用各种各样的 JSP 标志(Tag), 构成 JSP 网页。Web 服务器在接收客户的访问要求时, 首先执行其中的程序片段, 并将执行结果以 HTML 格式返回客户。

3. 组件技术阶段

ASP.NET 和 Java(J2EE)技术是这个阶段的代表。这是一个由类和对象(组件)组成的全面面向对象的系统, 采用编译方法和事件驱动方式运行。系统具有高效、高可靠、高可扩展的特点。详细情况将在下面各章节中重点讲述。

1.1.3 ASP 的技术特点

1996 年 11 月微软公司推出了 ASP 技术。什么是 ASP 技术?

ASP 是英文 Active Server Page 的缩写, 是用来开发 Internet 或 Intranet 应用程序的技术。按照微软公司自己的定义: “ASP 是一种服务器端的脚本技术, 用来创建动态的、交互的网站。”它的特点是, 将 HTML 标记以及 Script 程序代码组织在一个网页文件中, 代码以 VBScript 或 JavaScript 脚本语言为基础。当客户访问 ASP 网页时, 服务器(PWS 或 IIS)将启动 ASP 文件的解释程序在服务器端执行。执行后, 动态地产生 HTML 文件, 然后传送到客户端的浏览器中显示。

ASP 采用的脚本语言使用起来比较简单, 除此以外系统还提供了几个内建对象, 利用它们创建动态网页比较容易, 因此 ASP 一出现立即受到普遍的欢迎, 一批用 ASP 技术创建的网站相继出现。

但是近几年来, 人们对网站提出了更多、更高的要求。电子商务的出现就是一个典型的例子。在电子商务中, 购物的过程不仅需要客户与服务器之间进行多次交互, 还涉及到各类网站之间的交互和计算。例如, 要通过认证机构证明双方的身份; 要通过银行网站付款或转账; 通过税务部门纳税; 通过保险公司保险; 通过物流公司发货等。

与此同时网络本身也有长足的进步: 服务器功能的增强, 传输网带的增宽, 速度的增快。新标准 XML/SOAP 的诞生等, 给网络技术的发展提供了更加雄厚的基础。

在这样的形势下, 新的要求与新的技术基础呼唤着新技术的诞生。

1.1.4 新一代 ASP.NET 技术

进入 21 世纪以来, 微软公司鲜明地提出了 .NET 的发展战略, 确定了创建下一代 Internet 平台的目标。什么是下一代 Internet? 下一代 Internet 的主要特征之一就是, 它将无处不在, 世界上任何一台智能数字设备都有可能通过宽带连接到因特网上。因此, 作为下一代 Internet 的平台应该实现以下要求:

- 为各种类型的客户服务。不仅能为现有的计算机、手提式计算机、移动电话等客户服务, 还要能为未来可能加入因特网的智能设备(如电视机、电冰箱、洗衣机等)提供服务。
- 强大的交互和运算能力。

- 跨平台交换数据的能力。
- 快速设计和部署的能力。
- 强有力的信息安全保障能力。

在这些思想的指导之下, 微软于 2000 年推出了基于 .NET 框架的 ASP.NET 1.0 版本, 2002 年推出了 ASP.NET 1.1 版本, 2005 年年底又推出了 ASP.NET 2.0 版本。ASP.NET 是在 ASP 的基础上发展起来, 但它不只是 ASP 的升级, 而是重新构筑的一个全新的系统。由于 ASP 存在着“先天”不足, 用修补、增添的办法已经很难满足要求。关键的问题有: ASP 不是一个完全的面向对象的系统, 它使用的脚本语言虽然简单、灵活, 但属于弱类型语言, 功能不强而且容易出错, 系统提供的内建对象也只有几十个, 同时 ASP 通过解释来执行代码, 效率比较低, 等等。

ASP.NET 是建立在 .NET 框架平台上的完全面向对象的系统。ASP.NET 与 .NET 框架平台紧密结合是 ASP.NET 的最大特点。有了 .NET 框架的支持, 一些单靠应用程序设计很难解决的问题, 都可以迎刃而解。 .NET 框架平台给网站提供了全方位的支持, 这些支持包括:

- 强大的类库。利用类库中的类可以生成对象组装程序, 以实现快速开发、快速部署的目的。
- 多方面服务的支持。如智能输出(对不同类型的客户自动输出不同类型的代码)、内存的碎片自动回收、线程管理、异常处理等。
- 允许利用多种语言对应用进行开发。
- 跨平台的能力。
- 充分的安全保障能力。

1.2 .NET 框架的体系结构

在 .NET 框架中使用了很多全新的技术, 带来了根本性的、深层次的创新。框架给因特网构筑了一个理想的工作环境。在这个环境中, 用户能够在任何地方、任何时间、使用任何设备从 Internet 网中获得所需要的信息, 而不需要知道这些信息存放在什么地方以及获得这些信息的细节。

.NET 框架的体系结构包括 5 大部分, 它们是:

- 程序设计语言及公共语言规范(CLS)
- 应用程序平台(ASP.NET 及 Windows 应用程序等)
- ADO.NET 及类库
- 公共语言运行库(CLR)
- 程序开发环境(Visual Studio.NET)

其结构如图 1.2 所示。 .NET 框架的上述结构可以简化为图 1.3。

下面重点介绍 .NET 框架平台使用的语言、平台的作用以及采用的通信协议三方面的问题。

1. .NET 框架使用的语言

在.NET 框架上可以运行多种语言,这是.NET 的一大优点。.NET 框架中的 CLS 实际上是一种语言规范。由于.NET 框架支持多种语言,并且要在不同语言对象之间进行交互,因此就要求这些语言必须遵守一些共同的规则。公共语言规范(Common Language Specification, CLS)就定义了这些语言的共同规范,它包括了数据类型、语言构造等,同时 CLS 又被设计得足够的小。

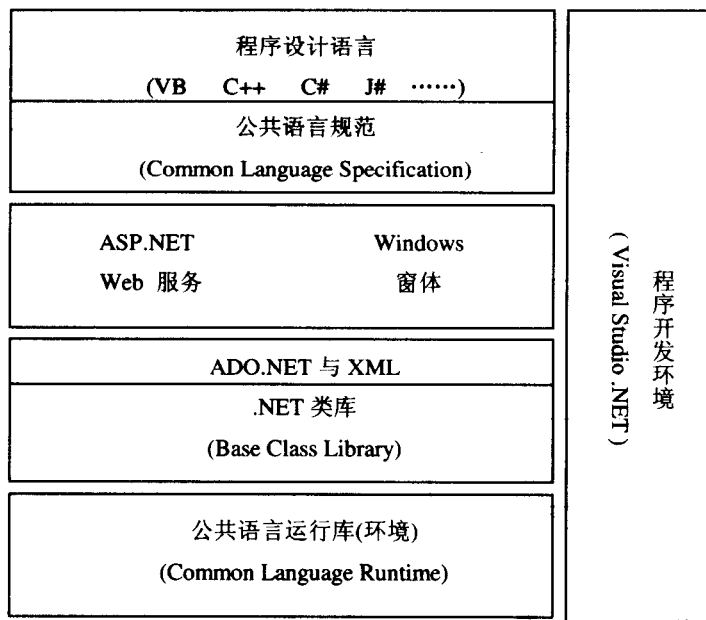


图 1.2 .NET 框架的体系结构

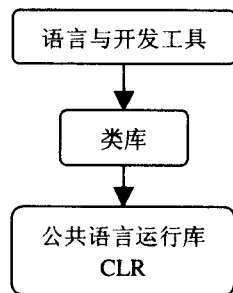


图 1.3 框架的简化图

凡是符合 CLS 规范的语言都可以在.NET 框架上运行。目前已经有 C# .NET、VB .NET、C++ .NET、J# .NET、Jscript .NET 等(VBScript 已不再使用)。预计还将有 20 多种语言可以运行在.NET 框架中。目前,有些公司还在创建符合 CLS 规范的自己的语言。

JavaScript 是各类浏览器采用的通用语言。传统的 JavaScript 是一种基于面向对象的脚本语言,现在 ASP.NET 中采用的 Jscript .NET 与 JavaScript 语言完全兼容,但却已将它改造成成为一种完全面向对象的语言,不仅给语言增添了很多新功能,还得到.NET 框架的完全支持。

由于多种语言都运行在.NET 框架之中,因此它们功能都基本相同,只是语法有区别。程序开发者可以选择自己习惯或爱好的语言进行开发。VB .NET 和 VC .NET 与原来的 VB、VC 相比已经有很多地方不兼容。VB .NET 和 VB 相比变化更大,VB .NET 是一种完全面向对象的语言(而 VB 只是基于面向对象的语言)。Visual J# 是.NET 框架 1.1 版本以后才增加进来的语言,供原来使用 Java 语言的程序员转向使用.NET 框架的应用程序时使用。

Visual C#是为.NET 框架“量体裁衣”开发出来的语言,非常简练和安全,最适合于在.NET 框架中使用。本书的示例都是用 C#编写的。C#的简要语法请参阅本书的附录 A。

各种语言经过编译后,并不直接产生 CPU 可执行的代码,而是先转变为一种中间语言(Intermediate Language,简称 IL 或 MSIL)。执行时再由公共语言运行库载入内存,通过实时解释将其转换为 CPU 可执行代码。转换的过程如图 1.4 所示。

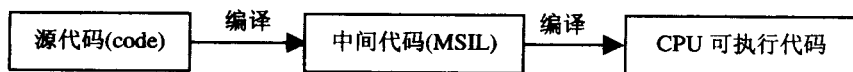


图 1.4 从源代码到 CPU 可执行代码的转换过程

为什么要设置中间语言?设置中间语言是为了跨平台的需要。源程序经过编译转换为中间语言(IL 或 MSIL)。各类平台只要装上不同的转换引擎,就可以将其转换为本 CPU 需要的代码^①。由于中间语言类似于汇编语言,与二进制代码非常接近,因此实时解释的速度也很快。

转换的过程如图 1.5 所示。

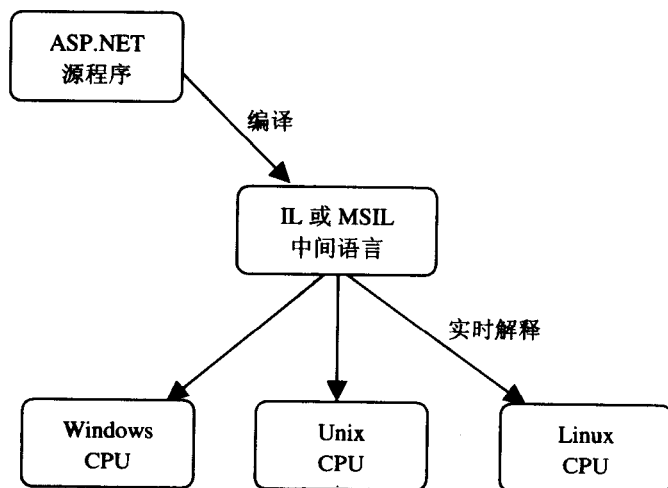


图 1.5 利用中间语言转换的过程

2. 类库

.NET 框架的另一个主要组成部分是类库,包括数千个可重用的“类”。各种不同的开发语言都可以用它来开发传统的命令行程序或者图形用户界面(GUI)应用程序。

.NET 框架中的类被拆分为命名空间。命名空间(NameSpace)是类库的逻辑分区。类库所采用的命名空间呈层次结构,即命名空间下面又可以再分成子命名空间。每个命名空间都包含一组按照功能划分的相关的类。这样,一个大型的.NET 库就变得易于理解和便于使用。

例如:

- 所有微软公司提供的类都以 System 或 Microsoft 命名空间开头。
- 有关网络协议和简单的编程接口的类放在 System.Net 命名空间中。

^① 到目前为止除 Windows 以外,还没有其他操作系统安装有这样的引擎。

- 有关文件 I/O、内存 I/O、独立存储的类放在 System.IO 命名空间中。
- 基于 Windows 应用程序的用户界面的类放在 System.Windows.Forms 命名空间中。
- 有关 Web 服务器与浏览器交互，以及 Web 服务的类都放入 System.Web 及其子命名空间中。
- 所有用于处理 XML、XML 架构、XSL/T 转换、Xpath 表达式的类都放入 System.Xml 及其子命名空间中。

3. 公共语言运行库 (CLR)

公共语言运行库 (CLR，也称公共语言运行环境) 就相当于 Java 体系中的“虚拟机”，它是 .NET 框架的核心。它提供了程序运行时的内存管理、垃圾自动回收、线程管理和远程处理以及其他系统服务。同时，它还能监视程序的运行，进行严格的安全检查和维护工作，以确保程序运行的安全、可靠以及其他形式的代码的准确性。

在 ASP.NET 2.0 新版本中，CLR 又增加了用户认证、角色授权以及个性化服务等新的服务项目。

运行库不仅提供了多种软件服务，同时也为以往的软件提供了支持。托管和非托管代码之间的互操作性使开发人员能够继续使用原来开发的 COM、ActiveX 控件和 DLL 动态链接库等。

1.3 XML：可扩展标记语言

当前，如何解决跨平台交换数据的问题，已经成为 Internet 进一步发展的“瓶颈”。为了解决这一问题，各个公司都曾经花费了大量的人力和物力。例如微软公司开发的 DCOM，SUN 公司开发的 CORBA 等，但都没能完全解决问题。现在 XML 的出现为解决这类问题提供了最好的机会。

1.3.1 什么是 XML

XML (Extensible Markup Language) 是一种可以扩展的标记语言，用来描述层次化的文档。XML 是由 World Wide Web (W3C) 1998 年发布的标准，到现在已经发展成熟。下面先举一个简单的示例来说明什么是 XML。

一个简单的 XML 文档的结构如下：

```
<root>
  <a>
    <b>...</b>
    <c>...</c>
    <d>...</d>
  </a>
  <a>
    <b>...</b>
    <c>...</c>
```