

代码详尽的实用手册

ASP.NET

开发人员手册

Pure
ASP.NET

[美] Robert Lair 著
Jason Lefebvre
张俊 何长松 等译

SAMS



电子工业出版社
Publishing House of Electronics Industry
www.phei.com.cn

ASP.NET 开发人员手册

Pure ASP.NET

[美] Robert Lair 著
Jason Lefebvre

张 俊 何长松 等译

電子工業出版社

Publishing House of Electronics Industry

北京 · Beijing

内 容 简 介

这是一本代码详尽的程序开发人员用书,其最大特点是使用大量“商业级”的程序代码来传授企业应用程序的开发技巧。书中所有的示例都分别提供C#和Visual Basic.NET两种语言的代码,便于读者的对比学习。本书的第一部分简明地介绍了ASP.NET框架,第二部分以大量的代码实例讨论ASP.NET的各个方面,最后是使用广泛的Microsoft .NET名字空间的完全参考。本书涉及ASP.NET的全面细节,主要包括通用语言运行时、确认控件、Web控件、列表控件、用户控件、组件构建、ADO.NET和XML Web服务等。

代码详尽是本书的特点,书中的代码可以从原出版商的站点下载。本书是Web开发人员编程中随时查阅的必备手册用书。

Authorized translation from the English language edition published by Sams Publishing. Copyright © 2002. All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher. Simplified Chinese language edition published by Publishing House of Electronics Industry, Copyright © 2002.

本书中文简体版专有翻译出版权由Pearson教育集团所属的Sams Publishing授予电子工业出版社。其原文版权及中文翻译出版权受法律保护。未经许可,不得以任何形式或手段复制或抄袭本书内容。

著作权合同登记号:图字:01-2001-3312

图书在版编目(CIP)数据

ASP.NET开发人员手册/(美)莱尔(Lair, R.)等著:张俊等译.-北京:电子工业出版社,2002.4
书名原文:Pure ASP.NET
ISBN 7-5053-7544-X

I. A... II. ①莱... ②张... III. 主页制作-程序设计 IV. TP393.092

中国版本图书馆CIP数据核字(2002)第018434号

责任编辑:赵宏英

印 刷:北京兴华印刷厂

出版发行:电子工业出版社 www.phei.com.cn

北京市海淀区万寿路173信箱 邮编:100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:31.25 字数:780千字

版 次:2002年4月第1版 2002年4月第1次印刷

定 价:49.00元

凡购买电子工业出版社的图书,如有缺页问题,请向购买书店调换。若书店售缺,请与本社发行部联系。
联系电话:(010) 68279077

译 者 序

程序开发是一门实践性很强的工作,以详细的代码实例来学习和掌握一门语言是较为便捷的方法。在具体的编程工作中,我们要经常查看帮助文档和手册,以了解如何使用一个函数、变量、方法和属性等。这就是一本有关ASP.NET的详细的手册用书,可以在Web应用程序开发中随时提供帮助。本书的主体是示例代码,全书分为三个部分,首先是对ASP.NET的快速介绍;第二部分详细讨论ASP.NET的方方面面,这一部分的最大特点是其代码的密集性;第三部分是最广泛使用的Microsoft .NET名字空间的完整参考,其特点仍然是代码密集性。每一个示例都提供了C#和Visual Basic.NET两种代码形式,以便于读者对比学习。

在前两年.com公司炙手可热的发展浪潮中,Web开发是一个令人羡慕的好工作。世界经济发展的放缓,使得全球IT界都处于全面调整的艰难之中,作为新经济代表的互联网业当然也不例外。但是,任何一个有远见的人都不能忽视这个行业的发展潜力。因此,译者不认为在这样的气氛中没有出版这类书籍的必要。冬天是一个蓄势待发的季节,相信我们会很快拥抱明媚的春天。

本书由张俊和中华全国归国华侨联合会的何长松担任主要翻译工作,以下人士参与部分翻译工作并提供了大力帮助:魏玮、孙华、何晓波、赵宏亮、王俭保、李刚、王勇、李胜利、梁天锋、赵彩云、唱文丽、胡凤燕、党晓青、张锦,感谢他们的支持。还要感谢《程序员》杂志孟迎霞女士提供的帮助。译文中难免有疏漏乃至谬误之处,还请广大业界同仁海涵、斧正。

关于作者

Robert Lair 是一位国际知名的程序员、技术作家和培训者，他在位于俄亥俄州代顿市的 Wright State 大学学习计算机科学与工程，并已经在软件开发领域工作了5年以上。他在软件产业的最新成果包括他在 IBuySpy 项目（Vertigo Software 公司）上的工作，以及在展示 ASP.NET 新功能的演示程序上的贡献。他职业生涯中的成果还包括 ProposalMaster 和 RFPMaster 项目（Sant Corporation 公司）、OpenTable.com 项目（Vertigo Software 公司）以及 Shoes.NET 项目（Vertigo Software 公司），最近一个项目是在 DevDays 2000 上展示 Microsoft.NET 企业服务器的演示程序。他业务上的主要客户包括 Microsoft 公司、MCI 公司、Williams Communications 公司和 Prognostics 公司。

Robert 还在讨论 Microsoft.NET 和 ASP.NET 的技术会议上提供技术培训，并就如何通过简化复杂应用程序的维护来增加软件开发过程中的效率提供培训。他曾经在 ASP Connections 会议和 ASP Developer's Conference 会议上发表演说。在 Vertigo Software 公司和 Sant Corporation 公司工作期间，他出席了大量国际技术会议。

Robert 给诸如 “Visual C++ Developer's Journal”，“Visual Basic Programmer's Journal” 和 “.NET Developer” 等美国著名杂志著作了大量的论文，他的论文包括广泛的论题，包括 C#，Microsoft.NET，ASP.NET 和 Visual Basic.NET。Robert 还是 “.NET Developer” 一个每月专栏的作者，这个专栏深入讨论 IBuySpy 的应用程序。

Jason Lefebvre 在一家直销服务公司 Protocol（www.protocolusa.com）任程序员。他在位于俄亥俄州代顿市的 Wright State 大学学习数学和计算机科学，并已经在软件开发领域工作了5年以上。他与 Robert 一起参加了上面所提到的所有项目。

Jason 给诸如 “Visual C++ Developer's Journal” 等国际著名杂志著作了大量的论文，他的论文涉及广泛的论题，包括 C#，Microsoft .NET，ASP.NET 和 Visual Basic.NET。

关于技术编辑

John Timney 与妻子 Philippa 一起住在英格兰北部的一个小城，他是 Nottingham 大学的研究生，在获得管理系统学士学位之后，获得了信息科学硕士学位，并从 Humberside 大学获得信息科学的研究生毕业证书。John 致力于 Internet 解决方案的工作，他在计算技术方面的造诣使他获得了 Microsoft MVP（Most Valuable Professional，最优秀专家）奖。

引 言

在阅读本书之前,应该理解本书是如何组织的,这可以帮助读者确定自己要阅读什么内容,以及何时需要阅读这些内容。这里提供本书组织结构的一个概览。

本书是为对编程结构有基本的理解并对 HTML 有些许理解的程序员而编写的,但是,本书并不要求读者具有对 Microsoft.NET 框架的任何预备知识。

本书分为具有逻辑联系的三个部分。第一部分是对 ASP.NET 的概述,包括 8 个章节,是针对初次接触 ASP.NET,需要了解 ASP.NET 及其相关技术的程序员准备的。第一部分中极少包含代码,因而,阅读这一部分之后,几乎所有的程序员都可以理解 ASP.NET 的基本概念。

第二部分是有关 ASP.NET 程序开发的内容,是全书的核心部分,这些章节都是建立在第一部分提供的的内容的基础上的。在这一部分,我们将从零开始学习 Web 表单 (Web Form) 和 Web 服务 (Web Service), 还将学习创建组件和用户控件的技巧。第二部分在提供大量的代码的同时,还提供了掌握这些技巧需要的技术讨论。

第三部分是 Microsoft .NET 参考,提供对 System.Collection, System.Data.SqlClient, System.Web 和 System.Web.UI.WebControls 名字空间的参考。这是本书代码最为密集的部分,只提供很少对概念的讨论。这里包括很多例子,这些例子演示如何使用最重要的 Microsoft.NET 名字空间中的属性和方法。

目 录

第一部分 ASP.NET 概述

第 1 章	ASP.NET 和 Microsoft .NET 框架	2
1.1	Microsoft .NET 概述	2
1.2	Microsoft .NET 的目标	2
1.3	Microsoft .NET 框架技术	2
1.4	Microsoft .NET 框架的优点	5
第 2 章	通用语言运行时	8
2.1	通用语言运行时概述	8
2.2	Microsoft 中间语言	8
2.3	JIT 编译	8
2.4	部件	9
2.5	Microsoft.NET 运行时中的垃圾收集	10
第 3 章	ASP.NET Web 表单	13
3.1	服务器端代码块	13
3.2	ASP.NET Web 表单事件	17
3.3	Visual Studio 和后代码方法	19
第 4 章	ASP.NET 控件	20
4.1	ASP.NET 控件产生 HTML	20
4.2	视图状态管理	22
4.3	HTML 控件	23
4.4	Web 控件	24
4.5	自定义控件	25
第 5 章	列表控件	26
5.1	列表控件的类型	26
5.2	使用列表控件的优点	28
5.3	如何使用列表控件工作	32

第 6 章 确认用户输入	34
6.1 什么是确认控件	34
6.2 使用确认控件的优点	34
6.3 确认控件类型	35
6.4 显示错误	36
第 7 章 理解用 ADO.NET 进行数据访问	38
7.1 ADO.NET 的优点	38
7.2 ADO.NET 对象模型	39
第 8 章 Web 服务概述	41
8.1 什么是 Web 服务	41
8.2 为什么使用 Web 服务	42
8.3 发布 Web 服务	43
8.4 消耗 Web 服务	44

第二部分 ASP.NET 开发

第 9 章 使用 HTML 和 Web 控件创建 ASP.NET 页面	48
9.1 Web 控件技术概述	48
9.2 核心 Web 控件	48
第 10 章 用用户控件封装 ASP.NET 的页面功能	85
10.1 创建一个简单的用户控件	85
10.2 碎片缓存	90
第 11 章 使用 ASP.NET 列表控件	91
11.1 为列表控件捆绑数据	91
11.2 Reapter 列表控件	104
11.3 DataList 列表控件	108
11.4 DataGrid 控件	121
第 12 章 使用 ASP.NET 确认控件	149
12.1 确认控件通用的属性和方法	149
12.2 在 Web 表单上放置控件	150
12.3 格式化错误信息	150
12.4 RequiredField 确认控件	150
12.5 Range 确认控件	154
12.6 Compare 确认控件	156
12.7 RegularExpression 确认控件	158
12.8 Custom 确认控件	161

12.9 ValidationSummary Web 控件	164
第 13 章 使用 ADO.NET 进行数据访问	169
13.1 使用 Connection 对象	169
13.2 使用 Command 对象	170
13.3 使用 DataReader 对象	171
13.4 使用存储过程	180
13.5 使用事务	193
第 14 章 构建 ASP.NET 组件	197
14.1 什么是组件	197
14.2 使用组件的优点	197
14.3 Microsoft Windows DNA	198
14.4 组件和 Microsoft .NET	198
14.5 名字空间	198
14.6 类	200
14.7 COM 互操作性	207
14.8 COM+ 服务：使用来自 Microsoft .NET 的事务	209
第 15 章 构建 Web 服务	219
15.1 创建 Web 服务	219
15.2 在现有的应用程序中构建 Web 服务	221
15.3 使用 Web 服务	224
15.4 使用预编译的 Web 服务	229
15.5 从 Web 表单使用 Web 服务	230
15.6 从 Windows 表单使用 Web 服务	232
第 16 章 配置和优化 ASP.NET 应用程序	234
16.1 配置 ASP.NET 应用程序	234
16.2 优化 ASP.NET 应用程序	236
第 17 章 保护和部署 ASP.NET 应用程序	238
17.1 ASP.NET 安装优点	238
17.2 部署 ASP.NET 页面	238
17.3 部署组件	239
17.4 保护应用程序	240

第三部分 Microsoft.NET 名字空间参考

第 18 章 System.Collections 参考	246
18.1 ArrayList 类	247

18.2	BitArray 类	267
18.3	Hashtable 类	276
18.4	Queue 类	285
18.5	SortedList 类	293
18.6	Stack 类	308
第 19 章	System.Data.SqlClient 参考	316
19.1	SqlCommand 类	317
19.2	SqlConnection 类	326
19.3	SqlDataReader 类	333
19.4	SqlDbType 类	374
19.5	SqlParameter 类	375
19.7	SqlTransaction 类	389
第 20 章	System.Web 参考	393
20.1	HttpBrowserCapabilities 类	393
20.2	HttpCookie 类	403
20.3	HttpRequest 类	409
20.4	HttpResponse 类	427
第 21 章	System.Web.UI.WebControls 参考	441
21.1	WebControl 类	441
21.2	HyperLink 类	449
21.3	Button 类	451
21.4	Calendar 类	452
21.5	Label 类	464
21.6	Image 类	464
21.7	Panel 类	466
21.8	The TextBox Class	467
21.9	CheckBox 类	471
21.10	ImageButton 类	472
21.11	LinkButton 类	475
21.12	RadioButton 类	476
21.13	BaseValidator 类	479
21.14	CompareValidator 类	481
21.15	CustomValidator 类	483
21.16	RangeValidator 类	484
21.17	RegularExpressionValidator 类	485
21.18	RequiredFieldValidator 类	485
21.19	ValidationSummary 类	486
21.20	选择静态类	488

第一部分

ASP.NET 概述

- 第 1 章 ASP.NET 和 Microsoft.NET 框架
- 第 2 章 通用语言运行时
- 第 3 章 ASP.NET Web 表单
- 第 4 章 ASP.NET 控件
- 第 5 章 列表控件
- 第 6 章 确认用户输入
- 第 7 章 理解用 ADO.NET 进行数据访问
- 第 8 章 Web 服务概述

第1章 ASP.NET 和 Microsoft .NET 框架

1.1 Microsoft .NET 概述

随着 Microsoft .NET 框架的提出, Microsoft 已经致力于维持其在 Internet 开发方面的优势地位。新的框架基于提供大量的新工具, 使得当前范例从标准 Web 应用向真正综合的 Internet 用户技巧转换。

为确保该框架能够被广泛接受, Microsoft .NET 平台包括了开放的标准, 例如可扩展标记语言 (eXtensible Markup Language, XML) 和简单对象访问协议 (Simple Object Access Protocol, SOAP)。这使得 ASP.NET 很容易与支持这些开放标准的其他平台上的应用相互作用。

1.2 Microsoft .NET 的目标

Microsoft .NET 的两个主要目标是提供对 ASP (Active Server Pages) 以前版本的支持和提供一个较为简单的编程模型。

1.2.1 对 ASP 以前版本的支持

Microsoft .NET 平台的一个目标是使新的 ASP.NET 页面不费力地与以前的 ASP.NET 页面 (例如 ASP 3.0) 相集成。使得两种页面能很容易地在同一台机器上并存。事实上, ASP.NET 处理器仅尝试编译和运行 Microsoft .NET 框架安装过程中指定的文件类型, 例如 .aspx 和 .asmx 文件。因而, 在同一应用程序中可以同时使用 ASP 和 ASP.NET 页面。

此外, 通常只须对代码进行很小的改动, ASP 页面就可以容易地转换成 ASP.NET 页面。从 ASP 到 ASP.NET 更简易的转换方法是可以使用 ASP.NET 页面上的组件对象模型 (Component Object Model, COM) 和 COM+ 对象, 这使用了 Microsoft .NET 框架的互操作特性。要想了解 COM 对象的更多信息, 请参见第 14 章。

1.2.2 较简单的编程模型

Microsoft .NET 提供了比 COM 和 COM+ 更为简单的编程模型。COM 中一些最麻烦和最易混淆的方面, 如系统注册和全球惟一标识符 (GUID), 在 Microsoft .NET 编程模型中虽然仍然可以获取, 但却不是有效部分。开发者不再需要为在注册表中注册组件而担心, 更重要的是, 无须为不同版本中的组件保证二进制兼容而担心。

1.3 Microsoft .NET 框架技术

Microsoft .NET 框架使得几项新技术在 ASP.NET 中成为有效的。通用语言运行时 (Common

Language Runtime) 引擎给了 ASP.NET 真实的平台和语言独立性。它也提供了大量服务器端控件, 使得创建应用程序更加快速。

1.3.1 通用语言运行时引擎

Microsoft .NET 平台最有意义的一部分就是创建了通用语言运行时引擎。在 Microsoft .NET 平台上, 代码并不是编译成汇编机器代码, 而是编译成称为 Microsoft 中间语言 (Microsoft Intermediate Language, MSIL) 的一种中间代码。这就构成了 Microsoft .NET 平台成功的几项重要因素。有关这方面内容的更多细节, 可以参考第 2 章。

通用语言运行时引擎提供了两项主要优点: 平台独立性和语言独立性。

平台独立性

因为 ASP.NET 源代码并不针对特定的平台编译成二进制, 因此具有平台独立性。将来通用语言运行时引擎也可能用于其他平台, 例如 Linux 或 Macintosh 系统。ASP.NET 中的应用程序代码不需要在平台间重新编译。例如, 可以利用 Visual Studio.NET 和其他开发工具在 Windows 2000 环境下开发程序, 然后为满足用户的需求在基于 Unix 操作系统中对其编译和配置。

语言独立性

可以在 Microsoft .NET 允许范围内选择任何语言编制代码。因为所有代码都将编译成同样的中间语言, 所以语言的选择纯属个人的喜好问题。事实上, 同一应用程序中的不同部分也可以采用不同的语言编制。一个用 C# 编写的类可以参照 Visual Basic.NET 代码块, 就像已经用 Visual Basic.NET 编写这个类一样容易。

在本书写作时, 对 Microsoft .NET 开发仅有 4 种语言有效: C#, Visual Basic.NET, JScript 和 Managed C++。但是, 创建多种语言编译器的大量工作正在进行中, 这包括 Perl 和其他很多种语言。

1.3.2 ASP.NET Web 表单

Microsoft .NET 平台的核心是 Web 表单 (Web Form), 也称为 ASP.NET 页面。Web 表单是 ASP.NET 控件和其他 Web 表单元素 (如服务器端脚本、标准 HTML 和其他客户端代码) 的基本容器项。

1.3.3 ASP.NET 控件

ASP.NET 提供了大量丰富的服务器端控件, 这些控件能帮助开发者创建用于浏览器上的 Web 应用程序。通过自动探测发出请求的浏览器类型并为浏览器生成正确的 HTML, ASP.NET 控件去除了多种网络浏览器代码间的冲突, 这些冲突以前需要大量的编码时间。

ASP.NET 控件在服务器上仅使用单个代码行就可以为客户产生大量的 HTML。这加速了开发过程, 并增强了代码的可靠性。而且, 因为现存的 ASP.NET 控件已经过测试, 因而应用程序很容易调试。

开发者不限于使用现有的 Microsoft 控件。使用诸如 C# 或 Visual Basic.NET 语言很容易创建自己的服务器控件。可以根据要求的控件和仅用于特定浏览器的输出 HTML 确定浏览器类型,

从而创建控件。创建自己的控件可能需要消耗时间, 但将浏览器特定的显示逻辑封装于控件之中比在应用程序中控制更好, 这是因为:

- 当新版本的浏览器出现时, 仅需要更新自定义的控件即可。控件中的代码经过更新和测试后, 可以确保更新后的浏览器中应用程序的正确性。
- 可以在应用程序中使用自定义的控件, 而无须为 Web 表单中任何浏览器特定的显示逻辑担心。这些全部由控件进行控制。

1.3.4 视图状态管理

许多开发者在设计 Web 应用程序时面临的最大困难之一是保持状态 (也就是, 确保重载网页时多个表单中的值保持一致)。标准 ASP.NET 在编译后的公共网关接口 (CGI) 应用程序中 (使用诸如 C 和 C++ 等语言编写) 通过对 Cookie 和会话对象的简单访问对这一问题做了少量的改善。然而, 因为这些方法可能涉及到严重的性能问题, 所以许多开发者用隐藏的表单元素保持状态。

在 ASP.NET 应用中, 保持状态不再是一项艰巨的任务, 这完全归功于视图状态管理。视图状态管理是一种网页表单自动记忆表单内控件值的内置性能。这项性能去除了手工保持表单元素状态的必要。

1.3.5 确认控件

在开发 Web 应用程序时, 很多时间必须花费在创建用于确认用户输入的客户端脚本上。Microsoft .NET 提供了大量的内置确认控件, 这些控件使得开发者在无须客户端编码的情况下, 即可很容易地确认用户的输入。使用这些确认控件, 开发者可以在需要时标记表单元素, 同时也可以设置长度、比较或键入需要。Microsoft .NET 甚至包括基于 Perl 的正则表达式语言, 该语言可以用于确认更多高级字段元素 (例如, 确保电话号码字段仅包括带有破折号的数字)。如果存在比内置确认控件提供的更高级的确认需求, 则可以用少量的代码创建自定义的确认控件。

1.3.6 Web 服务

到目前为止, 从一台 Web 服务器远程调用另一台 Web 服务器上的方法仍是相当困难的。然而, 通过使用 Web 服务, 开发者可以创建一种组件, 该组件内的方法可以被其他 Web 服务器上的应用程序调用。图 1.1 显示了 Web 服务如何被找到 (exposed) 且如何被多个不同种类的客户使用。因为使用的是标准协议, 如 XML 和 SOAP, 因此 Web 服务在点对点通信以及共享数据方面比以前更容易。

代码重用

开发者可以通过创建可供任何客户使用的 Web 服务来创建页面和应用程序之间经常重用的公共方法。

支持标准 Internet 协议

为确保 Web 服务对多种客户有效, Microsoft .NET 包括了标准 Internet 协议。XML SOAP 提供了传送数据的标准格式, 这使得 Web 服务能够被运行在不同操作系统下的远程客户使用。

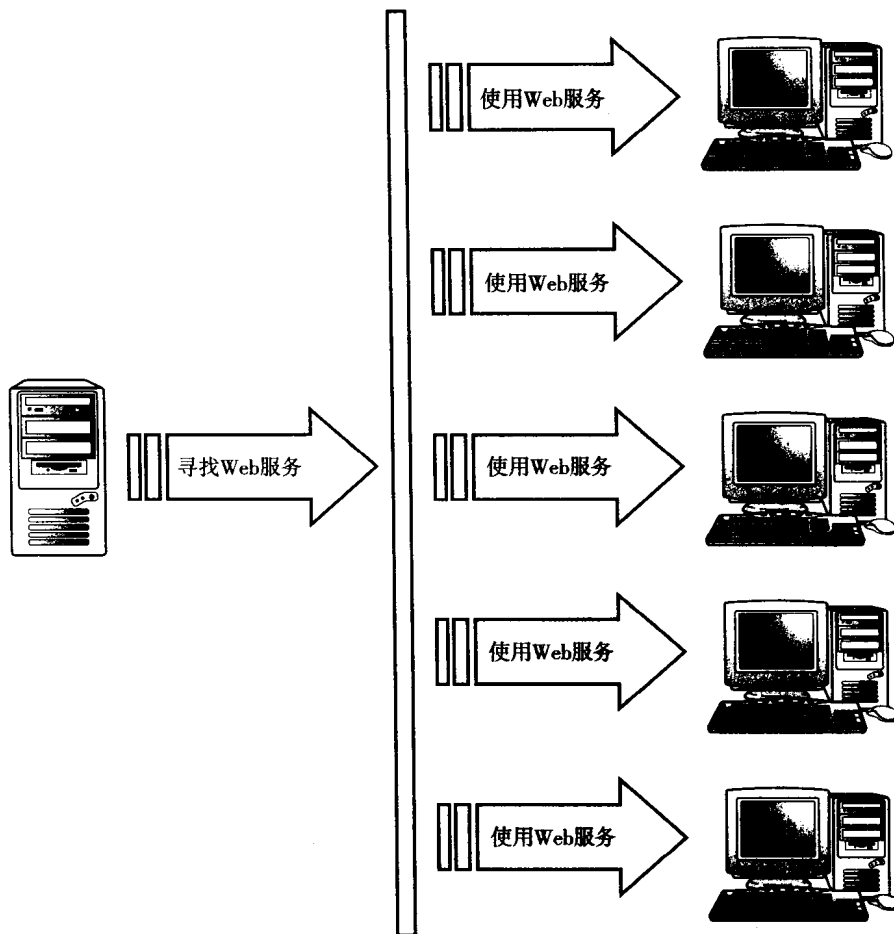


图 1.1 Web 服务的寻找和使用

1.4 Microsoft .NET 框架的优点

Microsoft .NET 框架有许多优点。本节较详细地对其中的一些优点进行了综述。

1.4.1 很少的代码行

通过使用 Web 控件, 开发者可以在编制用户界面时花费很少的时间。例如, 用较少的时间完成在支持 HTML 3.2 的任意浏览器上的大量完善页面功能的工作, 这使开发者有更多的时间集中于应用程序的设计和应用程序的全面流程。

此外, 由于 Microsoft .NET 框架语言健壮的类型特性, ASP.NET 强制开发者使用更可靠的开发技巧。

许多 ASP 应用问题归因于使用松散的类型语言。例如在 ASP 中, 如果需要给字符串变量分配一个整型值, 可以简单地使用 “=” 操作符, 整数值可以在后台转换成字符串。而在 ASP.NET 中, 这是绝对不允许的。在将整型值分配给字符串变量之前, 必须先将整数转换成 string 类型。

这听起来有些困难,但是使用 ToString() 函数很容易实现这一点,这一函数在 Microsoft .NET 框架中对所有整数都有效:

```
string myVar = 4.ToString();
```

健壮的类型变量有助于消除运行时发生类型转换带来的问题。在编译时而不是运行时能被解决的问题越多,应用程序就越强壮。

1.4.2 编译完全

Microsoft .NET 框架另一个重要的部分是可以实现完全编译,包括 ASP.NET 页面自身。一个页面在修改后第一次使用时,所有服务器端标识(诸如 Web 控件)、Web 表单以及服务器端代码块都将被编译,然后响应被创建并发送到客户。由于页面已经编译,因而任意用户对同一页面的并发点击可以更快。

1.4.3 配置轻松

Microsoft .NET 框架最令人感兴趣的优点之一是配置模型。在传统的 ASP 中,配置 Web 应用程序必须执行几个步骤。ASP 页面必须上传,同时应用中的所有组件必须上传并在操作系统中注册。这可以通过使用描述了所有步骤的详细 Windows 脚本主机 (.wsh) 文件来完成。这些脚本创建费时并且通常只用于大型项目。对较小的应用程序,特别是在面向大量站点的环境中,通常将应用文件上传,然后将组件发送给管理员注册。

在设计 Microsoft .NET 框架时,Microsoft 公司考虑到 ASP 的这些问题并提出了另一种方法。在 Microsoft .NET 中,组件可以在自己的机器上被远程编译,然后和页面一起上传并且放置在应用程序根目录的 \bin 路径下。除非需要注册继承的 COM 对象,否则无须注册。上传可以用标准的文件传输协议 (File Transfer Protocol, FTP) 执行。不需要本地服务器管理。

同时,配置 Web 应用程序不需要停止任何服务。ASP.NET 处理器不对任何文件加锁。编译过的组件被直接复制到当前运行的组件文件的上面。ASP.NET 处理器仅使用内存中的组件或页面版本来满足当前的需求,然后将新组件或页面重载到内存以满足新的要求。

1.4.4 Web 设置和 web.config

Microsoft .NET 应用的配置是通过 web.config 来实现的,web.config 与剩余的 Web 应用程序文件存放在同一目录下,是一种基于 XML 的文本文件。ASP.NET 配置方法有许多优点,有助于有效和快速地安装和配置 ASP.NET 应用。web.config 文件是一个 XML 文档,它是可以直接阅读的,也可以进行程序式的修改。而且,因为 web.config 可以通过文本编辑器修改,因此无须任何详细的脚本或直接访问服务器,站点配置的变化就可以很快地执行。

层次配置结构

web.config 文件与剩余的 Web 应用文件存放在同一目录下,控制该目录及其子目录下的文件的配置设置。由于一个应用程序中可以有多个 web.config 文件,所以可以分别独立配置应用程序的不同部分。

可直接阅读性

由于 web.config 文件是一个基于 XML 的文本文件, 因此它是可以直接阅读的。这就使得无论是对创建系统的开发者还是对维护系统的管理者来说, 修改配置设置都很简单, 可以使用简单的文本编辑器或 XML 编辑器来修改该文件。ASP.NET 配置系统只是对一个 Microsoft .NET 应用程序进行了配置。系统管理员可以使用 xcopy 或 ftp 随同 Web 表单和组件一起, 将 web.config 复制到 Web 服务器上。

自动探测变化

当 web.config 文件被修改时, 系统会直接探测到变化。Web 应用程序无须重启。

1.4.5 输出缓存、数据缓存和碎片缓存

Web 应用程序中最耗时和最高代价的任务之一是数据库访问。如果应用程序中涉及了大量的用户通信量、网络等待以及数据库服务器处理数据请求的能力等问题, 数据库访问可能很快就會成为应用程序的瓶颈。Microsoft .NET 框架给表单缓存中的这类问题提供了解决方案。高速缓存是将被访问数据放入内存以便实现更快的连续检索的一种处理方法。Microsoft .NET 框架支持三种类型的缓存: 输出缓存、数据缓存和碎片缓存。

输出缓存

输出缓存 (output caching) 也称为页面缓存, 提供在指定持续时间内将页面输出到 ASP.NET 高速缓存器的方法。如果在一定时间内用户请求缓存页面, 缓存的内容就会发送给用户。这种方法会使页面性能难以置信地增长, 这些页面中包括很少改变的数据。

数据缓存

数据缓存 (data caching) 提供将特定数据查询的响应放入 ASP.NET 高速缓存器的方法, 从而直接从高速缓存中供应连续相同的查询。这种对数据缓存更明确的控制赋予开发者更多增强应用性能的能力。数据缓存在预定义的间隔或者数据库中的数据发生变化时都会自动到期。

输出和数据缓存很容易使用和执行, 而且都会对 ASP.NET 应用程序产生巨大的性能增长, 特别是当服务器通信量增长时更是这样。

碎片缓存

碎片缓存 (fragment caching) 与输出缓存类似, 但它只具有缓存部分页面的能力, 而不能缓存整个页面。这意味着能缓存包括一些动态数据的页面。碎片缓存的一个很好的例子是内置私有特性的页面。因为页面被私有化, 它们将依赖于用户而产生经常的变化。因此, 输出缓存不能提供解决途径。碎片缓存允许将不具私有性的页面部分放入高速缓存, 这有助于提高 Web 应用程序的性能。