PROGRAMMER TO PROGRAMMER™

Beginning ASP.NET 2.0 and Databases

ASP.NET 2.0 数据库入门经典

(特别版)

(美) John Kauffman 著
Bradley Millington
高 猛 译

wrox™

清华大学出版社

ASP.NET 2.0
数据库入门经典
(特别版)

(美) John Kauffman 著
Bradley Millington
高 猛 译

清华大学出版社
北 京

John Kauffman，Bradley Millington
Beginning ASP.NET 2.0 and Databases
EISBN：0-471-78134-7

北京市版权局著作权合同登记号 图字：01-2006-6177

图书在版编目(CIP)数据

ASP.NET 2.0 数据库入门经典(特别版)/(美) 考夫曼(Kauffman, J.)，(美) 米林顿(Millington, B.) 著；高猛 译. —北京：清华大学出版社，2007.8
书名原文：Beginning ASP.NET 2.0 and Databases
ISBN 978-7-302-15483-9
Ⅰ.A…　Ⅱ.①考…　②米…　③高…　Ⅲ.主页制作—程序设计　Ⅳ.TP393.092

中国版本图书馆 CIP 数据核字(2007)第 090293 号

责任编辑：王　军　梁卫红
装帧设计：孔祥丰
责任校对：成凤进
责任印制：孟凡玉
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：北京市世界知识印刷厂
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**：30.75　**字　数**：748 千字
版　　次：2007 年 8 月第 1 版　　**印　　次**：2007 年 8 月第1 次印刷
印　　数：1～4000
定　　价：58.00 元

前　言

ASP.NET 2.0 小组完成了开发 2.0 版本的艰巨工作，其意义在于以下两个方面：首先，它使得在 ASP 中使用数据比以往任何一个版本都要容易；其次，它使得网站设计人员可以执行比以往任何时候都复杂的、与数据相关的任务。ASP.NET 2.0 的两个成功之处就是更快速的开发及其扩展能力，这也使得本书值得一读。我有幸向 ASP 团体传递这些好消息。

本书是其前身 *Beginning ASP.NET 2.0 Databases Beta Edition* 的主要更新。因为我们很幸运地使用在 2005 年发布的 Beta 版本作为这个新的最终发布版本的起点，所以能够在这本最终版本书籍中添加重要的新内容。甚至在 Microsoft 于 2005 年的秋天发布 ASP.NET 2.0 和相关的工具集后，Microsoft 的 Web Platform and Tools 团队仍然在继续为 ASP.NET 开发人员发布工具，并且我们在本书中已经包括了许多这些工具。我们还使用第二组技术编辑器来重新测试了代码，以保证书中代码的正确性。

本书读者对象

本书的读者对象与以前的版本保持一致。假如读者对使用无数据的 ASP 有基本的了解，那么可以从每一章和每一个练习获得详尽的解释。阅读本书的一般前提条件是读者应该阅读过由本书作者团队(包括 John Kauffman)编写的 *Beginning ASP.NET 2.0* 一书(具有 VB 代码示例的版本的 ISBN 为 0-7645-8850-8；或者是具有 C#代码示例的版本，其 ISBN 为 0-470-04258-3)。或者，读者应该熟悉 ASP.NET 1.x 版或者 ASP 3.0 版。

读者还应该对数据库有所了解。本书没有深入研究设计或管理数据库的细节。

如何使用本书

本书的讲解遵循由简单主题到复杂主题的原则。一般来说，我建议您从头开始阅读本书，然后按照自己的方式读完本书，并且认真完成每一道练习题。但是，有几个例外。

第 2~4 章讲述了 Access 的连接(第 2 章)、SQL Server 的连接(第 3 章)和其他数据源的连接(第 4 章)。如果确定将只使用一种 Microsoft SQL Server 版本(7.0、2000、2005、SQL Server 2005 Express Edition 或 MSDE)，那么第 3 章会介绍相关技术，并且可以跳过第 2 章。如同第 2 章所解释的那样，虽然 Access 适合于学习 ASP.NET 2.0 和数据库技术，但不推荐将其用于产品 Web 服务器配置。因此，即使现在正在使用 Access，通过阅读第 3 章并使用免费的 SQL Server 2005 Express Edition，使用 SQL Server 学习 ASP.NET 2.0 和数据库也会非常容易上手。可以根据需要阅读介绍使用 ASP.NET 2.0 和其他数据源的第 4 章。

约定

为了帮助您最有效地使用本书并跟踪发生的情况，在全书中使用了一些约定。

试一试

“试一试”是应该完成的练习，几乎每一章中都包含这样的练习。

(1) 它们通常由一组步骤组成。

(2) 每个步骤有一个编号。

(3) 按顺序执行步骤。

示例说明

在大多数“试一试”后面，都会详细介绍输入的代码。

源代码

当处理本书中的示例时，可以选择手动输入所有的代码，也可以使用与本书配套的源代码文件。本书中使用的所有源代码都可以从 http://www.wrox.com 或 http://www.tupwk.com.cn/downpage 上下载。在这个站点上，只要找到本书的书名(通过使用 Search 框或者使用书名列表)，并且在本书详细页面上单击 Download Code 链接即可获得本书的所有源代码。

提示：

因为很多书都有相似的书名，所以通过搜索 ISBN 可以很容易地找到本书，本书的 ISBN 是 0-471-78134-7。

一旦下载了代码，只要用常用的压缩工具将其解压缩即可。或者，可以到 Wrox 专门的代码下载页面 http://www.wrox.com/dynamic/books/download.aspx 上查看本书所用的代码以及其他 Wrox 书籍的代码。

勘误表

我们尽全力保证文本或代码中没有一个错误。但是，凡事不可能完美，错误在所难免。如果您在我们的书中发现了错误，比如拼写错误或者代码错误，我们将非常欢迎您的反馈。通过发送勘误表，您也许可以节省其他读者的时间，并且可以帮助我们提供更加高质量的信息。

为了找到本书的勘误表页面，请浏览 http://www.wrox.com 并使用 Search 框或者书名列表找到本书的书名。然后，在本书的详细页面上单击 Book Errata 链接。在这个页面上，可以查看所有在本书中出现的错误列表。也可以从 www.wrox.com /misc-pages/booklist.shtml 上获得一个完整的书籍列表，包括至每本书的勘误表的链接。

如果您没有在 Book Errata 页面上发现“您的”错误，请浏览 www.wrox.com /contact/techsupport.shtml 并填写其中的表格，将您发现的错误发送给我们。我们将检查收到的信息，如果属实，将添加消息至本书的勘误表页面中并在本书后续版本中解决这个问题。

p2p.wrox.com

如果要与作者和同行进行讨论，可以加入 p2p.wrox.com 上的 P2P 论坛。这个论坛是基于 Web 的系统，您可以提交有关 Wrox 书籍和相关技术的消息并与其他读者和技术用户进行交流。论坛还提供订阅服务，当在论坛中发表新文章时，论坛将把您选择的感兴趣的主题通过电子邮件发送给您。Wrox 作者、编辑、其他业界专家以及您的读者同伴都将会聚集在这些论坛中。

在 http://p2p.wrox.com 上，您将发现有些论坛不仅可以对您阅读本书给予帮助，还可以有助于您开发应用程序。如果要加入论坛，请执行如下步骤：

(1) 进入 p2p.wrox.com，单击 Register 链接，阅读使用条款，并单击 Agree 按钮。

(2) 完成加入所需的信息和任何您想要论坛提供的信息，然后单击 Submit 按钮。

(3) 您将收到一封电子邮件，介绍如何验证您的账户并完成加入过程。

注意：

可以阅读论坛中的信息而不加入 P2P，但是只有加入论坛才能发表文章。

一旦您加入论坛，就可以发表自己的新文章并且可以回复其他用户发表的文章。可以在任何时候阅读该 Web 站点上的信息。如果您想将一个特定论坛上的新文章通过电子邮件发送给您，可以通过论坛列表中的论坛名称并单击 Forum 图标中的 Subscribe 发送。

如果需要更多有关如何使用 Wrox P2P 的信息，一定要阅读 P2P FAQ 中有关论坛软件如何工作的问题的回答，以及对很多有关 P2P 和 Wrox 书籍常见问题的回答。如果要阅读 FAQ，可以在任何 P2P 页面上单击 FAQ 链接。

目　录

第 1 章

ASP.NET 2.0 和 ADO.NET 简介

ASP.NET 2.0 和.NET Framework 2.0 这个新推出的版本为初级程序人员带来了快捷和强大的开发能力，这使我急切地想把它介绍给大家。我相信您将会经常惊诧于自己花费很少的努力却能得到高质量的回报。我想，就像我编写第一个示例时一样，您也会经常发出这样的感叹，“喔，事情就应该是这样”。您的经理和客户们也将对您创建的访问快速且内容精确的 Web 站点留下深刻的印象。

本章将会介绍如何在 ASP.NET 2.0 Web 页上使用数据，如何使用本书提供的软件和数据来构建一台能够使用 ASP.NET 2.0 的机器，另外还演示了 ASP.NET 2.0 的一些强大功能，这些功能还会在后面详细介绍。有些内容是回顾一些预备知识(ASP.NET 的基本知识并熟悉基本的数据库操作)。以下就是本章要讲述的主要内容：

- 概述相关技术，包括.NET Framework 2.0、ASP.NET 2.0 和 ADO.NET。简要回顾旧有的版本(1.x)如何使用数据，以及概述一些关键术语
- 讨论在 ASP.NET 2.0 页上使用数据所需的组件
- 指导完成本书所需的安装任务
- 演示几个应用了数据的 ASP.NET 2.0 页面

和以前版本一样，各章后面都包括了一个如下列表：常见错误、本章小结、用于检查所学知识的练习题。

1.1 .NET 技术概述

大约有一百万名开发人员使用.NET Framework 的第一个版本开发网站。所以在 2003 年的夏天，当从 Microsoft 传出将有新版本发布的消息时，许多人都满怀期待。果然，Microsoft 不负众望，这个新的版本将使得创建 ASP.NET 页所需的代码行数减少 70%。像这样大规模地提高生产效率在程序设计中是不多见的。

当 ASP.NET 2.0 的代码样例在 2003 年秋的 Microsoft 专业开发人员大会(Microsoft Professional Developer’s Conference)上演示时，结果比预计的还要好。以前使用 ASP.NET 的第一个版本的程序员需要花费几个小时才能做出的 Web 页，现在使用 ASP.NET 2.0 只需要几分钟就可以完成。简单来说，在.NET Framework 2.0 最终版本发布之后，任何继续使用第一个版本创建ASP.NET页面的程序人员都将花费大量的额外时间才能达到相同的结果。

与其他任何领域不同的是，ASP.NET 2.0 所提供的优势是能够方便地将数据集成到页面中。程序员不再需要知道详细的连接、命令以及数据阅读器和数据适配器对象，就能执行普通的数据任务。ASP.NET 2.0 使得基本数据的使用简单易学，并且使初学者也可以进行更复杂的数据应用。

1.1.1 .NET Framework 简介

Microsoft 开发出了作为基本原则的.NET 和一系列技术，用于在 Internet 中让计算机共同工作。总的目标就是让信息和进程在大范围的系统和设备之间顺畅地交流。.NET 不是一种语言，也不是特别的产品。更确切地说，它是一套标准和指导原则，并已经应用于自 2002 年以来 Microsoft 发布的所有产品中。

.NET 包含了一种使用开放标准的 XML 格式交换信息的标准化格式。可扩展标记语言(Extensible Markup Language，XML)不需要请求者具备任何有关数据存储如何保存信息的专门知识——数据都以自描述的 XML 格式取出。同样地，目前几乎所有的数据存储都可以用 XML 来提供信息，这对于所有.NET 数据客户都具有吸引力。.NET 支持信息交换的其他标准(例如，t-SQL、加密的强流以及不支持 XML 的 HTML)。不过，一开始我们会更多地使用 XML。

.NET 支持软件的 Web Services 标准，可请求在使用开放平台标准的简单对象访问协议(Simple Object Access Protocol，SOAP)和 XML 语言的远程软件上运行代码。.NET 网站可以从另外一个网站上找到该网站所提供的服务，并使用这些服务。这样可以使得网站从其他的网站上获得 HTML、计算结果或者数据集。

作为.NET 一个关键的部分，Microsoft 发布了一套运行时编程工具和应用编程接口(API)，称为.NET Framework，让开发团队能够创建客户端-服务器应用程序、.NET 客户端和服务器应用程序和 XML Web Services。

.NET Framework 由公共语言运行库(Common Language Runtime，CLR)和一套统一的类库组成。CLR 为运行的应用程序提供了一个完全托管的执行环境，其中包括几个服务，例如，程序集装载和卸载、进程和内存管理、安全性实施以及即时编译等。CLR 名称的意思就是指能够用多种语言编写应用程序，并且将源代码编译成 CLR 能够读懂并运行的中间语言，而无需考虑原来所使用的语言。这种"语言独立性"就是 CLR 的关键特性(也是 ASP.NET 的特性)，它允许开发人员使用自己喜欢的语言工作，比如 C#、VB.NET 或者 COBOL，都能够获得.NET Framework 的所有特性。

.NET Framework 还包括了一套类库，这是程序员可以使用的大型代码集。这套类库提供了每一个应用程序所需的常用功能。可以使用.NET Framework 支持的任何语言来访问这些类库。这些类库提供如下的服务(以及相应的命名空间)：

- **系统类型(System)** 包括大多数简单数据类型的定义、类型之间的转换函数、一些数学函数、数组以及处理异常的方法。
- **输入/输出(System.IO)** 包括读取和写入数据流(例如，Pocket PC 等外部设备的输出)的类，以及读取和写入驱动器上文件的类。

- **数据访问(System.Data)**　包括与外部数据(一般是数据库)交互的类。这个命名空间包括 ADO.NET 的大多数内容，由 ASP.NET 2.0 数据控件使用，因此本书将对其重点介绍。
- **安全性(System.Security)**　包括实现安全性基础的类，特别是权限系统。
- **数据结构(System.Collections)**　包括组织和维护.NET 中数据集的类，这些数据集包括列表、字典和散列表。(注意：System.Data 命名空间用于与外部数据集通信)
- **配置(System.Configuration)**　包括建立配置设置和使用该设置处理错误的类。
- **联网(System.Net)**　包括使用联网协议(例如，IP 和套接字)的类。
- **反射(System.Reflection)**　包括允许程序查看自身以标识自己的特征的类，例如，特性数据类型和代码的组织。
- **全球化(System.Globalization)**　包括保存文化特有设置(例如，日期和货币表示法)的类。
- **绘画和制图(System.Drawing)**　包括可以与图形用户界面交互的类。这个命名空间包括许多特定绘制类型的类，例如，Pen、Brush 和 Imaging。
- **跟踪和诊断(System.Diagnostics)**　包括分析问题的类，包括系统进程、进程计数器和事件日志的分析。
- **窗口(客户端)应用程序模型(System.Windows.Forms)**　包括允许实现 Windows 用户界面的类。
- **Web 应用程序模型(System.Web)**　包括为产生网页提供编程模型的类。网页代码运行在服务器上，并且提交 HTML 给客户端的浏览器。这个类包括 12 个以上的子类，这些子类关注各种类型的浏览器和服务器服务。本书将重点介绍这些内容，因为 System.Web 是 ASP.NET 的基本内容。

注意，.NET Framework 包含了两个应用程序编程模型，一个用于客户端应用程序(System.Windows.Forms)，另一个用于基于 Web 的应用程序(System.Web)。本书着重讲解后一个模型。.NET Framework 中的 System.Web 命名空间是.NET Framework 的一部分，它提供了 ASP.NET 功能。换句话说，ASP.NET 就是构建应用程序的所有.NET Framework 的一部分。

1.1.2　ASP.NET 简介

ASP.NET 是一种用于创建基于 Web 的应用程序的编程模型。从本质上来说，运行时和.NET Framework 类库集可以用于创建动态 Web 页。ASP.NET 需要在 Web 服务器的上下文中运行，例如，Microsoft Internet Information Server(Internet 信息服务器，IIS)，并且在服务器上处理编程指令以服务浏览器请求。与直接由 Web 服务器服务的静态 HTML 文件不同的是，ASP.NET 页构建于服务器上以产生动态结果。页面的最后呈现也许是由许多不同的指令或数据源构造的。

ASP.NET 页以.aspx 扩展名在 Web 服务器上存储。ASP.NET 页由程序员将文本、标记(例如，HTML)以及 ASP.NET 特定服务器标记和脚本组合在一起。可以将存储后的 ASP.NET 页看成是一组描述如何创建 HTML 页的指令。当请求浏览该页面时，处理服务器端逻辑以创建客户端浏览器可以显示的 HTML 页。因为呈现后的输出是 HTML 标记(没有专门的代

码)，所以任何浏览器都能够读懂。所有的动态处理过程都发生在 Web 服务器。ASP.NET 特定服务器标记非常强大，例如，它可以对用户的动作做出反应，连接至数据存储以及自动创建非常复杂的 HTML 结构。

正像前面提到的，ASP.NET 是.NET Framework 的一部分，所以 ASP.NET 页面可以利用这个架构提供的所有服务，包括联网、数据访问、安全性以及更多其他服务。因为 ASP.NET 可以使用所有这些架构服务，所以相比以前，能够创建更加丰富的 Web 应用程序。在 ASP.NET 中，不必要花太多的时间来构建所有应用程序所需的构建块，可以将大多数时间用在应用程序独有的特殊逻辑上。

ASP.NET 还引入了一些 Web 编程的新技术，可以就典型的动态服务器页(Active Server Pages，ASP)来极大地改善开发模型：

- **语言独立性** 因为 ASP.NET 是.NET Framework 的一部分，所以可以使用选择的语言来构建 ASP.NET 应用程序，例如，Visual C#、Visual Basic .NET 或 J#。另一方面，典型的 ASP(版本 1，2 和 3)则仅限于 JScript 或者 VBScript 页面。
- **编译而不是解释** 与典型的 ASP 在每一次页面请求时都解释编程指令不同，ASP.NET 在服务器端动态地将页面编译成可以运行得非常快的本机编程指令。可以很明显地看到，典型的 ASP 页面的运行比相同的 ASP.NET 页面要慢一个数量级。
- **事件驱动编程模型** 在典型的 ASP 中，页面总是以自顶向下的线性方式执行，并且 HTML 标记常常与编程指令混合在一起。任何一个有一定典型 ASP 经验的人都知道这样会使得页面难以阅读，甚至更加难以维护。而 ASP.NET 引入了事件驱动模型，这个模型允许将代码与标记内容分离，将代码并入处理专门任务的有意义的单元中，例如，响应客户端的按钮单击动作。类 VB 的事件模型极大地提高了页面的可读性和可维护性。
- **服务器控件** 典型的 ASP 需要动态地将 HTML 片断代码接合在一起呈现，这样做的结果就是在应用程序中一遍又一遍地编写相同的代码(需要多少次才能从数据库查询中构建一张数据表)。ASP.NET 带给 Web 编程的一个最大益处就是能够将执行公共行为的代码封装进服务器控件，这样可以在应用程序中很方便地重复使用。就像 HTML 标记一样，服务器控件以声明的形式创建，但是表现为一个位于服务器端的可编程对象，它可以与代码进行交互并输出定制的动态 HTML 呈现。ASP.NET 包含了大约 80 多个服务器控件，这些控件封装了从标准表单元素到复杂控件(如网格和菜单)的所有行为。
- **控件设计时间的改善(当使用 Visual Web Developer 时)** 开发人员通过使用设计时间界面可以减少花费在开发复杂页面上的时间，这些界面包括敏捷任务面板、标签级导航栏和可以设置控件属性的向导。

如果您使用过 ASP，就会惊讶于使用 ASP.NET 2.0 所带来的生产效率的提高。如果从来没有使用过 ASP，可以听别人谈谈 ASP 的历史。

1.1.3 ASP.NET 2.0 简介

ASP.NET 的第一个版本(1.0 和 1.1)在 2001 年至 2005 年间迅速风靡了 Microsoft 的开发人员阵营。程序员很快便感觉到他们通过使用强大和灵活的.NET Framework 可以大大减少

编程时间，而且 CIO 们也看到当程序员花在解决客户代码的疑难问题上的时间减少时，他们就可以将更多的精力投入到更高级的 IT 结构的改善上。ASP.NET 确实是一个里程碑式的版本，它简化了 Web 开发人员的工作。

而且，就在 1.0 版本发布之前，Microsoft 的 ASP.NET 小组就已经开始着手 ASP.NET 2.0 的开发工作。同时，他们设立了下面这些雄心勃勃的设计目标：

- 使创建一个典型的 Web 应用程序所需的代码行数减少 70%。
- 提供一套可扩展的应用程序服务，用来为通用应用程序任务提供构建块，例如，成员、角色、个性化以及导航等。
- 创建一系列基于任务的服务器控件，这些控件可以调节上述服务，交付完全的、可定制的用户界面(UI)，以最小的代码量来展示这些服务。
- 改进页面处理性能，从而增加整体的 Web 服务器吞吐量。
- 提供管理功能，以便加强 ASP.NET 服务器的部署、管理和运行。
- 改善宿主公司所用的工具，以便可以支持多站点并能够将开发人员的项目迁移至公共部署环境。
- 让 ASP.NET 的几乎所有特性都能够方便地扩展或者用定制的高级任务的实现替换。

在这里，我们谈一谈第一个目标，也就是使编写一个动态 Web 应用程序所需的代码量减少 70%。这有可能吗？ASP.NET 小组已经仔细考虑过各种当前以定制代码执行的通用任务，并且专门制定了将这些任务封装进构建块(提供服务的服务器控件或类)的方式，这样就能够自动地完成这些任务。例如，大多数 Web 应用程序都需要安全性、导航或者个性化服务来为用户提供定制的体验。在 ASP.NET 2.0 中，这些任务是通过一系列可配置的应用程序服务，以及与这些应用程序服务进行通信的服务器控件来实现的。通过提供这些类库，Microsoft 极大地减少程序员实现这些通用任务所需的代码量。

但是，在所有这些通用任务当中，有一个任务是绝对独立于其他应用程序的，这就是数据访问服务。数据访问是驱动大多数动态 Web 应用程序的公用线程，所以毫不奇怪，ASP.NET 小组为了减少代码量和在 ASP.NET 2.0 的应用程序中执行数据访问所需的概念，制订了一些大胆的目标。具体包括如下各项：

- 在 ASP.NET 中可以通过声明的(无代码)方式来定义一个数据源。
- 可以通过声明的(无代码)方式用控件显示数据，无需在页面的执行生存期中的特定时间进行显式的数据绑定。
- 可以通过声明的(无代码)方式执行通用数据任务，例如，排序、分页、过滤、更新、插入以及删除数据。
- 可以使用多种 UI 控件来显示数据，包括灵活的 GridView 和 DetailsView 控件，该控件可以显示和操作数据。
- 为创建定制的数据源，启用可扩展的模型来支持新的数据类型。
- 为创建定制的控件，启用可扩展的模型来以新的方式显示数据。

前面这些目标的结果是 ASP.NET 2.0 具有一些程序员可以使用的特定的服务器控件，用于在页面上添加数据交互。这些专门的数据控件分为两组：数据源控件和数据绑定控件。数据源控件创建与数据存储(数据库或其他存储库)的连接。数据绑定控件则从数据源控件获取信息，并在页面上创建呈现。这种简单的双控件模式可用于多种情况。