Mc Graw Hill Education

ADO.NET技术参考大全
与Visual Studio .NET 2003和
Windows .NET Framework 1.1同步

技术参考大全

ADO.NET : The Complete Reference

ADO.NET

(美) Michael Otey
Denielle Otey 著
史创明 崔金铃 译

理解.NET Framework和ADO.NET体系结构

最大程度地构建平台的互操作性和实现可伸缩的数据访问

获得所有ADO.NET对象的现成的代码示例

清华大学出版社

ADO.NET 技术参考大全

(美) Michael Otey, Denielle Otey 著

史创明 崔金铃 译

清华大学出版社

北 京

Michael Otey, Denielle Otey
ADO.NET: The Complete Reference
EISBN：0-07-222898-9

北京市版权局著作权合同登记号　图字：01-2003-3807

本书封面贴有 McGraw-Hill 公司防伪标签，无标签者不得销售。

图书在版编目(CIP)数据

ADO.NET 技术参考大全/(美)迈克尔·奥蒂，(美)笛莱尼·奥蒂著；史创明，崔金铃译.
—北京：清华大学出版社，2003
书名原文：ADO.NET: The Complete Reference
ISBN 7-302-07296-5
Ⅰ.A…　Ⅱ.①迈…②笛…③史…④崔…　Ⅲ. 数据库—接口—程序设计　Ⅳ.TP311.11
中国版本图书馆 CIP 数据核字(2003)第 085150 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：曹康
文稿编辑：李阳
封面设计：康博
版式设计：康博
印 刷 者：北京市清华园胶印厂
装 订 者：三河市新茂装订有限责任公司
发 行 者：新华书店总店北京发行所
开　　本：185×260　**印张**：40　**字数**：1023 千字
版　　次：2003 年 10 月第 1 版　2003 年 10 月第 1 次印刷
书　　号：ISBN 7-302-07296-5/TP·5296
印　　数：1～3000
定　　价：75.00 元

原　　序

无论您是否熟悉 ADO.NET 或具有相关工作经验，本书都将为您提供关于数据访问的重要内容。ADO.NET 的初学者可以通过基础概念部分的学习，为构建更复杂的数据访问打下坚实的基础，而高级专题和详尽的 ADO.NET 类参考文档对于经验丰富的开发者来说更是无价之宝。

本书共分为 7 个部分，描述了 ADO.NET 的核心特征和数据对象。首先介绍了 ADO.NET 的基础内容，描述了数据访问技术的发展史，以及 ADO.NET 如何解决以前的数据访问方法中存在的一些难题。随后的 4 个部分深入描述了主要的 ADO.NET 对象，以及应如何运用这些对象。这些部分通过在 Visual Studio .NET 开发环境中使用可视化的 ADO.NET 对象，指导您进行快速、简单的数据操作；同时也介绍了如何在程序代码中使用 ADO.NET 类，使应用程序更灵活多变，能够应付那些需要更多高级数据访问技术的复杂情况。第Ⅵ部分解释了 XML 和 ADO 数据集成，附录则提供了 ADO.NET 中命名空间(namespace)的完整、全面的参考资料。下面简要介绍本书各部分的内容。

第Ⅰ部分——ADO.NET 基本概念　本部分介绍了数据访问技术的历史概况和 ADO.NET 的大致情况，然后进一步讨论了对.NET Framework 的理解并概述了完整的 ADO.NET 体系结构。第 1 章回顾了 ADO.NET 之前的数据访问技术，如 DAO、RDO、ODBC 和 ADO 等知识。我们应该了解 ADO.NET 是如何解决过去的数据访问方法中的缺陷的，这将有助于更高效地使用 ADO.NET。在编写一个稳健的应用程序时，了解系统的需求是十分必要的，第 2 章介绍了与之相关的内容。还阐述了.NET 体系结构与.NET Framework 类库方面的知识，以及用于构建 ADO.NET 的托管代码平台的相关知识。第 3 章详细描述了 ADO.NET 体系结构，并扼要介绍了 ADO.NET 中使用的类。该章还介绍了.NET Data Provider 提供的一些 ADO.NET 类，这些类允许您访问正在使用的、特定的后端数据资源。这一章里还列举了用于处理数据并且适用于任意数据库的通用 ADO.NET 类。

第Ⅱ部分—— ADO.NET Connection 对象　这一部分主要描述了 ADO.NET 中的 Connection 对象，每一个.NET Data Provider 都包含该对象。Connection 对象是打开数据资源的主要链接，所有.NET Data Provider 都包含一个 Connection 对象，用来优化到特定数据库的链接。本部分的每一章都使用一个不同的.NET Data Provider 的 Connection 对象，以展示不同.NET Data Provider 使用的连接技术。这一部分探究了打开可信连接、使用一个 UDL 文件连接数据库、无 DSN 连接、以及启用连接池等技术。这些连接数据源的方法可以适用于任何 Connection 对象。例如，第 4 章介绍了如何使用 SqlConnection 对象打开信任连接，但也可以使用 OracleConnection 对象、OleDbConnection 对象或 OdbcConnection 对象打开信任连接。我们分别列出了一些.NET Data Provider 的对象，通过完整的例子来展示连接到不同数据源时存在的一些语法差异。

第Ⅲ部分——ADO.NET Command 对象　像 ADO.NET 的 Connection 对象一样，每一个.NET Data Provider 都包含自己的 Command 对象。应用程序经常进行的一些数据库操作，包括执行 SQL 语句和执行存储过程。这一部分中的章节不仅演示了动态 SQL 查询，而且演示了如何使用.NET Data Provider 的 Command 对象执行参数化查询。此外还阐述了一些包含返回值和输出参数且使用 Command 对象的存储过程，并介绍了.NET Data Provider 的 Parameter 类和数据源

特定的参数标记符号的用法。这一部分中还描述了与事务相关的知识——事务允许您把多项操作集合在一起，作为一个单元执行。您将了解如何综合利用.NET Data Provider 的 Transaction 类和 Command 类来确保数据库的完整性。这一部分中描述的 Command 类、Parameter 类和 Transaction 类方法适用于所有.NET Data Provider 的 Command、Parameter 或 Transaction 类。

第Ⅳ部分——ADO.NET DataReader 对象　本部分介绍了 ADO.NET 的 DataReader 对象。DataReader 对象可快速检索只向前(forward-only)的结果集。此处再次强调，每个.NET Data Provider 都要提供自己的 DataReader 和方法，并且在每章中描述的方法适用于其他.NET Data Provider 的 DataReader 对象。这一部分还列举了返回只向前的结果集、多结果集和层次化结果集的例子。当需要详细的表模式信息时，DataReader 可用来检索仅包含模式(schema-only)的信息。另外，这一部分也讲述了如何填充 DataSet 和检索 Binary Large Object(BLOB，二进制大型对象)的方法，了解它们是非常有用的。

第Ⅴ部分——ADO.NET DataSet 对象　ADO.NET 的 DataSet 对象是 ADO.NET 应用程序的核心对象。DataSet 类实质上是一个保存在内存中的数据库。这一部分的第 16 章讨论了如何构建 DataSet 以及组成 DataSet 对象元素的通用.NET Framework 类，包括 DataTable、DataColumn、DataRow、DataRelation、DataView 类以及 Unique 和 ForeignKey 约束。后续的章节转而介绍重要的 DataAdapter 类。DataAdapter 是一种机制，决定数据源中哪些数据应当被加载到 DataSet，并决定 DataSet 信息在经过改动后，哪些信息会被回送到数据库中。DataAdapter 与不同的数据源交互，因此每个.NET Data Provider 都需要提供自己的 DataAdapter 类。第 17～21 章介绍了如何使用不同.NET Data Provider 中的 DataAdapter 在不同的数据源中构建 DataSet，而且，这几个章节中使用的方法适用于任何 DataAdapter。另外，该部分还介绍了如何使用 DataAdapter 与 SQL 语句或存储过程向 DataSet 加载数据，同时还介绍了如何使用多个 DataAdapter 将多个数据源中的数据填充到 DataSet。一旦将数据加载到 DataSet 中，您就可以浏览、搜索、分类或选择记录并执行更新操作。第 22～24 章中描述了这些数据的浏览方法和特征。在使用完 DataSet 中的信息后，需要将所做更改写回到数据源。第 25～28 章演示了通过 DataAdapter 类使用 DataSet 中的数据更新数据库的方法。这里再次强调，每一章中使用的、由.NET Data Provider 提供的 DataAdapter 的更新数据库技术适用于任何 DataAdapter 类。这些章节中演示了如何使用自定义更新命令来更新数据源，以及如何使用 CommandBuilder 类自动生成更新命令。该部分随后介绍了 DataForm Wizard 的使用方法，通过 DataForm Wizard 能够快速地创建简单的数据绑定 Windows 窗体。这里还介绍了如何使用参数和存储过程，以及如何选择具有某种更新类型的行(如仅选择已插入的记录)，以便以特定的顺序更新数据库。第 29 章是该部分的最后一章，描述了一些高级更新技巧和方法，主要包括合并 DataSet、使用 AutoIncrement 字段、检测 DataSet 的错误、以及在将 DataSet 行发送回数据库前决定是对其执行 Accepting 操作还是 Rejecting 操作。

第Ⅵ部分——ADO.NET 数据集成　本部分阐述了如何运用 ADO.NET、将数据映射到 XML 以及同时使用 ADO 和 ADO.NET 的情况。第 30 章提供了关于 XML 术语、XML 文档和组成部分的总体介绍，并且讲述了如何在 ADO.NET Framework 中运用 XML。这一章中描述了 XMLReader 类、XML DiffGram、XPath 查询语言，以及 XSLT/L。第 31 章讨论了如何将基于 COM 的 ADO 对象导入到.NET Framework 中。在这里，您将会了解到如何在 ADO.NET 中使用 ADO 记录集，并且学会处理同时使用 ADO 和 ADO.NET 时某些共存的问题。

第Ⅶ部分——附录　本部分包含了附录 A 到附录 F。这些附录为 ADO.NET 各种命名空间提供了详细的参考信息，阐述了各种命名空间中所需的类、委托、枚举和接口，并提供了代码示例。这些命名空间的参考资料主要与 ADO.NET 对象相关，并且经过特别编排，高效、易于读者使用。此外，该部分还为每个类对象构造函数提供了完整的 C#代码示例，可以给您一些提示，从而帮助您快速生成正确的程序代码。

Web 资源　可以在 Osborne/McGraw-Hill 的 Web 站点下载本书提供的所有应用程序代码。您可以在脱机情况下直接执行这些可执行程序和源代码。可以在 www.osborne.com 站点搜索 ADO.NET: The Complete Reference，以下载相关代码。另外，通过剪切并粘贴示例程序，可以帮助您快速开发自己的 ADO.NET 应用程序。

目　　录

第Ⅰ部分　ADO.NET 基本概念

第 II 部分　ADO.NET Connection 对象

第III部分　ADO.NET Command 对象

第Ⅳ部分　ADO.NET DataReader 对象

第Ⅴ部分　ADO.NET 的 DataSet 对象

第Ⅵ部分　ADO.NET 数据集成

第Ⅶ部分　附　　录

第Ⅰ部分　ADO.NET基本概念

第1章　ADO.NET简介

ADO.NET 是 Microsoft 的数据访问技术发展中最新的成果。本章将介绍 Microsoft 的这个新的 ADO.NET 数据访问架构。首先，简要回顾 ADO.NET 之前的历史，以对其产生的原因和背景有所了解。您将目睹 Microsoft 的数据访问策略演变的轨迹，以及每种新技术是如何解决原有技术所面临的问题和技术上的挑战的，这些演变最终导致了 ADO.NET 的产生。您还会看到 ADO.NET 如何比其前任技术更有效地解决了当今的数据访问问题。接下来，我们将潜心研究.NET Framework。这一节中将对.NET Framework 的体系结构进行高度概括，您会清楚地认识到 ADO.NET 技术在.NET Framework 中处于一个什么样的位置。如果想再进一步，可以在第 2 章找到关于.NET Framework 的更详尽的内容。本章结束之际对组成 ADO.NET 的不同.NET 命名空间和类进行了概述。本书附录中提供的参考资料深入讲述了 ADO.NET 类。

1.1　Microsoft 的数据访问技术

Microsoft 首次介入标准数据库访问机制是在 1992 年，其时发布了针对 16 位的 Windows 3.1 的开放数据库互联(Open Database Connectivity，ODBC)技术。ODBC 实质上代替了其前任技术，即嵌入式 SQL，成为主要的数据库访问机制。嵌入式 SQL 灵活性不够，但更糟的是，它是和目标数据库系统紧紧绑定的。使用嵌入式 SQL 基本上不可能创建一个能访问多种数据库平台的应用程序。另外，语法的差异通常也会妨碍源代码的兼容。在向应用程序提供数据库互操作性方面，ODBC 是第一次真正成功的尝试。Microsoft 的数据访问技术自那时以来一直在不断地演变，既解决了应用技术的环境，又发展了数据库应用程序开发技术。Microsoft 以后的数据访问技术 DAO、RDO 和 ODBCDirect 在开发时都考虑了基于 ODBC 的数据访问。直到组件对象模型(Component Object Model，COM)的诞生，Microsoft 才从 ODBC 转向它的后续技术 OLE DB。OLE DB 的一个消费者，活动数据对象(Active Data Object，ADO)成为了新的 OLE DB 技术首选的数据库访问编程模型。虽然 OLE DB 和 ADO 在解决客户机/服务器风格的应用程序的数据访问问题上居功至伟，但对于 n 层的 Web 应用程序却不是最适宜的技术。而且，这些技术在企业范围的部署方面还面临着不少困难。为了克服这些难题，Microsoft 开发了.NET Framework 和 ADO.NET。本节将对 ADO.NET 出现之前的 Microsoft 数据库访问技术进行适当的讲解。

1.1.1　开放数据库互联(ODBC)

ODBC 是 Microsoft 定义的一种数据库访问标准。ODBC 的设计宗旨是为不同平台上的 SQL

数据库提供标准的桌面访问方法。一个 ODBC 应用程序可以用来访问本地 PC 数据库中的数据，但它的真正意图是访问异构平台上的数据库，如 SQL Server、Oracle 或 DB2。虽然 ODBC 最初是一种 Windows 标准，但在其他平台上也实现了 ODBC，包括 UNIX。

ODBC 实质上是一种数据库访问应用程序编程接口(Application Program Interface，API)。ODBC API 是独立于数据库的，基于 X/Open 和 ISO/IEC 开发的 SQL/Call Level Interface (SQL/CLI)规范。ODBC 3.0 标准是最新的 ODBC API 版本，包括 76 个函数，记录在 Microsoft 共分三卷的《ODBC 3.0 Reference and SDK Guide》中。ODBC API 虽然从表面来看是由一组函数调用组成的，但 ODBC 的核心是 SQL。ODBC 函数的主要目的是把 SQL 语句发送到目标数据库，然后处理这些 SQL 语句的结果。显然，服务器必须能够支持 SQL，才能通过 ODBC 访问。

作为一个编程接口，ODBC 要求使用包括在 ODBC API 中的具体函数调用。虽然可以在 VB 这样的语言中使用这些 API，但在 C 语言中用起来要容易得多，因为 API 就是用 C 语言编写的。使用 API 相对来说比较困难，因为需要理解函数调用的必要顺序，以及每个 API 的必要参数及参数的数据类型。

虽然 ODBC 是通过一组标准的函数调用实现的，但它有一个最大的好处就是，它是一种广泛采用的桌面标准。一般来说，最终用户使用 ODBC 时不需要知道或理解这些函数——使用 ODBC 需要的所有代码都内置到了支持 ODBC 的应用程序中，如 Microsoft Access、Word、Excel 或 Visio。支持 ODBC 的桌面应用程序实际上数以百计。

1.1.2　数据访问对象(DAO)

DAO(Data Access Object)是 Microsoft 第一次尝试建立基于 COM 的数据库访问技术。自从 DAO 1 随 Visual Basic 3 一起发布以来，DAO 就一直是最初的 Visual Basic 的一部分。DAO 是 Visual Basic 默认的数据访问方法，它的主要目的是通过 Jet 引擎为 Visual Basic 应用程序提供对本地 Access 数据库的访问。然而，Microsoft 当初设计 DAO 是让它能访问其他几个不同的数据库(如 dBase、Paradox)，甚至是 ODBC 数据库(如 Oracle 和 SQL Server)。这种灵活性是 DAO 的一个强项。DAO 和 Jet 引擎能够使用相同的代码访问一系列不同的数据库。作为一种本地查询处理程序，Microsoft Jet Database Engine 赋予了 DAO 在大多数其他客户机/服务器架构的数据访问机制中所没有的功能。例如，DAO 和 Jet 能够对来自不同数据库的数据进行异构连接操作。

虽然 DAO 很灵活，但这种灵活性是有代价的。DAO 有它自己的本地查询处理程序，这给 DAO 应用程序带来了很大的开销。还有，DAO 主要不用于处理基于 ODBC 的数据源。DAO 实际上是一种一层的桌面数据库应用程序，数据库和程序位于同一个平台。实质上，DAO 以处理本地数据源的方式来处理 ODBC 数据源。这经常导致数据访问代码十分低效，因为要频繁地执行像打开和关闭这样开销很大的操作。对于本地基于文件的数据库而言这没有什么问题，但会严重降低客户机/服务器架构的基于 ODBC 的数据库应用程序的性能。

1.1.3　远程数据对象(RDO)

RDO(Remote Data Object)是继 DAO 之后开发的技术，它综合了 DAO 的容易编程特征和 ODBC API 的高性能。DAO 是在 Microsoft Jet 引擎之上的一个对象层，而 RDO 是封装了 ODBC API 的一个对象层。Microsoft 没有打算用 RDO 来访问像 Access 这样的本地数据库，而是明确

地针对客户机/服务器风格的网络数据库访问。没有了开销极大的Jet引擎，再加上RDO和ODBC的紧密联系，这使得RDO在访问兼容ODBC的数据库(如SQL Server)时，在性能上比DAO更胜一筹。RDO提供对兼容ODBC数据库的编程访问，不会发生本地查询处理程序(如Microsoft Jet引擎)的开销。实际上RDO能提供对ODBC API全部功能的访问，但基于COM这一点让它用起来非常轻松。利用RDO，没有必要再在BAS或CLS文件中手工声明所有的函数。相反，RDO中COM的实现使其能无缝地同Visual Basic和其他COM开发环境集成，其方法只需在IDE中添加引用即可。这些引用可以让IDE提供所有RDO对象方法和属性的智能感知提示。虽然RDO很好地解决了DAO的ODBC数据库访问中的问题，但它并没有存在多久。RDO是基于ODBC的，不久就让位于基于OLE DB的数据访问技术ADO，ADO提供了更加简单的编程模式和对异构数据源的访问。

1.1.4　ODBCDirect

像以往一样，Microsoft为解决DAO遇到的数据库访问问题提供了不止一种技术。虽然RDO是第一个开发的技术，但不久又开发了ODBCDirect。ODBCDirect是介于DAO和RDO之间的一个混合技术。ODBCDirect使用的对象模型类似于DAO。但是和RDO一样，ODBCDirect也是位于ODBC之上的一个对象层，而且不使用Microsoft Jet Database Engine。事实上，和RDO完全一样，ODBCDirect也只能连接到ODBC数据源，不能连接到本地的Access数据库。ODBCDirect也支持RDO提供的大多数高级ODBC功能。其他相同之处还有，ODBCDirect支持使用ODBC游标，和预编译的SQL语句、参数化查询以及存储过程返回参数。虽然ODBCDirect似乎具备了和RDO差不多完全相同的功能，但它有一个非常重要的优势，即它是DAO 3.6的一部分，因此在成本较低的Visual Basic Professional Edition中提供了该技术。而RDO只在成本要高得多的VB Enterprise Edition中才有。但ODBCDirect也和RDO一样短命，不久就被OLE DB和ADO代替了。

1.1.5　OLE DB

Microsoft引入了OLE DB作为ODBC的后继技术。考虑到ODBC的广泛成功，OLE DB面临着要更胜于其前任的艰巨任务。ODBC API得到了几乎是普遍的认可和接受。除了为几千个自定义数据库应用程序提供数据库支持外，大多数用收缩膜包装的(shrink-wrapped)桌面应用程序，如Microsoft Office，都支持ODBC API。实际上所有主要的数据库系统都有ODBC驱动程序。然而，ODBC主要的设计目的是处理关系数据，它从来都没打算用于处理非关系数据源。

OLE DB是作为一种普遍的数据访问技术创建的。像ODBC一样，OLE DB提供对关系数据的访问，但OLE DB扩展了ODBC提供的功能。OLE DB被设计为所有类型数据的一个标准接口。除了关系数据库访问之外，OLE DB还提供对多种数据源的访问，包括表格数据(如Excel电子表格)，ISAM文件(像Access和dBase)，Active Directory，甚至是IBM host DB2数据。OLE DB实际上可以用来访问任何能以行列格式返回数据的数据源。OLE DB的创建也使用了更为现代的COM接口，将开发人员从ODBC较老的调用级接口中解放出来。每个OLE DB提供者提供数据访问，并通过它提供的COM接口反映其功能。但是，OLE DB COM接口是一个要求支持指针、数据结构和直接内存分配的低层接口。这些低层构造最适合于C++程序员。直接使用