数据访问组件 ADO. NET 的特性及编程

朱 帆
(复旦大学软件学院 上海 200433)

摘 要 ADO. NET 作为微软最新的数据访问技术,在应用系统开发中是一组实用的组件。本文结合自己学习编程的实际经验,首先介绍 ADO. NET 的组件结构和主要特性,然后重点描述了在 ASP. NET 中通过 ADO. NET 实现数据库操作的编程方法和使用技巧。

关键词 ADO. NET,数据库访问,组件

1 微软数据访问方式的发展

微软的数据访问方式经历了 ODBC,OLE DB,ADO 和 ADO. NET 几个阶段[1]。图 1 以图表的形式描绘了 ADO 和 ADO. NET 基本构架,这里简单解释几种方式的主要特征。

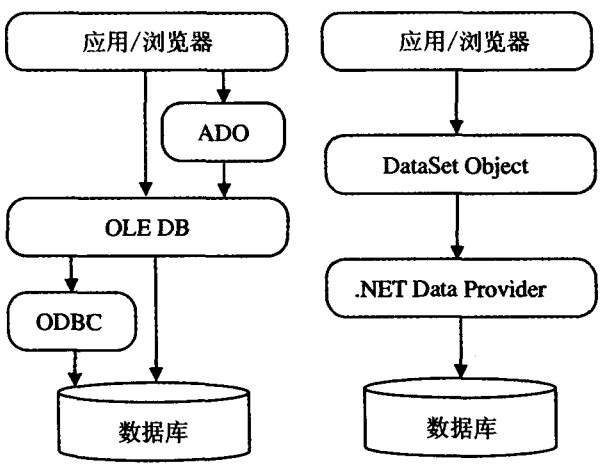

图 1 ADO 和 ADO. NET 的构架

(1)ODBC(Open Database Connectivity)阶段:这是最初使用 SQL 访问不同关系数据库的数据访问技术。使用 ODBC,应用程序能够通过单一的命令操纵不同的数据库,而开发者需要做的是针对不同的应用加入相应的 ODBC 驱动。

(2)OLE DB 阶段:随着越来越多的数据以非关系型格式存储,需要一种新的架构来提供这种应用和数据源之间的无缝连接,于是,基于 COM(Component Object Model)的 OLE DB 应运而生。

(3)ADO 阶段:ADO 基于 OLE DB,它不仅消除了 OLE DB 的多种弊端,而且更简单、更适合 VB 程序员。ADO 使用类似于单表的 Recordset 存储数据;ADO 的操作时采用在线方式,这意味着不论是浏览或更新数据都必须实时进行;ADO 使用 COM 技术,这就要求所使用的数据类型必须符合 COM 规范。

(4)ADO. NET 阶段:针对 ADO 使用 Recordset 存储数据的不便,ADO. NET 通过允许多个表集的组件 DataSet 来表示数据;针对 ADO 在线操作的缺陷,在访问数据的时候 ADO. NET 采用离线方式,它利用 XML 制作数据幅本,只有在与数据库的连接时才需要在线操作。ADO. NET 基于 XML 格式,数据类型比 ADO 更为丰富,且不需要再做 COM 编排导致的数据类型转换,从而提高了整体性能。

2 ADO. NET 组件结构

ADO. NET 是 ASP. NET 内置的重要组件,它的主要目的是为了存取或修改数据源的数据,或向指定的数据源增加数据。ADO. NET 由 Dataset 和. NET Framework 两个核心组件负责完成对数据源的数据的操作任务[2,3]。

ADO. NET Dataset 是 ADO. NET 的断开式结构的核心组件。Dataset 的目的是为了实现独立于任何数据源的数据访问,它可以用于多种不同的数据源,如 XML 数据或应用程序的本地数据。Dataset 包含一个或多个 DataTable 对象的集合,这些对象由数据行和数据列、主键、外键、约束,以及有关 DataTable 对象中数据的关系信息组成。DataSet 提供了一种内存中数据关系的表示形式,一整套包括表在内的数据(这些表包含数据、对数据进行排序并约束数据)以及表间关系。

ADO. NET 结构的另一个核心组件是数据提供程序. NET Framework,该组件的目的是实现数据操作和对数据的快速只读访问。它主要包括 Connection、Command、DataReader 和 DataAdapter 对象。Connection 提供与数据源的连接;Command 提供访问数据、返回数据、修改数据、运行存储过程以及发送或检索参数信息的数据库命令;DataReader 从数据源中提供高性能的数据流;DataAdapter 提供连接 Dataset 对象和数据源的桥梁;DataAda-

pter 使用 Command 对象在数据源中执行 SQL 命令，以便将数据加载到 Dataset 中，并使对 Dataset 中数据更改与数据源保持一致。

开发者可以为多种不同数据源编写数据提供程序。. NET Framework 提供了四种数据提供程序：ODBC . NET Framework 数据提供程序、OLE DB . NET Framework 数据提供程序、SQL Server . NET Framework 数据提供程序和 Oracle . NET Framework 数据提供程序。因此，在 ASP. NET 程序中，利用 ADO. NET 能够存取 ODBC 驱动程序所能存取的所有数据库（如 SQL Server，Oracle，Sybase，DB2 等），除数据库外，还能够访问一些小型数据表（如 Excel、FoxPro、Access 或文本文件等）。图 2 阐释了 ADO. NET 的组件结构。

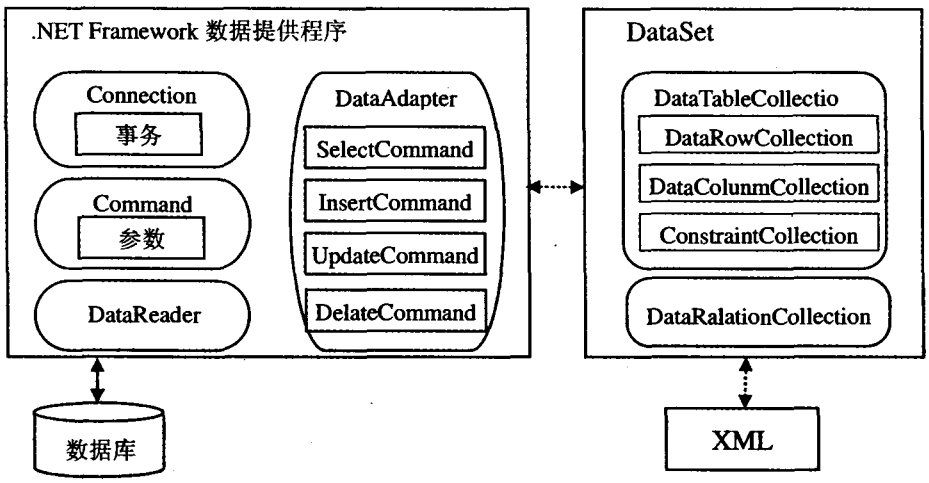

图 2　ADO. NET 组件结构图

3　通过 ADO. NET 实现对数据库的操作

下面具体介绍在 ASP. NET 程序中通过 ADO. NET 实现对数据库操作的编程方法。

3.1　创建数据源名(DSN)

首先，提供一条使 ADO 定位、标识和与数据库通信的途径，数据驱动程序通过使用 DSN(数据源名)来定位和标识特定的 ODBC 数据库，将信息从 Web 应用程序传递给数据库。DSN 一般包含数据库配置、用户安全性和定位信息，且可获取 Windows 2000 注册表项中的表格。通过 ODBC 选择希望创建的 DSN 的类型：系统 DSN、用户 DSN、文件 DSN。其中，系统 DSN 和用户 DSN 存储在 Windows 注册表中；系统 DSN 允许所有用户登录到特定服务器上去访问数据库；用户 DSN 使用适当的身份证明限制数据库到特定用户的连接；文件 DSN 用于从文本文件中获得表格，提供对多用户的访问，通过复制文件 DSN，很容易从一个服务器转移到另一个服务器。以下是建立系统 DSN 的具体步骤。

第一步：启动控制面板，双击 ODBC Data Source 图标，在“ODBC 数据源管理器”对话框中选择“系统 DSN 标签页”。

第二步：单击“添加”按钮，在弹出的“创建数据源”对话框中选择指定的驱动程序 SQL Server，并单击“完成”按钮。

第三步：在“建立新数据源到 SQI Server”对话框中输入“名称”、“说明”、“服务器”等选项，完成后单击“下一步”按钮。“名称”表示被建立的 DSN 的名称，如取 mydata；“说明”描述被建立的 DSN 的说明，属于可选择项；“服务器”指定 DSN 所连接的数据库服务器，使用时应根据情况输入适当的数据库服务器名称。

第四步：设置相关的用户权限，可决定使用 Windows 2000 服务器或 MS SQL Server 服务器的用户账号作为权限的账号。设置完毕后单击“下一步”按钮。

第五步：设置与数据库连接的选项，设置完毕单击“下一步”按钮。

第六步：出现“ODBC Microsoft SQL Srver 安装”对话框，框中将显示刚建立的 DSN 的相关设置值，单击“确定”按钮，整个过程完成。

3.2　数据库连接

访问数据库前需要与数据库中指定的数据源建立连接：应用 Connection 对象可建立应用程序与 ODBC 数据库之间的连接；调用该对象提供的方法和属性，可打开和关闭数据库的连接、发出查询请求等。方法如下：

第一步：用 Server 对象的 Createobject 方法建立 Connection 对象，记为 dbConnection：

set dbConnection = Server. Createobject (“ADODB. Connection”)

第二步：用 Connection 对象的 Open 方法打开数据库连接，设置连接并连接到数据库：

dbConnection. Open “DSN = mydata; UID = zf2004; PWD = 123456; Database = Logdata”

式中，DSN 表示在控制面板 ODBC 所设置的 DSN，UID 表示用户账号，PWD 表示用户密码，Database 指定在数据库服务器上的一个数据库名称。

3.3 访问数据库

方法一：用 Connection 对象的 Execute 方法来操作数据。通过执行结构化查询语言 SQL 命令，查询数据库。如对数据库表 Table_1 的操作语句：

dbConnection. Execute "insert into Table_1 (UserName) values('Li')"

dbConnection. Execute "update Table_1 set UserName='Zhu' where UserName='Li'"

dbConnection. Execute "delete from Table_1 where UserName='Liu'"

Connection 对象的优点是可以用于直接检索和显示数据库信息，以简化连接数据库和查询任务。Connection 对象的不足是它不能用于创建脚本，必须确切知道要对数据库做出哪些更改，然后才能使用查询实现更改。

方法二：对于检索数据、检查结果、更改数据库，ADO. NET 提供了 Recordset 对象。使用 Recordset 对象，根据查询条件，可检索并显示一组数据记录，并保持查询返回的记录位置。也可以通过执行结构化 SQL 命令，查询数据库并检索结果。从使用效率来讲，数据库应用程序一般使用 Connection 对象建立连接并使用 Recordset 对象处理返回数据，通过协调这两个对象的功能，可设计出能完成任何数据处理任务的数据库应用程序。步骤如下：

第一步：建立一个 Recordset 对象 Dbzhu：

Set Dbzhu = Server. Createobject ("ADODB. Recordset")

第二步：用 Recordset 对象的 Open 方法打开一个记录集：

SqlQuery = "select LogTime from Tabl_1 where day(LogTime)=day(getdate()) and month (Logtime)=month(getdate()) and year(Logtime) =year(getdate())"

Dbzhu. Open SqlQuery, dbConnection, 3, 3

第三步：用 SQL 语句对数据库进行操作。用 select 实现查询并把结果赋予 Recordset 对象；用 insert、update 和 delete 实现数据记录的增加、更新和删除等，如：

StrQuery="select * from Dbzhu where Caption='WorkName&'"

ADOcommand. CommandText=StrQuery

Set RS = Sever. Createobject ("ADODB. Recordset")

第四步：用 Response 和 Recordset 对象显示查询结果：

〈%Response. Write(RS. (Title))%〉

第五步：用 Recordset 对象的 Close 方法关闭指定的 Recordset 对象：

Dbrs. Close

用 Command 对象执行查询和用 Recordset 对象执行查询基本相似，不同的是 Command 对象可以在数据库源上准备和编译查询并反复使用一组不同的值来发出查询，可最大程度地减少向现有查询重复发出修改请求所需的时间，另外还可以在执行之前通过用户查询的可变部分的选项使 SQL 查询保持局部未定义。

3.4 关闭连接

当脚本执行完后，连接应当被终止。在不需要连接时应当将其关闭，以便其他用户能够建立和使用该连接。可以用 Connection 对象的 Close 方法来关闭已启动的 Connection 对象及其他对象，断开与数据库之间的连接，如：

dbConnection. Close

4 ADO. NET 编程的一些技巧

4.1 对于下列情况，在应用程序中需要使用 DataReader

(1)不需要缓存数据时。

(2)当要处理的数据结果集太大，内存中放不下时。

(3)当需要以仅向前、只读方式快速访问数据时。

在填充 DataSet 时，DataAdapter 使用 DataReader。因此，使用 DataAdapter 取代 DataSet 提升的性能表现为节省 DataSet 占用的内存和填充 DataSet 需要的循环。

4.2 基于索引在 DataSet 中搜索数据

在 DataSet 中查询与特定条件相匹配的行时，可以利用基于索引的查找提高搜索性能。当将 PrimaryKey 值赋给 DataTable 时，会创建一个索引。当给 DataTable 创建 DataView 时，也会创建一个索引。下面是利用基于索引进行查找的技巧。

(1)对于 DataTable 的 PrimaryKey 列进行查询，应使用 DataTable. Rows. Find 而不是 DataTable. Select。

(2)对于涉及非 PrimaryKey 列的查询，可使用 DataView 为数据的多个查询提高性能。当把排序顺序应用到 DataView 时，会建立一个搜索时使用的索引。DataView 公开 Find 和 FindRows 方法，以便查询 DataTable 中的数据。

(3)可通过为 DataTable 创建 DataView 来利用基于索引的查找。只有对数据执行多个查询操作时，创建索引才会带来好处。如果只执行单一查询，创建索引所需要的处理反而会降低查询性能。

参 考 文 献

1 孟庆臣. ADO 与 ADO. NET. www. microsoft. com/china/community/program/originalarticles

2 Sceppa D 著. 梁超，张莉，贺堃译. ADO. NET 技术内幕. 清华大学出版社，2003

3 Bipin J, Paul D, et al. Professional ADO. NET Programming. Wrox Press Limited, UK, 2005