**[ADO.NET入门教程（四） 品味Connection对象](http://www.cnblogs.com/liuhaorain/archive/2012/02/15/2349886.html)**

**摘要**

前几篇文章，我都没有详细讲解Data Provider核心对象，因为我希望在讲解这些对象之前，让大家对一些基础的概念有很好的认识。在上一篇文章[《你必须知道的ADO.NET（三） 连接字符串，你小觑了吗》](http://www.cnblogs.com/liuhaorain/archive/2012/02/12/2347914.html)中，我详细讲解了连接字符串，相信大家都和我一样意识到它的重要性了。如果说连接字符串是打开数据源大门的钥匙，那么我今天要讲解的则是如何用这把钥匙打开数据源的大门。作为Data Provider的第一核心对象，Connection对象肩负起连接数据源的重任。下面就让我们好好认识这位重量级人物吧！

**目录**

* [理解Connection对象](http://www.cnblogs.com/liuhaorain/archive/2012/02/15/2349886.html#title_1)
* [必须掌握的几个方法](http://www.cnblogs.com/liuhaorain/archive/2012/02/15/2349886.html#title_2)
* [必须掌握的几个属性](http://www.cnblogs.com/liuhaorain/archive/2012/02/15/2349886.html#title_3)
* [说说ConnectionState](http://www.cnblogs.com/liuhaorain/archive/2012/02/15/2349886.html#title_4)
* [实例：连接SQL Server的SqlConnection对象](http://www.cnblogs.com/liuhaorain/archive/2012/02/15/2349886.html#title_5)
* [编写优雅而又安全的代码](http://www.cnblogs.com/liuhaorain/archive/2012/02/15/2349886.html#title_6)

**1. 理解Connection对象**

Connection对象，顾名思义，**表示与特定数据源的连接**。如果把数据源比作大门，那么连接字符串则是钥匙，而连接对象则是拿着钥匙开门的人。对于ADO.NET而言，不同的数据源，都对应着不同的Connection对象。具体Connection对象如下表：

|  |  |  |
| --- | --- | --- |
| 名称 | 命名空间 | 描述 |
| SqlConnection | System.Data.SqlClient | 表示与SQL Server的连接对象 |
| OleDbConnection | System.Data.OleDb | 表示与OleDb数据源的连接对象 |
| OdbcConnection | System.Data.Odbc | 表示与ODBC数据源的连接对象 |
| OracleConnection | System.Data.OracleClient | 表示与Orale数据库的连接对象 |

 不管哪种连接对象，它都继承于DbConnection类。我们看看DbConnection类的实现结构：

public abstract class DbConnection : Component,   
 IDbConnection, IDisposable

从上面，我们可以看出DbConnection是抽象基类，并且继承Compoent,IDbConnection,IDisposable类。由于DbConnection类是抽象基类，因此它不能实例化。DbConnection类封装了很多重要的方法和属性，下面我将详细讲解几个重要的方法和属性。

**2. 必须掌握的几个方法**

**Open:**使用 ConnectionString 所指定的设置打开数据库连接。

**Dispose:**释放由 Component 使用的所有资源。

**Close:**关闭与数据库的连接。 **此方法是关闭任何已打开连接的首选方法。**Close 方法回滚任何挂起的事务。 然后，它将连接释放到连接池，或者在连接池被禁用的情况下关闭连接。

**3. 必须掌握的几个属性**

**Database:**在连接打开之后获取当前数据库的名称，或者在连接打开之前获取连接字符串中指定的数据库名。

**DataSource:**获取要连接的数据库服务器的名称。

**ConnectionTimeOut:**获取在建立连接时终止尝试并生成错误之前所等待的时间。

**ConnectionString:**获取或设置用于打开连接的字符串。

**State:**获取描述连接状态的字符串。

**4. 说说ConnectionState**

上面我们知道，**State属性描述了与数据源的连接的当前状态。**ConnectionState是一个枚举类型。它包括以下成员：

**Closed:**连接处于关闭状态。

**Open:**连接处于打开状态。

**Connecting:**连接对象正在与数据源连接。

**Executing:**连接对象正在执行命令。

**Fetching:**连接对象正在检索数据。

**Broken:**与数据源的连接中断。

**5. 实例：连接SQL Server的SqlConnection对象**

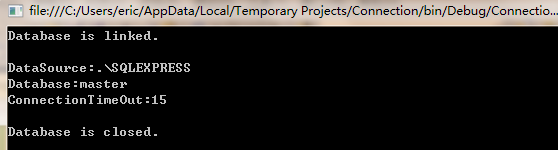
上面说了那么多理论知识，下面就讲一个连接SQL Server的实例吧！代码如下：

[复制代码](javascript:void(0);)

1 using System;  
 2 using System.Collections.Generic;  
 3 using System.Linq;  
 4 using System.Text;  
 5 using System.Data;  
 6 using System.Data.SqlClient;  
 7   
 8 namespace Connection  
 9 {  
10 class Program  
11 {  
12 static void Main(string[] args)  
13 {  
14 //构造连接字符串  
15 SqlConnectionStringBuilder connStr = new SqlConnectionStringBuilder();  
16 connStr.DataSource = @".\SQLEXPRESS";  
17 connStr.InitialCatalog = "master";  
18 connStr.IntegratedSecurity = true;  
19   
20 SqlConnection conn = new SqlConnection();//创建连接对象  
21 conn.ConnectionString = connStr.ConnectionString;//设置连接字符串  
22   
23 conn.Open();//打开连接  
24   
25 if(conn.State == ConnectionState.Open)  
26 {  
27 Console.WriteLine("Database is linked.");  
28 Console.WriteLine("\nDataSource:{0}",conn.DataSource);  
29 Console.WriteLine("Database:{0}",conn.Database);  
30 Console.WriteLine("ConnectionTimeOut:{0}",conn.ConnectionTimeout);  
31 }  
32   
33 conn.Close();//关闭连接  
34 conn.Dispose();//释放资源  
35   
36 if(conn.State == ConnectionState.Closed)  
37 {  
38 Console.WriteLine("\nDatabase is closed.");  
39 }  
40   
41 Console.Read();  
42 }  
43 }  
44 }

[复制代码](javascript:void(0);)

结果：



**6. 编写优雅而又安全的代码**

**（1）添加try...catch块**

我们知道连接数据库时，可能出现异常，因此需要添加异常处理。对于C#来说，典型的异常处理是添加try...catch代码块。finially是可选的。finially是指无论代码是否出现异常都会执行的代码块。而**对数据库连接资源来说，是非常宝贵的**。因此，我们应当确保打开连接后，无论是否出现异常，都应该关闭连接和释放资源。所以，我们必须在finially语句块中调用Close方法关闭数据库连接。

[复制代码](javascript:void(0);)

1 SqlConnection conn = new SqlConnection(connStr);  
 2 try  
 3 {  
 4 conn.Open();  
 5 }  
 6 catch(Exception ex)  
 7 {  
 8 ;//todo  
 9 }  
10 finially  
11 {  
12 conn.Close();  
13 }

[复制代码](javascript:void(0);)

**（2）使用using语句**

另外一种优雅的方法，则是使用using语句。如果你还不熟悉using语法，我就再啰嗦几句。**using语句的作用是确保资源使用后，并很快释放它们。**using语句帮助减少意外的运行时错误带来的潜在问题，它整洁地包装了资源的使用。具体来说，它执行以下内容：

* 分配资源。
* 把Statement放进try块。
* 创建资源的Dispose方法，并把它放进finally块。

因此，上面的语句等同于：

1 using(SqlConnection conn = new SqlConnection(connStr))  
2 {  
3 ;//todo  
4 }