Is my overview of .Net Data Providers and ADO.Net correct?

Asked 14 years, 4 months ago Modified 14 years, 4 months ago

Viewed 651 times

1

Recently I had to type up some documentation on .net data providers and ado.net. I am trying to get feedback on my findings. Please review and provide corrections or opinions.

Summary This is a high level summary of the basic .Net API's for interacting with a database. As a developer with mainly a Java and PHP background I was unclear about how ADO.Net related to OleDb and I had no idea what was meant by the term ".Net Data Provider". I created this because the msdn documentation is HEAVILY focused on ADO.Net and does not give a clear picture of how the many namespaces, interfaces, and classes interact.

.Net Data Provider

- http://msdn.microsoft.com/en-us/library/a6cd7c08(v=VS.71).aspx
- A .NET Framework data provider is used for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, or placed in an ADO.NET DataSet.

- What that actually means is that a .Net Data Provider implements the interfaces defined in the System.Data namespace.
- A .Net Data Provider is similar to a JDBC driver in Java

System.Data

- http://msdn.microsoft.com/en-us/library/system.data.aspx This page contains text that makes you believe that ADO.Net is the CORE part of .Net data access, however the reality is that ADO.Net is the highest level of data access and is built upon the .Net Data Providers that implement the interfaces in the System.Data.
- In my opinion it almost seems like microsoft is trying to hide how database connections work, so that users are trapped using controls provided by visual studio. The System.Data namesapce contains Interfaces that need to be defined by ALL .Net Data Providers

System.Data Core Interfaces

- IDbConnection
- IDbCommand
- IDataAdapter
- IDataReader

Examples of System.Data Implementations

- The following namespaces include classes that implement the core “.Net Data Provider” interfaces defined in System.Data
- System.Data.SqlClient
- System.Data.OleDb
- System.Data.Odbc
- IBM.Data.DB2
- ByteFX.Data.MySqlClient

ADO.Net

- http://msdn.microsoft.com/en-us/library/27y4ybxw(v=VS.71).aspx
- ADO.Net is a database query and manipulation API built on top of the basic .Net Data Provider classes. ADO.Net focuses on disconnected, multi-tier database interaction.
- In my opinion the core ADO.Net classes should be in a separate namespace like System.Data.ADO just for the sake of clarity.

ADO.Net Core Classes

- DataSet
- DataTable
- DataColumn
- DataRelation

.net ado.net

Share

Improve this question

Follow

asked Aug 6, 2010 at 18:17

Blake Mills

844 ● 2 ● 9 ● 15

2 Answers

Sorted by: Highest score (default)

3

System.Data is the "package" that contains everything for working with "Data providers" in .NET. It is true that ADO, is one strategy for working with data, but it is the primary strategy in .NET.

ADO is less about specific DB technologies (as it is not necessarily meant to be a database specific technology) and more about data relationships. The terms: Set, Table, Column, Row and Relationship are well understood modeling terms and ADO.NET makes them first class objects in the .NET space.

Data providers provide low-level implementation specific details for supporting the core ADO.NET concepts (tables, rows, etc) and are meant to abstract away the direct implementation details of how to connect to a data provider. For instance, you should be able to, with relatively little effort, swap out a Jet Data provider, with a Oracle Data provider, in terms of DataTables, DataRows, and DataColumns (query details aside) such that your code is minimally impacted by the change. Why is this

important? Because it means you can work with non-homogenous data sources with similar command semantics (i.e. you can work with excel spreadsheets and MySql dbs in the same app with the same objects). This makes reuse and repurposing very easy and very straight-forward.

As a general view you can think of the system this way:

1. The .NET Data provider is where you get your data from. You will need to import from System.Data each provider that your app uses
2. The ADO.NET classes are the concepts you will be working with Tables, Rows, Columns, etc. These have nothing to do with queries, indexes, etc. Those are provider details
3. Your app should only rarely (I would say never, but there are always exceptions) need to be aware of the Provider and instead focus on consuming and producing DataSets, DataTables, etc.

Hope that helps.

answered Aug 6, 2010 at 18:32

GrayWizardx

21.1k ● 2 ● 33 ● 44

This helps a lot. I wish the MSDN docs were this clear. My background in Java and php made it hard to relate to ADO.Net(DataTable specifically). I totally agree about programming independent of a specific provider. You can do that even without the DataTable and DataSet classes, which

as you said are independnet of providers because they are concrete classes in the System.Data namespace. If you hide the "Data provider" implementation behind the interfaces in the System.Data namespace you can change your provider at anytime. Thx for the reply. – Blake Mills Aug 6, 2010 at 20:37

@Welzie its true you dont need to use the DataTable and DataSet as the DataReader provides a much easier to understand DB reader semantic, but DataTable (DT) and DataSet (DS) are meant to convey a different meaning than a one-way movement of data. The DT and DS are meant to provide proxy implementations of data usage and give you a disconnected interface to your data provider, such that the DT/DS "act" as the database, and the underlying details are not important to consuming code. I think in practice this is not how the DT and DS are used though, usually they are just containers for data. – GrayWizardx Aug 6, 2010 at 22:29

2

In reference to:

> In my opinion it almost seems like microsoft is trying to hide how database connections work, so that users are trapped using controls provided by visual studio.

The existence of 3rd party providers does not support the idea that MS is trying to "trap" us into anything.

Hiding "how database connections work" is a kind of abstraction, and is not subversive.

Share Improve this answer answered Aug 6, 2010 at 18:33

Follow

kbrimington

25.6k ● 5 ● 66 ● 75

Understood, but hiding how a crucial part of a system happens limits a developers understanding. In reality at this level everything is an abstraction. If you only learn the ADO.Net api and don't learn about connections, transactions, commands I feel a developers scope on database interaction is extremely limited. Thx for the reply. – Blake Mills Aug 6, 2010 at 20:32