

Re: Enterprise Library – DataAccessApplicationBlock – MySql/DB2

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.adonet/2006-03/msg00415.html>

- *From:* "W.G. Ryan – MVP" <WilliamRyan@xxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 13 Mar 2006 16:01:24 -0500
-

With respect to Connection Strings, I responded to one of your other posts (lemme see if I can find it) with how to address that. The DBConnectionStringBuilder implementations will handle this though.

With respect to JDBC though, (I admit, it's been 4 years since I used it), how does it handle named parameterization when the underlying db doesn't?

The fact that you set parameters differently for different providers isn't really ADO.NET's fault though. Different vendors implement their providers differently, but even if they wanted to, Microsoft would have a hard time forcing people to implement them in a standard way across the board. if I were to build BillsOracleClient b/c I didn't like the OracleClient implementation, what I'd need to do is work with the interfaces and base classes. However I could implement those interfaces however I pleased. If Oracle or Microsoft told me "We don't like the way you did that", I'd be free to tell them to go fly a kite. It's my provider I can implement it how I want. Now let's say that you did the same for a DB2 Client and chose something that was specifically much different from mine. Microsoft could call us up and say "Hey you guys, we're trying to keep things consistent so we want you to change it so people can use the same param constructors for each of your providers" but we'd be free to say no. Without changing the architecture in a serious way, there's not much they can do to force people to implement the interfaces a certain way, nor do i think it would all that well recieved if they did. Moreoever, by giving you the interfaces, the world is your with respect to creating providers. The downside is that you don't have cross -db compatibility in a true sense, but you don't anyway in most cases b/c you want to take advantage of db specific features. So if you wanted to use += for a join, you wouldn't be able to use it for all of the following, Oracle, Sql Server and Access. Well, Joins are a pretty fundamental staple and the syntactical differences would stop you. same for many functions, ISDATE doesn't produce the same results in both Sql Server and Access. There's no DUAL in Sql Server or Access. There are many more similar issues that stop you from having true agnostic DB access and these are much more serious than the parameter issue. If you can't use the same syntax because of the back ends, then the rest is more or less academic. To be honest, I really can't see this changing in any version and I don't see much desire to even move in that direction. Now more than ever Sql Server

and Oracle both have a ton of specific features that others don't have. To take advantage of them you need to use a specific library. And in most cases, the opportunity costs associated with not using these features far outweigh the benefit is being able to use the same syntax for both. I can't speak for Microsoft but I think these differences are going to widen more than get smaller.

"David Thielen" <thielen@xxxxxxxxxxxxxx> wrote in message
news:EEF77518-C58F-4B98-8C2E-247B99317878@xxxxxxxxxxxxxx

Hi;

Good points and I agree. JDBC does do this a lot better, but they have had a lot longer time to get it to that point. I do think the one major thing left to address in ADO.NET is a standard way of setting the parameters in the SQL statements. I don't think there is any inherent reason why setting parameters in a select must be different for each database.

I think it is going to take the next version of ADO.NET (2.1?) to get us to where I would like it to be. Where for simple CRUD the only difference is the connection string.

--
thanks – dave
david_at_windward_dot_net
<http://www.windwardreports.com>

"W.G. Ryan – MVP" wrote:

David:

The ApplicationBlock isn't ADO so it's portability limitations don't reflect ADO in any way.

The fact that there are specific providers and code handled differently is definitely by design as opposed to them leaving something out. In order to take advantage of many features of Sql Server or Oracle, you simply can't use generic features. The more generic you go, the more you compromise on either functionality or performance, and to the best of my knowledge,

Re: Enterprise Library – DataAccessApplicationBlock – MySql/DB2

this tradeoff is inherent and really couldn't change. You can write agnostic code but that comes at the expense of functionality or performance, something most developers don't typically want to compromise on. So you can have it either way, going totally portable or totally specific and the current arrangement provides the provider specific stuff by default. You have to write your own generic features.

the DataSet/dataTable model is truly portable though, it couldn't care less where data came from or where it's going. You decide how to get it where you want it to go. This separation seems logical and solid IMHO.

As far as 'true' portability. You can scrap Paramaters altogether (not advised but possible), and use dynamic sql. SQL is ANSI standard so the code should fire against any Sql Compliant DB. You can use the ProviderFactory to create the commands and connections and adapters. Yes, there's a lot of problems with this, performance aand security being two of the bigger ones. But everything is a tradeoff...

"David Thielen" <thielen@xxxxxxxxxxxx> wrote in message news:1DEF5911-6959-45C1-BB97-0D2DD52EB93A@xxxxxxxxxxxxxxxx

Ok – thanks this lets me know to not use DataAccessApplicationBlock – you just saved me some time.

It looks like I have to write my own code to handle each client. Maybe in 3.0 ADO will be like JDBC where it is truly portable...

—
thanks – dave
david_at_windward_dot_net
<http://www.windwardreports.com>

"Luke Zhang [MSFT]" wrote:

The DataAccessApplicationBlock currently only includes classes that are specific to Microsoft SQL Server and Oracle. You have to use ADO.NET

Re: Enterprise Library – DataAccessApplicationBlock – MySql/DB2

with
DB2 and/or MySql directly.

Regards,

Luke Zhang
(This posting is provided "AS IS", with no
warranties, and confers no
rights.)