

Re: ODBC/OLE DB Connection Pool

Source:

<http://www.tech-archive.net/Archive/Internet-Server/microsoft.public.inetserver.asp.db/2004-11/0416.html>

From: Stuart Carnie (*StuartCarnie_at_newsgroup.nospam*)

Date: 11/29/04

Date: Mon, 29 Nov 2004 13:20:54 -0700

- > *The first two are not relevant in ASP. The last one is also not relevant:*
- > *ADO is not free-threaded, so it is not recommended that a single global*
- > *connection be kept open for the application as this will serialize all*
- > *communications with the database.*

This is incorrect – OLE DB objects are free threaded. It does not serialize database access – my test application demonstrates this as it is very multithreaded, and the connections are reused without any problems across threads ONCE they are returned to the pool.

In actuality, the IDataInitialize object is global in the OLEDB32.dll per process already. All OLE DB connection objects are constructed through a serialized call to GetDataSource, however it is minimal overhead.

Having a global object that is not used anywhere else is just fine, as in my detailed article, it's only purpose is to `_initialize_` and retain a reference to IDataInitialize to enable OLE DB Resource Pooling and nothing else.

Cheers,

Stu

"Bob Barrows [MVP]" <reb01501@NOyahoo.SPAMcom> wrote in message news:ec%23njN%230EHA.3336@TK2MSFTNGP11.phx.gbl...

- > *Erland Sommarskog wrote:*
- > > *Bob Barrows [MVP] (reb01501@NOyahoo.SPAMcom) writes:*
- > > > *If you read Stuart's post carefully, you will see that you do not*
- > > > *have to do anything to turn Session pooling on. It's on by default.*
- > > > *You have to do extra work to turn it off.*
- > >
- > > *Well, in VB it as easy as:*
- > >
- > > *Public Function Button1_Click () As*
- > > *Dim cnn As new ADODB.Connection*
- > > ...

> > *Set cnn = Nothing*
> > *End Function*
> >
> > *And if there is no global ADODB.Connection, there will be no pooling.*
> > *(Which Stuart also discussed in his post.)*
> >
> > *If the above is possible to do in ASP, I don't know.*
>
> *Oh wait, It HAS been a while since I read this article. I now see this*
> *paragraph:*
>
>

> *Tips for ADO Users*
> *The ADO Connection object implicitly uses IDataInitialize. However, this*
> *means your application needs to keep at least one instance of a Connection*
> *object instantiated for each unique user—at all times. Otherwise, the pool*
> *will be destroyed when the last Connection object for that string is*
closed.
> *(The exception to this is if you use Microsoft Transaction Server. In this*
> *case, the pool is destroyed only if all of the connections in the pool*
have
> *been closed by client applications and are allowed to time out.)*
> *Note If you open a Connection object, remember to close it. Do not rely*
on
> *garbage collection to implicitly close your connections.*
> *****
>
>
> *I do question whether this means that the pool won't be recreated the next*
> *time a connection is opened*
>
> *And check out the exception noted above regarding the use of MTS. In ASP,*
if
> *you use Server.CreateObject, you are forcing the use of MTS. Using*
> *vbscript's CreateObject will not cause MTS to be used and you may lose*
the
>
>
> *And this one (I've added numbers):*
>

> *Unintended Disabling of Resource Pooling*
> *Resource pooling can be inadvertently disabled for your application. The*
> *persistence and behavior of resource pooling depend on several conditions*
> *that occur with a given set of user authentication criteria, such as the*
> *following:*
>
> *1. A given resource pool is specific to the connection attributes of that*
> *set of authentication criteria.*

- > *Do not call IDBInitialize::Uninitialize, which implicitly disables pooling.*
- > *Instead, use IDBInitialize::Release. While both IDBInitialize::Uninitialize*
- > *and IDBInitialize::Release close the connection, calling Uninitialize will*
- > *result in the pool being destroyed. (This is not an issue for ADO*
- > *developers.)*
- >
- > *2. The data source object must be created by using IDataInitialize or*
- > *IDBPromptInitialize, and not by CoCreateInstance directly on the OLE DB data*
- > *source provider. For pools to exist for a given set of connection attributes,*
- > *resource pooling requires at least one instance of an OLE DB data source*
- > *object created per unique set of connection properties.*
- >
- > *3. If using ADO, at least one ADO Connection object must remain open for*
- > *each set of user authentication credentials. Although ADO cannot call*
- > *Uninitialize, letting go of all ADO Connection objects for a given set of*
- > *connection attributes accomplishes the same thing—that is, it releases the*
- > *last instance of IDataInitialize.*
- >
- *****
- ****
- >
- > *The first two are not relevant in ASP. The last one is also not relevant:*
- > *ADO is not free-threaded, so it is not recommended that a single global*
- > *connection be kept open for the application as this will serialize all*
- > *communications with the database.*
- >
- >
- > *Bob Barrows*
- > --
- > *Microsoft MVP – ASP/ASP.NET*
- > *Please reply to the newsgroup. This email account is my spam trap so I*
- > *don't check it very often. If you must reply off-line, then remove the*
- > *"NO SPAM"*
- >
- >