

Re: Linking to a .NET dll from C#

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.general/2009-04/msg00208.html>

- *From:* Jesse Houwing <jesse.houwing@xxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 24 Apr 2009 09:25:52 +0000 (UTC)
-

Hello Bob,

Cor,

The application is for a destination gateway that a client is using for obtaining price quotes and performance is absolutely critical. The users of this data will all be on the same computer per the client's request (trust me on this issue – I am aware that other price servers usually use a socket connection to communicate price updates to clients on other machines). I suppose another way to do it would be to use shared memory or memory mapped files.

In the past when using VC++, I believe we used COM and called CreateInstance on the object which would return a pointer to the object and then method calls were very efficient – like making a virtual method call in C++. This was many years ago though so I'm a little rusty and don't remember if we used a different technique to actually link to an existing dll or not. The question is – can something like this be done in a managed .NET language like C#, and if so what is the best way? Should we consider using com interop for this? Thought com was dead...

Though I believe you're looking for a little extra performance that has no real merit here, there are a few things to consider. 1) Place the shared assembly in the GAC, so that all apps will share the same binary 2) Use pre-compilation to create native images of these assemblies

If you want only one instance live at any time, you could host this 'layer' so you want in a service and use WCF through a local transport (named pipes or net.tcp) in combination with binary serialization. It would mimic the COM solution, but at the same time would be an all .NET way to go.

If these machines you're working on are Vista or newer, you could use the Host Activation Service to make the hosting of the WCF library a little simpler.

Lastly, there are tools out there (like XenoCode Postbuild) that allow you to actually statically link any assemblies and merge them into the application. These tools will sometimes even allow you to link the whole framework with your application. Be aware that in these scenarios you're responsible for patching and

Re: Linking to a .NET dll from C#

updates, as the application will no longer rely on the clients machine to keep the framework up to date.

Hope this helps,

Jesse

"Cor Ligthert[MVP]" <Notmyfirstname@xxxxxxxx> wrote in message
news:e7b69YJxJHA.1212@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Bob,

Has this any goal, I did this when memory in Dos was limited to 640Kb
(and there was absolute nothing more for a program despite all kinds
of extended memory possibilities)

But now I don't see any reason anymore for that.

I am curious why you want this.

Cor

"Bob Bryan" <RobertGBryanREMOVETHIS@xxxxxxx> wrote in message
news:7A86E279-6A12-4627-8EAA-B84124960B97@xxxxxxxxxxxxxxxxxxxx

I am trying to find out if its possible to do the following from
C#:

1. Determine if a certain managed .NET dll or assembly is loaded into memory or not. If not, then load it.
2. If the dll is already loaded, then get a reference to it so that its members can be called directly. The apps that will be using the dll already knows all the methods.
3. There are several apps (exes) that will be using the dll and I'm wondering if there is a way to dynamically link to a specific dll that is already in memory and get back a reference (or pointer) that will be just as fast as if each app had used static linking.

Can this be done, if so an example or link would be appreciated.

Re: Linking to a .NET dll from C#

—
Jesse Houwing
jesse.houwing at sogeti.nl