

## Re: Trying to use unmanaged with C#

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.interop/2007-02/msg00051.html>

---

- *From:* "Steven Edison" <EdisonCPP@xxxxxxxxxxxxxxxxxxxxxx>
  - *Date:* Thu, 8 Feb 2007 15:38:12 -0500
- 

"Patrick Steele" <patrick@xxxxxxx> wrote in message  
<news:MPG.20354b8b5589ecee9896ee@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>

In article <uLirTZ5SHHA.3592@xxxxxxxxxxxxxxxxxxxxxxxxxx>,  
EdisonCPP@xxxxxxxxxxxxxxxxxxxxxx says...

Looking back, I probably didn't clarify very well that  
the original COM obj is unmanaged, and I'm trying  
to use it in a managed environment.

Sorry Steve. Looking back at your original post, I do see that you're  
consuming a COM object, not trying to create one in .NET. Sorry about  
that.

However, as part of the COM interface, do you need to pass a reference  
of a .NET object to COM? And you're doing this from an ASP.NET app?

It only became the need to be a .NET object when I "imported" it into my  
managed app. Originally it was a pointer to an IUnknown implemented  
class instance that would be a callback. Yes, it is ASP.NET.

The actual object is a front end for communications with a server app.  
The object houses all the protocol for clients. So instead of sending  
all the socket protocol to login you can simply:  
pObj->Authenticate(name, pwd);

And then ask questions of the server app:  
pObj->GiveMeAllTheseItems((IUnknown\*)pWhereToSendResults,  
PassThruPointer/\*so you can do something with results\*/);

The new site needs talk to this server, and the historical way we've talked  
to it through numerous apps, is this COM object. The methods house  
very lengthy protocol and error checking, that are proven and solid.

First off, I'm no ASP.NET expert, but I think you might have some

## Re: Trying to use unmanaged with C#

security issues running a COM object from inside ASP.NET since COM is unmanaged. I think the default security for ASP.NET is to not allow unmanaged execution of code.

Would you think this still be an issue since when I referenced it – it created a managed dll like: bin/Interop.MyLib.dll?  
BTW, it seems up until this callback the COM object is operating correctly.  
I can:  
pObject.Connect(IP, port);  
pObject.Authenticate(name, pword);  
//etc..  
It's just when the COM object tries to QueryInterface() internally on the class instance I created within C# that it throws an exception.

Second, if you \*do\* need to pass a reference of some .NET class to a COM object, then you'll need to make sure your assembly which contains this class is marked as COM visible. It looked like you made your class COM Visible via the ComVisible attribute, but the assembly (the "parent" in the hierarchy defined in the error message) defaults to COM visible false.

How can I make it be ComVisible(true) when it converts it to a managed assembly?

What I would do is create a class library that does what you need (interact with the COM object) and get that debugged and working outside ASP.NET. Then just add a reference to that class library inside your ASP.NET application once everything is working.

I'll give this a try. Create a managed class library assembly that references the comm object? Then reference this new assembly in a test app, but eventually in the ASP.NET pages? It would strike me that very quickly I will run into the same Query problems with the callback implementation.

The end goal is to ask an unmanaged C++ Windows Service questions, get answers, and have it do things. The unmanaged COM object can do this now, wrapping all the lengthy protocol and error checking.

Re: Trying to use unmanaged with C#

—  
Patrick Steele  
<http://weblogs.asp.net/psteele>