Displaying path/version information for a loaded COM object?

Asked 16 years, 2 months ago Modified 16 years, 2 months ago

Viewed 257 times

0

I've got some code that uses the Component Categories manager to load all of the classes that implement a particular category.

Is there an easy way to get a description, path and version information from the loaded DLL or EXE?

com

Share

Improve this question

Follow

asked Oct 7, 2008 at 11:06

Roger Lipscombe

91.6k • 59 • 252 • 396

2 Answers

Sorted by: Highest score (default)

1

When the object is loaded in-proc (i.e. from a DLL) in the same apartment, there are potentially some tricks you could do to find the DLL in memory. For instance, if you look in the virtual method table (vtable) for code pointers into the live object, they would usually point into the DLL.

You could then use some system calls to determine which loaded DLL that code is part of.

But there are many potential pitfalls. If you have to load these objects into a separate apartment, then the code pointer will point to a stub and not the actual code. Also, many COM libraries actually implement the common interfaces in wrapper classes provided by the runtime, so this would be highly likely to give you false information in many common use cases. (I.e. you would end up getting the info for a runtime DLL, typically MFC or ATL).

When an object is loaded out-of-proc (i.e. from an EXE), I don't know any plausible way to chase down which EXE corresponds to the live object. (Clearly this data must exist somewhere in your process or in the COM runtime but its buried somewhere below the stubs, and is probably dependent on the version of Windows you are running).

So unless you are looking at a very restricted set of objects (all in-proc that load in the same apartment as you), probably your best bet is to use the registration information in the Registry to look up what you need. This is troublesome because this is one of those things that can change from version to version of the OS, but fortunately COM has been around long enough that this hasn't changed a ton over the years.

Given a CLSID of an object you got from the Component Categories Manager, you would look up the corresponding binary file as follows:

1. Open registry key `HKEY_CLASSES_ROOT\CLSID\{xxxxxxxxx-yyyyy-zzzz-aaaa-bbbbbbbbbbbbb}` where string inside the braces is the CLSID of the object you want to find.
2. If this is an in-proc object, there will be a sub-key named InProcServer32 whose "default" `REG_SZ` contains the full path to the DLL you need.
3. If this is an out-of-proc object, there will be a sub-key named LocalServer32 whose "default" `REG_SZ` contains the full path to the EXE you need. In some cases you may have to trim command line switches off this string to get just the EXE path.
4. With the DLL or EXE from the previous steps, you can call GetFileVersionInfo() in Win32 (or use System.Diagnostics.FileVersionInfo.GetVersionInfo if you have .NET available) to retrieve the version information structure from the executable, which will have version and description in it.

edited Oct 7, 2008 at 19:51

answered Oct 7, 2008 at 19:46

Tim Farley
11.9k ● 4 ● 31 ● 30

One way to get this information, assuming that it's only being used for logging purposes, is to simply wait until all

0

the initialisation's done, and then to enumerate loaded modules, dumping details about each to the log file.

Obviously, this only works for inproc objects, and it won't work if the application's very dynamic, but it'll also catch non-COM DLLs.

Share Improve this answer

Follow

answered Oct 15, 2008 at 18:20

Roger Lipscombe
91.6k 59 252 396