Psychic debugging: Because of course when something doesn't work, it's because the program was sabotaged from above

devblogs.microsoft.com/oldnewthing/20110210-00

February 10, 2011

Raymond Chen

When something stops working, you begin developing theories for why it doesn't work, and normally, you start with simple theories that involve things close to you, and only after you exhaust those possibilities do you expand your scope. Typically, you don't consider that there is a global conspiracy against you, or at least that's not usually your first theory.

> I'm trying to use the XYZ.DLL that comes with your product. I have successfully registered this DLL (as specified in the documentation) by performing a `regsvr32 C:\path\to\XYZ.DLL`.
>
> According to the documentation, I should now be able to create a `Xyz.XyzWidgetizer` object, but when I try to do so from C#, I get the exception
>
> ```
> Retrieving the COM class factory for component with CLSID
> {...} failed due to the following error: 80040154.
> ```
>
> I tried using the Visual Basic code sample which comes with the documentation, which contains only two lines:
>
> ```
> Dim oXyzWidgetizer
> Set oXyzWidgetizer = WScript.CreateObject("Xyz.XyzWidgetizer")
> ```
>
> However, it still fails with the following error:
>
> ```
> Microsoft (R) Windows Script Host Version 5.7
> Copyright (C) Microsoft Corporation. All rights reserved.
> C:\test.vbs(2, 1) WScript.CreateObject: Could not create object
>                   named "Xyz.XyzWidgetizer".
> ```
>
> Has support for the XyzWidgetizer been silently dropped?

Let's look at the error message more closely. Error 80040154 is `REGDB_E_CLASSNOTREG`: The class is not registered. Therefore, whatever `regsvr32` did, it didn't register the class.

> My psychic powers tell me that you registered the 32-bit version of XYZ.DLL on a 64-bit machine.

Registering the 32-bit DLL records the entries into the 32-bit registry (because 32-bit programs run in an emulator), and the 32-bit registry is not consulted when you try to create a COM object from a 64-bit application. Letting 64-bit applications see the registration for 32-bit DLLs doesn't actually accomplish anything because you cannot load a 32-bit DLL into a 64-bit process and vice versa—even if a 64-bit process can figure out what DLL it wants, it won't able to load it.

It so happens that my psychic powers were correct. How did I know that the person asking the question was running the 32-bit version of XYZ on a 64-bit version of Windows? I didn't, but it was the simplest theory that fit the (extremely limited) data. And it didn't involve a global conspiracy.

Raymond Chen

Follow