How do I detect Windows 10 if I cannot GetProcAddress for the function IsWindows10OrGreater?

devblogs.microsoft.com/oldnewthing/20170112-00

January 12, 2017

Raymond Chen

A customer wanted to use the handy functions provided in the `VersionHelpers.h` header file, like `IsWindows10OrGreater`, while maintaining the same source code for both Windows 7 and Windows 10 targets. Their plan was to `LoadLibrary` for `kernel32.dll`, and then `GetProcAddress` for `IsWindows10OrGreater`, but they found that the `GetProcAddress` call always failed, even on Windows 10. They looked in `kernelbase.dll` and `ntdll.dll`; no luck there either. How is it possible that Windows 10 doesn't know whether it is Windows 10?

The customer investigated further and found that when their test program called `IsWindows10OrGreater`, there was no call to `LoadLibrary` at all!

```
0:000> k
# ChildEBP RetAddr
00 0133f9fc 00c01806 ntdll!VerSetConditionMask+0x14
01 0133fc2c 00c01739 Test!IsWindowsVersionOrGreater+0xa6
02 0133fd0c 00c01a33 Test!IsWindows10OrGreater+0x29
03 0133fe10 00c022be Test!main+0x23
04 0133fe24 00c02120 Test!invoke_main+0x1e
05 0133fe7c 00c01fbd Test!__scrt_common_main_seh+0x150
06 0133fe84 00c022d8 Test!__scrt_common_main+0xd
07 0133fe8c 77a962c4 Test!mainCRTStartup+0x8
08 0133fea0 77bd0609 KERNEL32!BaseThreadInitThunk+0x24
09 0133fee8 77bd05d4 ntdll!__RtlUserThreadStart+0x2f
0a 0133fef8 00000000 ntdll!_RtlUserThreadStart+0x1b
```

The customer wanted to know how to call functions like `IsWindows10OrGreater` dynamically.

The reason the customer cannot find the function `IsWindows10OrGreater` in `kernel32.dll` or any other DLL is simple: The function was inside you all along.

The functions in the `VersionHelpers.h` header file are all inline functions. They are not exported anywhere. These functions do the grunt work of figuring out the operating system so you don't have to write the version detection code yourself (and invariable mess up) by

building the appropriate query and calling `VerifyVersionInfo`, which has been available since Windows 2000 (possibly longer).

If you think about it, the answer must be like that, for how could `kernel32.dll` export all of these specific version-checking functions? The Windows 7 version of `kernel32.dll` would have to be clairvoyant and have exports for all of these functions like `IsWindows10OrGreater`, which would be quite a feat. Presumably the implementations would simply be hard-coded to return either `TRUE` or `FALSE`, as appropriate. (I guess you could imagine that each version of Windows exports only the functions for which it returns `TRUE`, and the absence of the function implies that the corresponding version is not installed.)

So just go ahead and use the functions in `VersionHelpers.h`. They will always work and give you an answer. (Well, unless you're targeting systems earlier than Windows 2000, but if you're doing that, then you probably aren't too interested in version detection since your customer that is still running Windows NT 4.0 is unlikely ever to upgrade.)

Bonus chatter: Note that the operating system version check does raise its own question: "Why are you doing an operating system version check at all?" Because that sort of thing gives the application compatibility team the heebie-jeebies. We asked, but the customer never did answer that question.

Raymond Chen

Follow