The program “G” is preventing you from shutting down

devblogs.microsoft.com/oldnewthing/20191030-00

October 30, 2019

Raymond Chen

So you’re trying to sign off or shut down, but you get a message saying that some program named “G” is preventing you. What’s going on? Who is this mysterious “G” program?

We noted some time ago that treating a UTF16-LE Unicode string as if it were an 8-bit string generally results in only the first character surviving, because the high-order byte of the UTF16-LE code unit is zero for most Western European characters, and the zero byte is interpreted as the end of the 8-bit string.

This mismatch can go undetected by the compiler if the data came in as an untyped blob (like, say, a byte stream), or if got smuggled through some other channel that erased the original type.

One case where you can see this is if you call the `RegisterClassW` function to register a window class. The `W` at the end of `RegisterClassW` indicates that you want the Unicode version, and one of the things you provide is a window procedure, which will therefore be treated as a Unicode window procedure.

Your window procedure consequently needs to cast its `WPARAM` and `LPARAM` parameters into the Unicode versions of whatever message you are receiving.

And if it chooses to continue with default message handling, it needs to call the Unicode version of `DefWindowProc`: `DefWindowProcW`. That tells the system that the `WPARAM` and `LPARAM` parameters that you’re passing along should be treated as Unicode parameters, not ANSI parameters.

If you pass your Unicode messages to `DefWindowProcA`, then you’ll find that a lot of strings get truncated at their first character.

And that brings us back to today’s topic. What is this mysterious “G” program?

At the time the GDI+ library was written, it needed to support Windows 98, which had very limited support for Unicode. Therefore, it was compiled as ANSI and consequently used the ANSI versions of functions like `RegisterClass`, like `CreateWindow`, and `Def-`

`WindowProc` to create and manage its helper window. The lack of Unicode support in the helper window didn't really cause a problem because the window never displayed any UI and never processed any text. The window was there to do things like listen for `WM_SETTINGSCHANGE` messages so it knew when to invalidate its caches.

A few years ago, the GDI+ team did a little cleanup, and one of the things they did was get rid of support for Windows 98 and Windows Me. Because c'mon, Windows 98 and Windows Me went out of support over a decade ago.

For the most part, this meant simply recompiling GDI+ as a Unicode component rather than an ANSI component.

Except that the notification window procedure contained an explicit call to `DefWindowProcA`. Most character set mismatches would be caught by the compiler due to a type mismatch. But the character set dependency in `DefWindowProc` is not encoded in the parameter types. It's implicit in how you received the message. This mismatch went undetected by the compiler.

This mismatch also went undetected by testing because the notification window doesn't do any text processing. The title of the window got truncated from `"GDI+ Hook Window"` to simply `"G"`, but that title isn't used for anything, so the error was of no consequence. The window title is never shown to the user.

Except when it is.

As we saw last time, one mistake that programs make is taking responsibility for the GDI+ helper window but failing to pump messages. When the user tries to sign out or shut down, every window is sent a `WM_QUERYENDSESSION` message to let it know that the user is signing out, and this is your last chance to save your work and clean up.

If the application is not pumping messages, then the window will be flagged as not responding to the `WM_QUERYENDSESSION` message and consequently is blamed for preventing you from signing out (or shutting down).

When a program prevents you from signing out or shutting down, Windows looks for a visible window belonging to that program and uses that to represent it in the Blocked Shutdown Resolver (BSDR) screen. But if the program has no visible windows, then the BSDR will take *any* window belonging to the program, visible or not. And sometimes the invisible window that gets chosen is the one named "G".

That's why you end up with a message that implicates some mysterious program named "G" as the one that is preventing you from shutting down.

Once the problem was identified, the fix was simple: Change `DefWindowProcA` to plain old `DefWindowProc`, so that it follows the character set preference of the rest of the component, namely Unicode. This restores the full name of the window: `"GDI+ Hook Window"`.

But we took the fix one step further. Since it was clear that this window was a source of problems, we put the name of the underlying process in the window title, so that when it gets blamed for preventing you from signing out or shutting down, you have a better idea of what program is responsible.

The title of the window is now `"GDI+ Window (contoso.exe)"`. The fix is available in Windows 10 Insider Preview Build 19013.

> Have you ever tried shutting down, and seen a message saying "G" is preventing the operation, and thought "What is G? 🤔"
>
> We investigated this, and found...https://t.co/RBzWGnEt0F#WindowsInsiders pic.twitter.com/0sDJHkgIZv
>
> — Jen Gentleman 🌺 (@JenMsft) October 29, 2019

Raymond Chen

Follow