How ownership of the Windows clipboard is tracked in Win32

devblogs.microsoft.com/oldnewthing/20210526-00

May 26, 2021

Raymond Chen

In Win32, there is the concept of a clipboard *owner*. How does that work?

The intended rule is that the clipboard owner is the window that created the data that is currently on the clipboard.

The intended usage pattern for putting data on the clipboard is

- Call `OpenClipboard(hwnd)` passing the window that you want to become the new clipboard owner. (For the purpose of discussion, let's say that the *clipboard opener* is the window handle that was passed to the `OpenClipboard` function.)
- Call `EmptyClipboard()` to wipe out the previous contents.
- Call `SetClipboardData()` once for each piece of data you want to put onto the clipboard.
- Call `CloseClipboard()` to indicate that you are done setting the clipboard data.
- Congratulations, you are now the clipboard owner.

The clipboard owner receives a `WM_RENDERFORMAT` message when somebody requests data from the clipboard that had been set as delay-rendered. It also receives a `WM_RENDERALL-FORMATS` message as part of the window destruction sequence if it is still the owner of the clipboard at the time it is destroyed. Delay-rendering lets you defer the creation of complicated clipboard data until the point it is requested. And if nobody ever requests it, then you saved yourself a lot of work.

Bonus reading: What is the proper handling of `WM_RENDERFORMAT` and `WM_RENDERALL-FORMATS`?

The clipboard owner also receives a `WM_DESTROYCLIPBOARD` message when the clipboard contents are subsequently emptied. This tells the clipboard owner that it can free any memory that it had been using to produce the delay-rendered data.

The intended usage pattern for reading data from the clipboard is

- Call `OpenClipboard(hwnd)` passing the window that is reading the clipboard.
- Call `GetClipboardData()` to retrieve data from the clipboard.
- Call `CloseClipboard()` to indicate that you are done reading the clipboard data.

If everybody follows the rules, then it all works out.

Spoiler alert: Not everybody follows the rules.

Some programs open the clipboard with the intent of *adding* data to the clipboard, rather than replacing it with new data. Those programs call `OpenClipboard`, followed immediately by `SetClipboardData` without an intervening `EmptyClipboard`.

There was historically no enforcement that the caller of `SetClipboardData` is the clipboard owner. Back in the days of 16-bit Windows, the system assumed that applications were honest and played by the rules for the common good. (And for compatibility, these shenanigans need to continue to work, even though the world has become a more dangerous place.)

This "bonus clipboard data" scenario creates a bit of a problem, since there is only one clipboard owner (which you can interrogate by calling `GetClipboardOwner()`), but there are now *two* windows who collaborated to put data onto the clipboard. Who is the owner?

Ownership of the clipboard changes as follows:

- When `EmptyClipboard()` is called, the current clipboard opener becomes the clipboard owner.
- When the clipboard owner is destroyed, the clipboard owner resets to null.

Therefore, the clipboard owner can be summarized as "the window that most recently called `EmptyClipboard`, if it still exists."

Bonus chatter: There is also special handling of the case where somebody passes `NULL` to `OpenClipboard`, indicating that "nobody" is opening the clipboard.

Raymond Chen

Follow