The clipboard viewer linked list is no longer the responsibility of applications to maintain, unless they want to

devblogs.microsoft.com/oldnewthing/20110919-00

September 19, 2011

Raymond Chen

Commenter Nice Clipboard Manager (with drop->clipboard) wonders why Windows still uses a linked list to inform programs about clipboard modifications. If any clipboard viewer fails to maintain the chain, then some windows won't get informed of the change, and if a clipboard viewer creates a loop in the chain, an infinite loop results.

Well, sure, that's what happens if you use the old clipboard viewer chain. So don't use it. The old clipboard viewer chain remains for backward compatibility, but it's hardly the best way to monitor the clipboard. (This is another example of people asking for a feature that already exists.)

Instead of using the clipboard viewer chain, just add yourself as a clipboard format listener via `AddClipboardFormatListener`. Once you've done that, the system will post you a `WM_CLIPBOARDUPDATE` message when the contents of the clipboard have changed, and you can respond accordingly. When you're done, call `RemoveClipboardFormatListener`.

By using the clipboard format listener model, you let Windows worry about keeping track of all the people who are monitoring the clipboard, as Clipboarder Gadget suggested. (Mind you, Windows doesn't go so far as making each clipboard viewer think that it's the only viewer in the chain, because there may be applications which break the chain on purpose. Changing the chain behavior will break compatibility with those applications.)

Let's turn our scratch program into a clipboard format listener.

```
void
SniffClipboardContents(HWND hwnd)
{
 SetWindowText(hwnd, IsClipboardFormatAvailable(CF_TEXT)
             ? TEXT("Has text") : TEXT("No text"));
}
BOOL
OnCreate(HWND hwnd, LPCREATESTRUCT lpcs)
{
 SniffClipboardContents(hwnd); // set initial title
 return AddClipboardFormatListener(hwnd);
}
void
OnDestroy(HWND hwnd)
{
 RemoveClipboardFormatListener(hwnd);
 PostQuitMessage(0);
}
... add to window procedure ...
 case WM_CLIPBOARDUPDATE: SniffClipboardContents(hwnd); break;
```

And that's it. Much, much simpler than writing a clipboard viewer, and much more robust since you aren't dependent on other applications not screwing up.

There's another alternative to registering a clipboard listener and that's using the clipboard sequence number. The window manager increments the clipboard sequence number each time the contents of the clipboard change. You can compare the sequence number from two points in time to determine whether the contents of the clipboard have changed while you weren't looking.

Now you have a choice. Do you use the notification method (clipboard format listener) or the polling method (clipboard sequence number)? The notification method is recommended if you want to do something as soon as the clipboard contents change. On the other hand, the polling method is more suitable if you perform calculations based on the clipboard contents and cache the results, and then later you want to verify that your cached results are still valid.

For example, suppose you have a program with a Paste function, and pasting from the clipboard involves creating a complex data structure based on the clipboard contents. The user clicks Paste, you create your complex data structure, and insert it into the document. Your research discovers that a common operation is pasting the same contents several times. To optimize this, you want to cache the complex data structure so that if the user clicks Paste five times in a row, you only have to build the complex data structure the first time and you can just re-use it the other four times.

```
void DocumentWindow::OnPaste()
{
 if (m_CachedClipboardData == NULL ||
     GetClipboardSequenceNumber() != m_SequenceNumberInCache) {
  delete m_CachedClipboardData;
  m_SequenceNumberInCache = GetClipboardSequenceNumber();
  m_CachedClipboardData = CreateComplexDataFromClipboard();
 }
 if (m_CachedClipboardData) Paste(m_CachedClipboardData);
}
```

When the `OnPaste` method is called, we see if we have clipboard data cached from last time. If not, then clearly we need to create our complex data structure from the clipboard. If we do have clipboard data in our cache, we see if the clipboard sequence number has changed. If so, then the cached data is no longer valid and we have to throw it away and create it from scratch. But if we have cached data and the sequence number hasn't changed, then the cache is still valid and we can avoid calling `CreateComplexDataFromClipboard`.

The old clipboard viewer is like DDE: please feel free to stop using it.

Raymond Chen

Follow