How do I call SetTimer with a timer ID that is guaranteed not to conflict with any other timer ID?

devblogs.microsoft.com/oldnewthing/20150924-00

September 24, 2015

Raymond Chen

Suppose you want to create a timer for a window, but you don't know what timers that window already has running. You don't want your timer to overwrite any existing timer, so you need to find a unique ID for your timer. If this were for a window message, you could use `RegisterWindowMessage` , but there is no corresponding `RegisterTimerId` function. How do you generate a unique timer ID?

Every window gets to decide how to assign its own timer IDs, which are 32-bit integers on 32-bit Windows or 64-bit integers on 64-bit Windows. If you are an outsider, then you will have to negotiate with the window for a timer ID. There is no standard for this, so you will have to talk to whoever wrote the code for the window and come to some sort of agreement. "Okay, Bob, I'll let you have timers in the range 1000 to 1999."

But what if you don't personally know the person who wrote the code for the window, and the documentation for the window class does not specify how to negotiate a timer ID?

If you don't have a way to negotiate a timer ID for a window, then just create your own window and put your timer there. Since you created your own window, you control the window class, and you can set the rules for how timer IDs are assigned for that window class.

It's hardly a coincidence that the timer ID space is the same size as the address space. You can allocate some memory to track the timer (you probably have already done this), and use the address of that memory as the timer ID. Another common design is to use the handle to the object associated with the timer as the timer ID.

Raymond Chen

Follow