Modality, part 9: Setting the correct owner for modal UI, practical exam

devblogs.microsoft.com/oldnewthing/20110121-00

January 21, 2011

Raymond Chen

Here's a question that came from a customer. You can answer it yourself based on what you know about modal UI. (If you're kind of rusty on the topic, you can catch up here.) I've left in some irrelevant details just to make things interesting.

> Our program launches a helper program to display an Aero Wizard to guide the user through submitting a service request.
>
> ```
> theWiz.DoModal(hwndMainFrame);
> ```
>
> There are no `EnableWindow` calls leading up to or returning from this call. the `DoModal` handles things nicely.
>
> When the user clicks "Cancel" to cancel the service request, we use a TaskDialog so we can get the new look for our confirmation message box. The task dialog setup goes like this:
>
> ```
> TASKDIALOGCONFIG config = { 0 };
> config.cbSize = sizeof(config);
> config.hwndParent = hwndMainFrame;
> ```
>
> When the user clicks "Yes" to cancel, then another window instead of our frame becomes active.
>
> On a hunch, I replaced the task dialog with a Win32 message box
>
> ```
> MessageBox(hwndMainFrame, ...);
> ```
>
> and bingo, we get the correct behavior. When our wizard exits, the main frame receives focus.
>
> I believe that the "automatic" modal behavior that comes with `DoModal()` that takes care of disabling and reenabling the main frame is somehow getting short-circuited by using `TaskDialog` from inside our `PSM_QUERYCANCEL` message handler.
>
> Right now, we've switched to `MessageBox`, but we would much prefer to use the task dialog.

Although it’s not common, it is legal to have a window’s parent or owner belongs to another thread or process. But it definitely makes things a bit more tricky to manage because it attaches the input queues of the two threads, and you now have two threads coöperating to manage a single window hierarchy.

Is the cross-process window hierarchy a contributing factor to the problem, or is it just a red herring?

Raymond Chen

Follow