Are SetProp and RemoveProp thread-safe?

devblogs.microsoft.com/oldnewthing/20160303-00

March 3, 2016

Raymond Chen

A customer wanted to know whether the `SetProp` and `RemoveProp` functions are thread-safe with respect to a specific window and property. “It seems to work under the debugger, but that doesn’t prove that it always will work.”

It depends on what you mean by “thread-safe”. If you have two threads which call `SetProp` or `RemoveProp` without synchronization between them, then each individual call will be atomic. For example, two non-synchronized calls to `SetProp` will result in the final property being one value or the other, not a mix of the two values.

On the other hand, you cannot predict what order the operations were ultimately performed. All that is guaranteed is that the result will be consistent with *some* ordering of the operations.

For example, suppose you have a window and a property whose initial value is 1. One thread calls `SetProp(hwnd, 2)` and the other calls `RemoveProp(hwnd)`. There are two possible outcomes:

Outcome 1		**Outcome 2**	
	Property value is 1		Property value is 1
`SetProp(hwnd, 2);`	Changes property to 2	`RemoveProp(hwnd);`	Property is removed Returns 1 (removed value)
`RemoveProp(hwnd);`	Property is removed Returns 2 (removed value)	`SetProp(hwnd, 2);`	Changes property to 2
	Property is removed		Property value is 2

The customer seemed satisfied with this answer.

Note that only the individual call to `SetProp` or `RemoveProp` is atomic. If you make multiple calls in succession, you cannot guarantee that another thread won't sneak in between your calls and mess with the property.

Next time, a follow-up question that was never asked.

Raymond Chen

Follow