VirtualLock locks your memory into the working set, even if your threads are blocked

devblogs.microsoft.com/oldnewthing/20140207-00

February 7, 2014

Raymond Chen

Today, a correction to an earlier article on `VirtualLock`. When you lock memory with `VirtualLock`, it will remain locked even if all your threads are blocked. As noted in the *Follow-up* section at the end of the referenced article, the behavior of the operating system never changed. Virtually-locked pages were never unlocked in practice. What changed is that an implementation detail was elevated to contract. The intention when `VirtualLock` was originally designed was that virtually-locked pages were potentially unlockable if the application is not running. However, the memory manager folks never got around to implementing that part. At some point, they decided that they would abandon any future intention to to do and strengthened the contract accordingly. Mind you, `VirtualLock` does not guarantee that the same physical frame will always be assigned to the memory. The memory manager may reassign the memory to another physical frame in order to defragment memory so that it can allocate physically contiguous pages, primarily for I/O purposes, but occasionally to satisfy a large-page request. All it guarantees is that the memory will always be present.

The memory manager folks tell me that locked memory remains locked even if the application is suspended. But I don't know whether that's an implementation detail or contractual, so I wouldn't run around relying on it.

Raymond Chen

Follow