A 32-bit application can allocate more than 4GB of memory, and you don't need 64-bit Windows to do it

devblogs.microsoft.com/oldnewthing/20090706-00

July 6, 2009

Raymond Chen

Commenter Herb wondered how a 32-bit program running on 64-bit Windows can allocate more than 4GB of memory. Easy: The same way it allocates more than 4GB of memory on 32-bit Windows! Over a year before Herb asked the question, I had already answered it in the tediously boring two-week series on the myths surrounding the /3GB switch. Here's a page that shows how you can allocate more than 2GB of memory by using shared memory (which Win32 confusingly calls file mappings). That code fragment allocated 4GB of memory at one go, and then accessed it in pieces (because a 32-bit program can't map an entire 4GB memory block at one go). To allocate more, either make the number bigger in the call to `CreateFileMapping` or just call `CreateFileMapping` multiple times. The following week, I talked about how you can use AWE to allocate physical pages. Again, you can allocate as much memory as you like, but if you allocate enormous amounts of memory, you will probably not be able to map them all in at once. The claims of the program are true, but 64-bit Windows wasn't necessary for the program to accomplish what it claims. It's like Dumbo and the magic feather. "Dumbo can fly with the magic feather in his trunk." Well, yeah, but he didn't actually need the feather.

(On the other hand, 64-bit Windows certainly makes it more convenient to use more than 4GB of memory, since you can map the memory into your address space all at once and use normal pointers to access it.)

Raymond Chen

Follow