

Re: Memory Space

Source:

<http://www.tech-archive.net/Archive/VB/microsoft.public.vb.general.discussion/2008-03/msg01752.html>

- *From:* Lorin <Lorin@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 17 Mar 2008 19:12:01 -0700
-

But this link did not answer my question.
It does not talk about .dll or .exe activex.
I was wanting to see if using either of these would help allow me to use more memory to store data arrays.
I think the answer I am getting is that my main app can use most RAM.
I have not seen disk paging yet, but maybe I am missing something.
I really do not want to use disk paging since that will slow the overall process.

"dpb" wrote:

Lorin wrote:
....

Google is your friend only when you know what terminology to google for.

"memory usage" is pretty well known phrase and MSDN has all kinds of discussions on it...

Otherwise you are supposed to be my friend.

Only insofar as you help...

A first attempt w/ "Windows process memory limit" even in Google returned the following within the top 10...

<http://msdn2.microsoft.com/en-us/library/aa366778.aspx>

That would certainly lead you to the answer to any of the actual process limitations and so on in as much detail as you care/need to know.

....

Re: Memory Space

The statement that the OS will allocate space as needed does not seem to be true, otherwise, why would I get an out of memory error when I have 250G of HD and 4G of RAM and am only using several hundred meg of RAM in an array.

....

Does anyone know the real answer?

....

Not w/o hard code to look at, no...

A single array of a few hundred MB would rarely (if ever) be a problem if it were allocated initially at one time --- however, if you were to have already allocated a bunch of memory for other things or had worked your way up to this array size in pieces so that there were lots and lots of other moderate-sized arrays already allocated, there might not be sufficient contiguous memory left.

Alternatively, your application could be structured such that it is creating multiple copies of arrays and or other data structures (in a processing loop, perhaps) and you've not been careful to have released memory for those when done with them. (Or perhaps you're not done with them and you need to look at alternative ways to structure your algorithm so you don't attempt to hold all data and intermediate data in memory at once.)

So, no, nobody can tell you the real answer for your particular problem unless you can describe it more fully. Ideal would be a small test case that you could post that demonstrates the problem.

Failing that, a pared-down piece of code that shows the allocation sequence might be informative.
