Subject: Re: Solve memory problems
Posted by Paul Van Delst[1] on Wed, 14 Jan 2009 14:59:40 GMT
View Forum Message <> Reply to Message

Corinne wrote:
> hi guys,
>
> now i got lots of inputs! thanks so much! will check memtest.pro, as
> well as this pointer stuff and heap_gc. if it helps, why not?

Your unit test suite should include a test case that checks for valid pointers (and objects if appropriate) after your cleanup routines. If there are still non-null pointers floating around, your cleanup is incomplete. It's very simple to test for this stuff -- even in what I would refer to as "research grade" code.

I second David's incredulity at suggestions that generic garbage collection is an acceptable solution when you have memory issues. When the problems stem from structures/variables that the programmer created, it behooves said programmer to fix the problems. Otherwise it's just sloppy programming -- and you're learning bad habits that, when transferred to other languages, may not be so easily corrected.

It reminds me of the old chestnut about three properties of software: fast, good, cheap. Pick any two. :o)

cheers,

paulv

>
> @david: my 'images' contain normally up to 9 bands. normally i handle
> them separate (read in only exactly the band i need), as the file
> would be too big otherwise.
>
> best regards,
> corinne
>
>
> On Jan 14, 7:26 am, Craig Markwardt <cbmarkwa...@gmail.com> wrote:
>> On Jan 13, 10:33 am, David Fanning <n...@dfanning.com> wrote:
>>
>>> Jean H. writes:
>>>> As Carsten has mentioned, play with memtest.pro (from ITTVIS) to find
>>>> out what is happening. It could as well be a memory leak (you create a
>>>> pointer but don't destroy it). In this case, make a call to "heap_gc"
>>>> after your function.
>>> What!? What kind of advice is this!
>>> Uh, do NOT be making a call to HEAP_GC unless your program

>>> has completely and utterly failed and it is late Friday
>>> afternoon and you are at wit's end. Believe me when I tell
>>> you there are MUCH better ways to handle this!
>> Uh, like using any other high-level language that doesn't force you to
>> free your own variables?
>
