Subject: Re: Releasing temporary variables created with IDL_MakeTempArray()
Posted by chris_torrence@NOSPAM on Thu, 02 Apr 2015 01:46:58 GMT
View Forum Message <> Reply to Message

On Wednesday, April 1, 2015 at 6:38:58 PM UTC-6, David Grier wrote:
> Following up on my earlier post, I have some additional insights
> and a couple of questions for folks who really understand IDL internals.
>
> As originally written, my program allocated a steady stream of temporary variables
> using IDL_MakeTempArray() at a rate of 30 allocations per second. After about 20 minutes,
> the whole system would lock up hard (no mouse or keyboard, no ssh). This behavior
> was reproducible.
>
> Just doing the math suggests that lock-ups occur after 2^15 allocations.
>
> If, for the sake of argument, the number of allocated temporary variables were
> maintained internally with a signed short integer, failures might arise when that
> counter rolls over.
>
> Attempting to release the temporary variable with methods 1 and 2 as I originally
> proposed does not fix the problem. I even tried passing the temporary
> array back to C to delete the variable (using IDL_DelTemp) , but the variable no
> longer has its "temporary" flag set.
>
> What works is to release the variable explicitly at the end of the same
> C routine that creates it, just like in the documentation and example code.
> This isn't useful, however, because I want to work with the variable in IDL.
>
> This leaves me with a question: What is the "correct" way to allocate a temporary
> variable in C, pass the variable back to IDL, and then free the variable's resources
> in IDL?
>
> Also, am I right about the upper limit on the number of temporary variables that can
> be allocated in IDL?
>
> All the best,
>
> David
>
> P.S. I've fixed my immediate problem by allocating my temporary variable
> just once and feeding it back to my C library for updating (30 times per second).
> The C code is messier, but seems to run reliably.

Hi David,

I'm at home right now, so I can't check the C code to see if there is a hard limit on the # of temporary variables, but I'm a bit confused as to how you could get into this situation.

Normally, if you write a C routine for IDL, at the end of the routine, you return your result as a temporary variable (created say using IDL_MakeTempArray, MakeTempVector, Gettmp, etc.). As soon as you pass the variable back to IDL, you no longer own it, and you should never need to Deltmp it yourself. If your IDL code takes that function result and assigns it to a new variable, then that new variable will no longer be a temporary, and the temp will be immediately returned to the pool (the actual array data will get handed off to the new variable).

So, if your code looks like:

result = MyCFunction(...)

Then result is now a named variable containing the array data, and your temporary is long gone.

Doing this in a loop should work "forever". So, your code must look somewhat different?

-Chris