Subject: Re: Surprising Odds and Ends
Posted by Liam E. Gumley on Tue, 15 Aug 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Mark Hadfield wrote:
>
>> The bottom line is that HEAP_GC doesn't belong in
>> your code
>
> Agreed.
>
>> ..What you should
>> do is CATCH errors, and handle things in such a way that
>> you never have a need for HEAP_GC. And that is what
>> this chapter in my book is going to be about. :-)
>
> I'd better buy your book when it comes out :-) Until then, HEAP_GC and its
> friends RETALL, WIDGET_CONTROL,/RESET and CLOSE,/ALL are mighty handy from
> the command line when cleaning up after an error.
>
> This has been discussed on the group before, but I'm not keen on excessively
> enthusiastic error handling in code. If in doubt, stop where the error
> occurred and let the user sort it out!

My colleague Paul van Delst recently pointed (ha!) me to some features of the PTR_VALID function which help in identifying and reclaiming dangling references (i.e. heap variables for which no valid pointer exists):

(1) When no argument is specified, PTR_VALID returns a vector of pointers to all existing heap variables, regardless of whether a valid pointer exists for each heap variable,

(2) The PTR_VALID keyword CAST creates a new pointer to the heap variable index identified in the first function argument.

These debugging methods are a little more subtle than HEAP_GC (which I agree should *never* be used in IDL programs).

Cheers,
Liam.
http://cimss.ssec.wisc.edu/~gumley