Subject: Re: One RETALL is not enough
Posted by David Fanning on Fri, 02 Nov 2012 04:33:42 GMT
View Forum Message <> Reply to Message

Jim P writes:

> The need for multiple RETALLs in the context of objects usually means the ::CLEANUP for one or more objects lacks a CATCH block or an ON_ERROR, 2 and the ::CLEANUP fails to handle unexpected error conditions.
>
> I'm a big fan of adding CATCH statements that at the least throw HELP, /LAST_MESSAGE output, at least during the development of code.
>
> Since RETALL isn't the "happy path" for object destruction, there's the potential for circular heap references and other pathologies that were unanticipated by the class' original programmer.
>
> For the illustrative purposes, introduce an error into your own cleanup method in a test object, then execute a RETALL during execution from somewhere in code that uses an object of this class.
>
> The original posting appears to indicate that the contents of a HASH at the time the destructor was executed included some "bad" pointer references internally. Whether these are references to pointers that are used only internally by the hash or if they are references to "user mode" pointers that have been stored in the hash by user code is unclear. A repeatable case that can be sent to Exelis VIS (support@exelisvis.com) would be helpful if the issue can be tracked to
a pathology in the internal IDL class cleanup code.

It seems reasonable to assume Cleanup method problems can
cause this condition, but it doesn't really ring true with
my own experience. I've been doing quite a lot of object
programming in the past week, with very simple Cleanup
methods that I can't really imagine causing errors.

And yet I still find myself in situations where I can't
explain what is going on. Just today, for example, something
strange was happening, and I tried to set a breakpoint to
understand it. But, I could NOT put the breakpoint where
I wanted it. I would click in the gutter, and the breakpoint
would be placed several lines above. A RETALL and repeated
recompiles of the routine just didn't help at all. I presume
this is a symptom of what is going wrong.

In this case, a .Reset fixed whatever the problem is, but
I had a number of variables at the main IDL level that I
really didn't want to lose.

I wish it we repeatable enough to create a test case for.
But, it seems to happen at odd times, and for no apparent

reason, other than the normal errors that we spend out time debugging. It does seem finicky.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thue. ("Perhaps thos speakest truth.")