Subject: One RETALL is not enough
Posted by wlandsman on Fri, 26 Oct 2012 20:26:20 GMT
View Forum Message <> Reply to Message

While debugging a program, I've been getting error messages after a RETALL like the following:

IDL> retall
% Invalid pointer: <POINTER (<PtrHeapVar2858>)>.
% Execution halted at: XYZ_DEFAULTS::CLEANUP 456
IDL> retall
% Invalid pointer: <POINTER (<PtrHeapVar2578>)>.
% Execution halted at: XYZ_DEFAULTS::CLEANUP 456
IDL> retall
% Temporary variables are still checked out - cleaning up...
IDL> retall

So one RETALL is not enough to get a normal return , but if I give four RETALLs then there is enough of an extra "push" to give a normal return ;-) I first thought this was just a timing problem, and that the pointer cleanup wasn't complete at the time of the first RETALL, but it was complete by the time of the fourth RETALL. But the errors always appear in the same pattern as above, requiring 4 RETALLs no matter how much time I give. Any suggestions as to what is happening? Thanks, --Wayne

P.S. Line 456 where the first errors occurs is the following.

 IF OBJ_VALID(self.files.class.Revclasshash) THEN OBJ_DESTROY,
self.files.class.Revclasshash

where 'files' and 'class' are structures, and Revclasshash is an object

Subject: Re: One RETALL is not enough
Posted by Jim Pendleton on Fri, 02 Nov 2012 03:41:32 GMT
View Forum Message <> Reply to Message

On Friday, October 26, 2012 2:26:22 PM UTC-6, wlandsman wrote:
> While debugging a program, I've been getting error messages after a RETALL like the following:
>
>
>
> IDL> retall
>
> % Invalid pointer: <POINTER (<PtrHeapVar2858>)>.
>
> % Execution halted at: XYZ_DEFAULTS::CLEANUP 456
>
> IDL> retall

>
> % Invalid pointer: <POINTER (<PtrHeapVar2578>)>.
>
> % Execution halted at: XYZ_DEFAULTS::CLEANUP 456
>
> IDL> retall
>
> % Temporary variables are still checked out - cleaning up...
>
> IDL> retall
>
>
>
> So one RETALL is not enough to get a normal return , but if I give four RETALLs then there is enough of an extra "push" to give a normal return ;-) I first thought this was just a timing problem, and that the pointer cleanup wasn't complete at the time of the first RETALL, but it was complete by the time of the fourth RETALL. But the errors always appear in the same pattern as above, requiring 4 RETALLs no matter how much time I give. Any suggestions as to what is happening? Thanks, --Wayne
>
>
>
> P.S. Line 456 where the first errors occurs is the following.
>
>
>
> IF OBJ_VALID(self.files.class.Revclasshash) THEN OBJ_DESTROY, self.files.class.Revclasshash
>
>
>
> where 'files' and 'class' are structures, and Revclasshash is an object

At the risk of double-posting a reply...

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.

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.")

Subject: Re: One RETALL is not enough
Posted by Yngvar Larsen on Fri, 02 Nov 2012 08:13:05 GMT
View Forum Message <> Reply to Message

On Friday, 2 November 2012 05:33:42 UTC+1, David Fanning wrote:
> Jim P writes:
>
>> 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.

In IDL version >= 8.0, there should not really be any need for an explicit cleanup method anymore. (Unless you have used external OS resources: open sockets, open files, temp files that needs to be removed, etc.) That is sort of the point with an automatic garbage collector.

--
Yngvar
