Subject: Hard crash
Posted by Helder Marchetto on Thu, 18 Dec 2014 15:20:46 GMT
View Forum Message <> Reply to Message

Ok, I've done it again. After the dictionary() bug here is a new one.

*disclaimer: don't do this at home. Your pc might go in flames and your cat might bite your nose. I take no responsibility for any losses, damages or whatsoever.

Create a new file, name it "testObj.pro"
Put the following code in it:

```
pro testObj::crash, res
;This is bad:
print, res.test[self.refPos]
end

function testObj::init
self.refPos = 0l
return,1
end

pro testObj__Define, class
class = {testObj, refPos:0}
end

pro testObj
o = obj_new('testObj')
o.crash, -1l
end
```

Save the file and run it.
IDL window , in my case, closes. No warnings. Nothing. Bang.

I ran a few tests. The following options in the method crash, will not make it crash:

```
print, res.test[0]

void = self.refPos
print, res.test[void]
```

```
IDL> !version
{
   "ARCH": "x86_64",
   "OS": "Win32",
   "OS_FAMILY": "Windows",
   "OS_NAME": "MicrosoftWindows",
```

```
   "RELEASE": "8.4",
   "BUILD_DATE": "Sep272014",
   "MEMORY_BITS": 64,
   "FILE_OFFSET_BITS": 64
}
```

Any explanations? Also on previous builds (<8.4)?

Cheers,
Helder

Subject: Re: Hard crash
Posted by chris_torrence@NOSPAM on Thu, 18 Dec 2014 16:08:22 GMT
View Forum Message <> Reply to Message

On Thursday, December 18, 2014 8:20:49 AM UTC-7, Helder wrote:
> Ok, I've done it again. After the dictionary() bug here is a new one.
>
> *disclaimer: don't do this at home. Your pc might go in flames and your cat might bite your nose. I take no responsibility for any losses, damages or whatsoever.
>
>
> Create a new file, name it "testObj.pro"
> Put the following code in it:
>
> pro testObj::crash, res
> ;This is bad:
> print, res.test[self.refPos]
> end
>
> function testObj::init
> self.refPos = 0l
> return,1
> end
>
> pro testObj__Define, class
> class = {testObj, refPos:0}
> end
>
> pro testObj
> o = obj_new('testObj')
> o.crash, -1l
> end
>
> Save the file and run it.
> IDL window , in my case, closes. No warnings. Nothing. Bang.
>

> I ran a few tests. The following options in the method crash, will not make it crash:
>
> print, res.test[0]
>
> void = self.refPos
> print, res.test[void]
>
> IDL> !version
> {
>   "ARCH": "x86_64",
>   "OS": "Win32",
>   "OS_FAMILY": "Windows",
>   "OS_NAME": "MicrosoftWindows",
>   "RELEASE": "8.4",
>   "BUILD_DATE": "Sep272014",
>   "MEMORY_BITS": 64,
>   "FILE_OFFSET_BITS": 64
> }
>
> Any explanations? Also on previous builds (<8.4)?
>
> Cheers,
> Helder

Here's an even simpler reproduce:

IDL Version 8.4, Mac OS X (darwin x86_64 m64)
IDL> foo = {bar:0}
IDL> (-1).test[foo.bar]
Segmentation fault: 11

I'm at home today, but I'll log a bug when I get back to work, and try to fix it for the next service pack.

Thanks for catching it!

-Chris
Exelis

Subject: Re: Hard crash
Posted by Matt Haffner on Fri, 19 Dec 2014 21:34:15 GMT
View Forum Message <> Reply to Message

Chris,

There seems to be something "magic" about a variable name "test" and implied print. Perhaps related:

```
IDL> !version
{
   "ARCH": "x86_64",
   "OS": "darwin",
   "OS_FAMILY": "unix",
   "OS_NAME": "MacOSX",
   "RELEASE": "8.4",
   "BUILD_DATE": "Sep272014",
   "MEMORY_BITS": 64,
   "FILE_OFFSET_BITS": 64
}
IDL> test = 1
IDL> test
```

Crashes IDL.

- mh

On Thursday, December 18, 2014 10:08:24 AM UTC-6, Chris Torrence wrote:
> On Thursday, December 18, 2014 8:20:49 AM UTC-7, Helder wrote:
>> Ok, I've done it again. After the dictionary() bug here is a new one.
>>
>> *disclaimer: don't do this at home. Your pc might go in flames and your cat might bite your nose. I take no responsibility for any losses, damages or whatsoever.
>>
>>
>> Create a new file, name it "testObj.pro"
>> Put the following code in it:
>>

```
>> pro testObj::crash, res
>> ;This is bad:
>> print, res.test[self.refPos]
>> end
>>
>> function testObj::init
>> self.refPos = 0l
>> return,1
>> end
>>
>> pro testObj__Define, class
>> class = {testObj, refPos:0}
>> end
>>
>> pro testObj
>> o = obj_new('testObj')
>> o.crash, -1l
>> end
```

>>
>> Save the file and run it.
>> IDL window , in my case, closes. No warnings. Nothing. Bang.
>>
>> I ran a few tests. The following options in the method crash, will not make it crash:
>>
>> print, res.test[0]
>>
>> void = self.refPos
>> print, res.test[void]
>>
>> IDL> !version
>> {
>>   "ARCH": "x86_64",
>>   "OS": "Win32",
>>   "OS_FAMILY": "Windows",
>>   "OS_NAME": "MicrosoftWindows",
>>   "RELEASE": "8.4",
>>   "BUILD_DATE": "Sep272014",
>>   "MEMORY_BITS": 64,
>>   "FILE_OFFSET_BITS": 64
>> }
>>
>> Any explanations? Also on previous builds (<8.4)?
>>
>> Cheers,
>> Helder
>
> Here's an even simpler reproduce:
>
> IDL Version 8.4, Mac OS X (darwin x86_64 m64)
> IDL> foo = {bar:0}
> IDL> (-1).test[foo.bar]
> Segmentation fault: 11
>
> I'm at home today, but I'll log a bug when I get back to work, and try to fix it for the next service pack.
>
> Thanks for catching it!
>
> -Chris
> Exelis

Subject: Re: Hard crash
Posted by Jim Pendleton on Sat, 20 Dec 2014 04:52:06 GMT
View Forum Message <> Reply to Message

On Friday, December 19, 2014 2:34:17 PM UTC-7, Matt Haffner wrote:
> Chris,
>
> There seems to be something "magic" about a variable name "test" and implied print. Perhaps related:
>
> IDL> !version
> {
>   "ARCH": "x86_64",
>   "OS": "darwin",
>   "OS_FAMILY": "unix",
>   "OS_NAME": "MacOSX",
>   "RELEASE": "8.4",
>   "BUILD_DATE": "Sep272014",
>   "MEMORY_BITS": 64,
>   "FILE_OFFSET_BITS": 64
> }
> IDL> test = 1
> IDL> test
>
> Crashes IDL.
>
> - mh
>
> On Thursday, December 18, 2014 10:08:24 AM UTC-6, Chris Torrence wrote:
>> On Thursday, December 18, 2014 8:20:49 AM UTC-7, Helder wrote:
>>> Ok, I've done it again. After the dictionary() bug here is a new one.
>>>
>>> *disclaimer: don't do this at home. Your pc might go in flames and your cat might bite your nose. I take no responsibility for any losses, damages or whatsoever.
>>>
>>>
>>> Create a new file, name it "testObj.pro"
>>> Put the following code in it:
>>>
>>> pro testObj::crash, res
>>> ;This is bad:
>>> print, res.test[self.refPos]
>>> end
>>>
>>> function testObj::init
>>> self.refPos = 0l
>>> return,1
>>> end
>>>
>>> pro testObj__Define, class
>>> class = {testObj, refPos:0}
>>> end

>>>
>>> pro testObj
>>> o = obj_new('testObj')
>>> o.crash, -1l
>>> end
>>>
>>> Save the file and run it.
>>> IDL window , in my case, closes. No warnings. Nothing. Bang.
>>>
>>> I ran a few tests. The following options in the method crash, will not make it crash:
>>>
>>> print, res.test[0]
>>>
>>> void = self.refPos
>>> print, res.test[void]
>>>
>>> IDL> !version
>>> {
>>>   "ARCH": "x86_64",
>>>   "OS": "Win32",
>>>   "OS_FAMILY": "Windows",
>>>   "OS_NAME": "MicrosoftWindows",
>>>   "RELEASE": "8.4",
>>>   "BUILD_DATE": "Sep272014",
>>>   "MEMORY_BITS": 64,
>>>   "FILE_OFFSET_BITS": 64
>>> }
>>>
>>> Any explanations? Also on previous builds (<8.4)?
>>>
>>> Cheers,
>>> Helder
>>
>> Here's an even simpler reproduce:
>>
>> IDL Version 8.4, Mac OS X (darwin x86_64 m64)
>> IDL> foo = {bar:0}
>> IDL> (-1).test[foo.bar]
>> Segmentation fault: 11
>>
>> I'm at home today, but I'll log a bug when I get back to work, and try to fix it for the next service pack.
>>
>> Thanks for catching it!
>>
>> -Chris
>> Exelis

Matt,

Is there any chance you have a DLM in your path that includes a routine named "test", one that perhaps needs to be recompiled for IDL 8.4? I can't repeat this behavior on Win7-64 and IDL 8.4.

IDL> where(strcmp(routine_info(/system), 'TEST'))

Jim P.

Subject: Re: Hard crash
Posted by chris_torrence@NOSPAM on Sat, 20 Dec 2014 22:34:31 GMT
View Forum Message <> Reply to Message

On Friday, December 19, 2014 9:52:08 PM UTC-7, Jim P wrote:
> On Friday, December 19, 2014 2:34:17 PM UTC-7, Matt Haffner wrote:
>> Chris,
>>
>> There seems to be something "magic" about a variable name "test" and implied print. Perhaps related:
>>
>> IDL> !version
>> {
>>   "ARCH": "x86_64",
>>   "OS": "darwin",
>>   "OS_FAMILY": "unix",
>>   "OS_NAME": "MacOSX",
>>   "RELEASE": "8.4",
>>   "BUILD_DATE": "Sep272014",
>>   "MEMORY_BITS": 64,
>>   "FILE_OFFSET_BITS": 64
>> }
>> IDL> test = 1
>> IDL> test
>>
>> Crashes IDL.
>>
>> - mh
>>
>> On Thursday, December 18, 2014 10:08:24 AM UTC-6, Chris Torrence wrote:
>>> On Thursday, December 18, 2014 8:20:49 AM UTC-7, Helder wrote:
>>>> Ok, I've done it again. After the dictionary() bug here is a new one.
>>>>
>>>> *disclaimer: don't do this at home. Your pc might go in flames and your cat might bite your nose. I take no responsibility for any losses, damages or whatsoever.
>>>>
>>>>
>>>> Create a new file, name it "testObj.pro"

>>>> Put the following code in it:
>>>>
>>>> pro testObj::crash, res
>>>> ;This is bad:
>>>> print, res.test[self.refPos]
>>>> end
>>>>
>>>> function testObj::init
>>>> self.refPos = 0l
>>>> return,1
>>>> end
>>>>
>>>> pro testObj__Define, class
>>>> class = {testObj, refPos:0}
>>>> end
>>>>
>>>> pro testObj
>>>> o = obj_new('testObj')
>>>> o.crash, -1l
>>>> end
>>>>
>>>> Save the file and run it.
>>>> IDL window , in my case, closes. No warnings. Nothing. Bang.
>>>>
>>>> I ran a few tests. The following options in the method crash, will not make it crash:
>>>>
>>>> print, res.test[0]
>>>>
>>>> void = self.refPos
>>>> print, res.test[void]
>>>>
>>>> IDL> !version
>>>> {
>>>>   "ARCH": "x86_64",
>>>>   "OS": "Win32",
>>>>   "OS_FAMILY": "Windows",
>>>>   "OS_NAME": "MicrosoftWindows",
>>>>   "RELEASE": "8.4",
>>>>   "BUILD_DATE": "Sep272014",
>>>>   "MEMORY_BITS": 64,
>>>>   "FILE_OFFSET_BITS": 64
>>>> }
>>>>
>>>> Any explanations? Also on previous builds (<8.4)?
>>>>
>>>> Cheers,
>>>> Helder
>>>

>>> Here's an even simpler reproduce:
>>>
>>> IDL Version 8.4, Mac OS X (darwin x86_64 m64)
>>> IDL> foo = {bar:0}
>>> IDL> (-1).test[foo.bar]
>>> Segmentation fault: 11
>>>
>>> I'm at home today, but I'll log a bug when I get back to work, and try to fix it for the next service pack.
>>>
>>> Thanks for catching it!
>>>
>>> -Chris
>>> Exelis
>
> Matt,
>
> Is there any chance you have a DLM in your path that includes a routine named "test", one that perhaps needs to be recompiled for IDL 8.4? I can't repeat this behavior on Win7-64 and IDL 8.4.
>
> IDL> where(strcmp(routine_info(/system), 'TEST'))
>
> Jim P.

I agree with Jim. I think you have something strange going on with your IDL installation or path. I can't get it to crash either, and there's no reason that the two issues should be related.

Cheers,
Chris

Subject: Re: Hard crash
Posted by Matt Haffner on Thu, 08 Jan 2015 21:45:27 GMT
View Forum Message <> Reply to Message

(sorry for the delay--holidays...)

Well, I guess it's good it's just me--thought it was a pretty odd one. But I don't see any obvious rouge routine.

```
IDL> where(strmatch(routine_info(/system), '*test*', /fold_case))
       -1

IDL> .run test
% Error opening file. File: test
```

I run a pretty vanilla IDL other than Astro routines and my library, which has no DLM--only routines and macros written in IDL.

I'll try it on my other machines...

- mh

On Saturday, December 20, 2014 at 4:34:33 PM UTC-6, Chris Torrence wrote:
> On Friday, December 19, 2014 9:52:08 PM UTC-7, Jim P wrote:
>> On Friday, December 19, 2014 2:34:17 PM UTC-7, Matt Haffner wrote:
>>> Chris,
>>>
>>> There seems to be something "magic" about a variable name "test" and implied print. Perhaps related:
>>>
>>> IDL> !version
>>> {
>>>   "ARCH": "x86_64",
>>>   "OS": "darwin",
>>>   "OS_FAMILY": "unix",
>>>   "OS_NAME": "MacOSX",
>>>   "RELEASE": "8.4",
>>>   "BUILD_DATE": "Sep272014",
>>>   "MEMORY_BITS": 64,
>>>   "FILE_OFFSET_BITS": 64
>>> }
>>> IDL> test = 1
>>> IDL> test
>>>
>>> Crashes IDL.
>>>
>>> - mh
>>>
>>> On Thursday, December 18, 2014 10:08:24 AM UTC-6, Chris Torrence wrote:
>>>> On Thursday, December 18, 2014 8:20:49 AM UTC-7, Helder wrote:
>>>> > Ok, I've done it again. After the dictionary() bug here is a new one.
>>>> >
>>>> > *disclaimer: don't do this at home. Your pc might go in flames and your cat might bite your nose. I take no responsibility for any losses, damages or whatsoever.
>>>> >
>>>> >
>>>> > Create a new file, name it "testObj.pro"
>>>> > Put the following code in it:
>>>> >
>>>> > pro testObj::crash, res
>>>> > ;This is bad:
>>>> > print, res.test[self.refPos]
>>>> > end
>>>> >
>>>> > function testObj::init

>>>> > self.refPos = 0l
>>>> > return,1
>>>> > end
>>>> >
>>>> > pro testObj__Define, class
>>>> > class = {testObj, refPos:0}
>>>> > end
>>>> >
>>>> > pro testObj
>>>> > o = obj_new('testObj')
>>>> > o.crash, -1l
>>>> > end
>>>> >
>>>> > Save the file and run it.
>>>> > IDL window , in my case, closes. No warnings. Nothing. Bang.
>>>> >
>>>> > I ran a few tests. The following options in the method crash, will not make it crash:
>>>> >
>>>> > print, res.test[0]
>>>> >
>>>> > void = self.refPos
>>>> > print, res.test[void]
>>>> >
>>>> > IDL> !version
>>>> > {
>>>> >   "ARCH": "x86_64",
>>>> >   "OS": "Win32",
>>>> >   "OS_FAMILY": "Windows",
>>>> >   "OS_NAME": "MicrosoftWindows",
>>>> >   "RELEASE": "8.4",
>>>> >   "BUILD_DATE": "Sep272014",
>>>> >   "MEMORY_BITS": 64,
>>>> >   "FILE_OFFSET_BITS": 64
>>>> > }
>>>> >
>>>> > Any explanations? Also on previous builds (<8.4)?
>>>> >
>>>> > Cheers,
>>>> > Helder
>>>>
>>>> Here's an even simpler reproduce:
>>>>
>>>> IDL Version 8.4, Mac OS X (darwin x86_64 m64)
>>>> IDL> foo = {bar:0}
>>>> IDL> (-1).test[foo.bar]
>>>> Segmentation fault: 11
>>>>
>>>> I'm at home today, but I'll log a bug when I get back to work, and try to fix it for the next

service pack.
>>>>
>>>> Thanks for catching it!
>>>>
>>>> -Chris
>>>> Exelis
>>
>> Matt,
>>
>> Is there any chance you have a DLM in your path that includes a routine named "test", one that perhaps needs to be recompiled for IDL 8.4? I can't repeat this behavior on Win7-64 and IDL 8.4.
>>
>> IDL> where(strcmp(routine_info(/system), 'TEST'))
>>
>> Jim P.
>
> I agree with Jim. I think you have something strange going on with your IDL installation or path. I can't get it to crash either, and there's no reason that the two issues should be related.
>
> Cheers,
> Chris

Subject: Re: Hard crash
Posted by chris_torrence@NOSPAM on Thu, 08 Jan 2015 22:34:28 GMT
View Forum Message <> Reply to Message

Hey Matt,

That's weird. I still don't have any clue what could be happening for your bug. Regarding the original bug from Helder, it's fixed for IDL 8.4.1 - it now throws the following error:
% Subscripts are not allowed with variable attributes.

Cheers,
Chris

Subject: Re: Hard crash
Posted by Jim Pendleton on Fri, 09 Jan 2015 02:17:32 GMT
View Forum Message <> Reply to Message

On Thursday, January 8, 2015 at 3:34:31 PM UTC-7, Chris Torrence wrote:
> Hey Matt,
>
> That's weird. I still don't have any clue what could be happening for your bug. Regarding the original bug from Helder, it's fixed for IDL 8.4.1 - it now throws the following error:

> % Subscripts are not allowed with variable attributes.
>
> Cheers,
> Chris

Is it the same behavior when running from command line IDL and from the Workbench?

Are there any files named hs_err*.log created in the default directory when the IDL process exits?

Subject: Re: Hard crash
Posted by Matt Haffner on Fri, 09 Jan 2015 19:31:36 GMT
View Forum Message <> Reply to Message

Jim (& Chris),

No, the command line was fine. But I understand why now...

Those log files were very helpful--I wasn't aware of those before. The stack track in there made it clear it was indeed trying to open a user routine somewhere. dtrace helped me ID a 'test.sav' that was trying to be loaded from the IDLWorkspace84/Default directory, which is why it was only crashing in eclipse--that directory isn't added to the command line IDL path by default.

Trying the same thing from the command line with that file in the path does then give the same result, with a bit more error output:

IDL> test
% Wrong number of tags defined for structure: LIST.
% Structure not restored due to conflict with existing definition: LIST.
Segmentation fault: 11

The file contained an old save of an object that does contain some LISTs buried in it's property hierarchy.

Restoring the file does not crash IDL:

IDL> restore,'./test.sav'
% Wrong number of tags defined for structure: LIST.
% RESTORE: Structure not restored due to conflict with existing definition: LIST.
% Wrong number of tags defined for structure: LIST.
% RESTORE: Structure not restored due to conflict with existing definition: LIST.
% Wrong number of tags defined for structure: LIST.
% RESTORE: Structure not restored due to conflict with existing definition: LIST.

Looks like the save was in IDL 8.2.3. Did LIST's data structure evolve since then?

None of this is critical for me to restore/retain... mostly morbid curiosity at this point. But let me know if it's useful to poke around further to characterize a bug.

- Matt

On Thursday, January 8, 2015 at 8:17:34 PM UTC-6, Jim P wrote:
> On Thursday, January 8, 2015 at 3:34:31 PM UTC-7, Chris Torrence wrote:
>> Hey Matt,
>>
>> That's weird. I still don't have any clue what could be happening for your bug. Regarding the original bug from Helder, it's fixed for IDL 8.4.1 - it now throws the following error:
>> % Subscripts are not allowed with variable attributes.
>>
>> Cheers,
>> Chris
>
> Is it the same behavior when running from command line IDL and from the Workbench?
>
> Are there any files named hs_err*.log created in the default directory when the IDL process exits?

Subject: Re: Hard crash
Posted by Jim Pendleton on Sat, 10 Jan 2015 01:13:53 GMT
View Forum Message <> Reply to Message

On Friday, January 9, 2015 at 12:31:39 PM UTC-7, Matt Haffner wrote:
> Jim (& Chris),
>
> No, the command line was fine. But I understand why now...
>
> Those log files were very helpful--I wasn't aware of those before. The stack track in there made it clear it was indeed trying to open a user routine somewhere. dtrace helped me ID a 'test.sav' that was trying to be loaded from the IDLWorkspace84/Default directory, which is why it was only crashing in eclipse--that directory isn't added to the command line IDL path by default.
>
> Trying the same thing from the command line with that file in the path does then give the same result, with a bit more error output:
>
> IDL> test
> % Wrong number of tags defined for structure: LIST.
> % Structure not restored due to conflict with existing definition: LIST.
> Segmentation fault: 11
>
> The file contained an old save of an object that does contain some LISTs buried in it's property hierarchy.
>
> Restoring the file does not crash IDL:
>
> IDL> restore,'./test.sav'

> % Wrong number of tags defined for structure: LIST.
> % RESTORE: Structure not restored due to conflict with existing definition: LIST.
> % Wrong number of tags defined for structure: LIST.
> % RESTORE: Structure not restored due to conflict with existing definition: LIST.
> % Wrong number of tags defined for structure: LIST.
> % RESTORE: Structure not restored due to conflict with existing definition: LIST.
>
> Looks like the save was in IDL 8.2.3. Did LIST's data structure evolve since then?
>
> None of this is critical for me to restore/retain... mostly morbid curiosity at this point. But let me know if it's useful to poke around further to characterize a bug.
>
> - Matt
>
> On Thursday, January 8, 2015 at 8:17:34 PM UTC-6, Jim P wrote:
>> On Thursday, January 8, 2015 at 3:34:31 PM UTC-7, Chris Torrence wrote:
>>> Hey Matt,
>>>
>>> That's weird. I still don't have any clue what could be happening for your bug. Regarding the original bug from Helder, it's fixed for IDL 8.4.1 - it now throws the following error:
>>> % Subscripts are not allowed with variable attributes.
>>>
>>> Cheers,
>>> Chris
>>
>> Is it the same behavior when running from command line IDL and from the Workbench?
>>
>> Are there any files named hs_err*.log created in the default directory when the IDL process exits?

I'll let Chris weigh in on the how and why, but yes the LIST data type definition has been updated. There will be an IDL Data Point blog post at the end of the month which talks about some of the unwanted side effects of this and how to avoid them.

A quick summary is that you should never use LIST(), HASH(), etc., in structure or class definition files when defining structure tags, and to always use the keyword SKIP=['LIST', 'HASH'], etc., when executing RESOLVE_ALL to create a new p-code SAVE file.

When *manually* restoring a SAVE file from an earlier version, you have the option of using the /RELAXED_STRUCTURE_ASSIGNMENT keyword to RESTORE. But if the SAVE file is loaded automatically, that's not the case.

Jim P.

Subject: Re: Hard crash
Posted by David Fanning on Fri, 30 Jan 2015 17:25:40 GMT

View Forum Message <> Reply to Message

Jim P writes:

> A new blog explaining this effect is now posted at
> http://tinyurl.com/ocohc6n

Hi, Jim, I read this article this morning. Talk about a recipe for disaster! :-)

I was going to offer to host a copy of this article on my web page, if you would like me to. I think a few more people might see it and benefit greatly from the information. That DataPoint blog is great, but really hard for some of us to get to.

Cheers,

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

Subject: Re: Hard crash
Posted by Lajos Foldy on Fri, 30 Jan 2015 18:36:56 GMT
View Forum Message <> Reply to Message

On Friday, January 30, 2015 at 5:37:28 PM UTC+1, Jim P wrote:

> A new blog explaining this effect is now posted at
> http://tinyurl.com/ocohc6n

I think the problem is more general. Save files created after any use of LIST or HASH will include the unwanted routines. For example:

```
IDL> l=list()
IDL> save,/routines,/verbose,file='test.sav'
% SAVE: Portable (XDR) SAVE/RESTORE file.
% SAVE: Saved procedure: COLLECTION::CLEANUP.
% SAVE: Saved procedure: COLLECTION::_OVERLOADBRACKETSLEFTSIDEARRAY.
% SAVE: Saved procedure: COLLECTION__DEFINE.
...
```

regards,
Lajos

Subject: Re: Hard crash
Posted by Heinz Stege on Fri, 30 Jan 2015 21:15:24 GMT
View Forum Message <> Reply to Message

On Fri, 30 Jan 2015 10:25:40 -0700, David Fanning wrote:

> Jim P writes:
>
>> A new blog explaining this effect is now posted at
>> http://tinyurl.com/ocohc6n
>
> Hi, Jim, I read this article this morning. Talk about a recipe for
> disaster! :-)
>
> I was going to offer to host a copy of this article on my web page, if
> you would like me to. I think a few more people might see it and benefit
> greatly from the information. That DataPoint blog is great, but really
> hard for some of us to get to.
>
That would be very helpful. The link from tinyurl leads me to
http://www.exelisvis.com/Company/PressRoom/Blogs/IDLDataPoin
tDetail/TabId/902/ArtMID/2926/ArticleID/14277/Problems-Assig
ning-LIST-HASH-etc-to-Class-or-Structure-Tags.aspx
and there I get the information:
"Article does not exist or Permission Denied."

Heinz

Subject: Re: Hard crash
Posted by Lajos Foldy on Fri, 30 Jan 2015 21:25:45 GMT
View Forum Message <> Reply to Message

On Friday, January 30, 2015 at 10:15:18 PM UTC+1, Heinz Stege wrote:
> On Fri, 30 Jan 2015 10:25:40 -0700, David Fanning wrote:
>
>> Jim P writes:
>>
>>> A new blog explaining this effect is now posted at
>>> http://tinyurl.com/ocohc6n
>>
>> Hi, Jim, I read this article this morning. Talk about a recipe for
>> disaster! :-)
>>
>> I was going to offer to host a copy of this article on my web page, if
>> you would like me to. I think a few more people might see it and benefit
>> greatly from the information. That DataPoint blog is great, but really
>> hard for some of us to get to.

>>
> That would be very helpful. The link from tinyurl leads me to
> http://www.exelisvis.com/Company/PressRoom/Blogs/IDLDataPoin
tDetail/TabId/902/ArtMID/2926/ArticleID/14277/Problems-Assig
ning-LIST-HASH-etc-to-Class-or-Structure-Tags.aspx
> and there I get the information:
> "Article does not exist or Permission Denied."
>
> Heinz

I can confirm that it was there two hours ago. Now it is here:

http://www.exelisvis.com/Company/PressRoom/Blogs/TabId/836/A
rtMID/2928/ArticleID/14277/Problems-Assigning-LIST-HASH-etc- to-Class-or-Structure-Tags.aspx

Read quickly before it changes its address again :-)

regards,
Lajos

Subject: Re: Hard crash
Posted by Heinz Stege on Fri, 30 Jan 2015 23:39:57 GMT
View Forum Message <> Reply to Message

On Fri, 30 Jan 2015 13:25:45 -0800 (PST), fawltylanguage@gmail.com
wrote:

> I can confirm that it was there two hours ago. Now it is here:
>
> http://www.exelisvis.com/Company/PressRoom/Blogs/TabId/836/A
rtMID/2928/ArticleID/14277/Problems-Assigning-LIST-HASH-etc- to-Class-or-Structure-Tags.aspx
>
> Read quickly before it changes its address again :-)
>
Thank you Lajos. The second link is working for me to. :-)

However I wonder, if the most important point isn't that in the
comment: "BUT you should use the SKIP keyword to RESOLVE_ALL before
creating your SAVE file to ensure the "hash.sav" file is NOT included
in your build."

I don't use lists and hashes frequently, but does it really make a
change to use OBJ_NEW() instead of LIST() in the class definition?
Don't misunderstand me, it is always a good idea to be clearly in
coding. So it may be more readable to use OBJ_NEW().

But what is the difference? When I define two structures

```
   s1={a:list()}
   s2={a:obj_new()}
```
IDL's help command
```
   help,s1,s2,/str
```
prints:
```
** Structure <1606fea8>, 1 tags, length=4, data length=4, refs=1:
   A               OBJREF    <ObjHeapVar1(LIST)>
** Structure <11587c78>, 1 tags, length=4, data length=4, refs=1:
   A               OBJREF    <NullObject>
```
There are object references in both structures s1 and s2. The first one s1 is already allocated in the heap, the second one s2 is a null object. But somewhere in the code I have to make it a list:
```
   s2.a=list()
```
And then the help command says, that s2 is equal s1:
```
** Structure <11587c78>, 1 tags, length=4, data length=4, refs=1:
   A               OBJREF    <ObjHeapVar2(LIST)>
```
Is there something wrong in my understanding?

Cheers, Heinz

Subject: Re: Hard crash
Posted by Jim Pendleton on Sat, 31 Jan 2015 01:24:33 GMT
View Forum Message <> Reply to Message

On Friday, January 30, 2015 at 4:39:51 PM UTC-7, Heinz Stege wrote:
> On Fri, 30 Jan 2015 13:25:45 -0800 (PST), fawltylanguage@gmail.com
> wrote:
>
>> I can confirm that it was there two hours ago. Now it is here:
>>
>> http://www.exelisvis.com/Company/PressRoom/Blogs/TabId/836/ArtMID/2928/ArticleID/14277/Problems-Assigning-LIST-HASH-etc- to-Class-or-Structure-Tags.aspx
>>
>> Read quickly before it changes its address again :-)
>>
> Thank you Lajos. The second link is working for me to. :-)
>
> However I wonder, if the most important point isn't that in the
> comment: "BUT you should use the SKIP keyword to RESOLVE_ALL before
> creating your SAVE file to ensure the "hash.sav" file is NOT included
> in your build."
>
> I don't use lists and hashes frequently, but does it really make a
> change to use OBJ_NEW() instead of LIST() in the class definition?
> Don't misunderstand me, it is always a good idea to be clearly in
> coding. So it may be more readable to use OBJ_NEW().
>

> But what is the difference? When I define two structures
> s1={a:list()}
> s2={a:obj_new()}
> IDL's help command
> help,s1,s2,/str
> prints:
> ** Structure <1606fea8>, 1 tags, length=4, data length=4, refs=1:
> A OBJREF <ObjHeapVar1(LIST)>
> ** Structure <11587c78>, 1 tags, length=4, data length=4, refs=1:
> A OBJREF <NullObject>
> There are object references in both structures s1 and s2. The first
> one s1 is already allocated in the heap, the second one s2 is a null
> object. But somewhere in the code I have to make it a list:
> s2.a=list()
> And then the help command says, that s2 is equal s1:
> ** Structure <11587c78>, 1 tags, length=4, data length=4, refs=1:
> A OBJREF <ObjHeapVar2(LIST)>
> Is there something wrong in my understanding?
>
> Cheers, Heinz

Sorry about the jumping links, folks.

I have no control over that side of things. It may be that if a comment is added after publication, the original URL changes. (I noticed someone recently purchased the idldatapoint dot com domain and that is now pushed to the top of the Google search hierarchy, so watch out if you access IDL Data Point via Google. Who knows what NSFW junk could end up there.)

With respect to your question, Heinz, the problem isn't with respect to what happens at run time in your example. If you never create a SAVE file from your compiled code, and you never redistribute your compiled code to other users of IDL running different versions of the product, you won't run into this problem.

It may still be relevant if you receive a SAVE file from a colleague and your IDL session starts crashing. This could happen if the SAVE file was built on an older version of IDL.

As we've seen, this is a possibility that has been reported in the IDL user community.

It's a feral catastrophe, and not simply a gedankenexperiment.

The key to understanding the problem is that the IDL compiler *must* restore the hash.sav at compile time if a structure tag is defined as LIST(), HASH(), etc. At that point, all is lost* with respect to IDL "forgetting" about the compiled definitions.

In contrast, a variable assignment call in an executable statement can be deferred until run time, such as "self.l = list()", which is where the COMPILE_OPT IDL2 comes in. In combination with RESOLVE_ALL, SKIP=['LIST','HASH', etc.], the contents of hash.sav will be prevented from being included in your own SAVE file, which should be your goal.

Jim P.**

*When I say "all is lost", it's not really. If you're a glutton for punishment you can, in fact, "edit" your SAVE file contents via the IDL_SaveFile class. This is an exercise left for the reader.

**I still work for Exelis.

Subject: Re: Hard crash
Posted by markb77 on Mon, 02 Feb 2015 15:09:10 GMT
View Forum Message <> Reply to Message

Wow - this is an important point! I have just gone through a large codebase and removed all of the references to list() and hash() from my class definitions.

One question: I also have a large collection of structure definition files, which are part of the project. Should I also remove list() and hash() from my structure definitions? This doesn't seem to make sense, however, since when the structure is created it will be initialized automatically with an empty list or hash.

thanks.

Subject: Re: Hard crash
Posted by Heinz Stege on Mon, 02 Feb 2015 21:40:15 GMT
View Forum Message <> Reply to Message

On Fri, 30 Jan 2015 17:24:33 -0800 (PST), Jim P wrote:

> The key to understanding the problem is that the IDL compiler *must* restore the hash.sav at compile time if a structure tag is defined as LIST(), HASH(), etc. At that point, all is lost* with respect to IDL "forgetting" about the compiled definitions.
>
Aah, that brings some light into the darkness. IDL 8.4 restores the hash.sav file when compiling a routine with LIST() as a structure tag? I didn't know this, since I'm still working with Version 8.0.1.* My version is different. It does _not_ restore hash.sav or list.sav when compiling a routine with a LIST() tag in the class structure definition.

Do you think, that it may be interesting for the news group, why this change was done?

Cheers, Heinz

* My company does not have the money to upgrade. :-(

Subject: Re: Hard crash
Posted by Dick Jackson on Tue, 10 Feb 2015 02:18:11 GMT
View Forum Message <> Reply to Message

Heinz Stege wrote, On 2015-02-02, 1:40pm:
> On Fri, 30 Jan 2015 17:24:33 -0800 (PST), Jim P wrote:
>
>> The key to understanding the problem is that the IDL compiler *must* restore the hash.sav at compile time if a structure tag is defined as LIST(), HASH(), etc. At that point, all is lost* with respect to IDL "forgetting" about the compiled definitions.
>>
> Aah, that brings some light into the darkness. IDL 8.4 restores the
> hash.sav file when compiling a routine with LIST() as a structure tag?
> I didn't know this, since I'm still working with Version 8.0.1.* My
> version is different. It does _not_ restore hash.sav or list.sav when
> compiling a routine with a LIST() tag in the class structure
> definition.
>
> Do you think, that it may be interesting for the news group, why this
> change was done?
>
> Cheers, Heinz
>
> * My company does not have the money to upgrade. :-(

I think I was caught by this issue, hoping to compile a .sav in 8.4 for a client to run in 8.3. Some problems were
relieved by:

1. In my project's Build, replacing the standard RESOLVE_ALL with post-process command:
   RESOLVE_ALL, skip=['list','hash','orderedhash']

2. Avoiding the use of myList = List(myArray, /EXTRACT)

But I'm seeing another problem I can't seem to avoid, where I'm adding an item to a list, and I get:
   IDL_CONTAINER::ADD: Object reference type required in this context: INDEXITEM.
   (INDEXITEM was my variable that I was Adding to the list)

I confirm that neither IDL 8.3 nor 8.4 actually require this of a List() object.

I tried adding 'idl_container' to the "skip" list in RESOLVE_ALL, to no avail.

Any suggestions of how to get around this?

--

Cheers,
-Dick

Dick Jackson Software Consulting Inc.
Victoria, BC, Canada
www.d-jackson.com

Subject: Re: Hard crash
Posted by Jim Pendleton on Tue, 10 Feb 2015 03:16:26 GMT
View Forum Message <> Reply to Message

On Monday, February 9, 2015 at 7:18:13 PM UTC-7, Dick Jackson wrote:
> Heinz Stege wrote, On 2015-02-02, 1:40pm:
>> On Fri, 30 Jan 2015 17:24:33 -0800 (PST), Jim P wrote:
>>
>>> The key to understanding the problem is that the IDL compiler *must* restore the hash.sav at compile time if a structure tag is defined as LIST(), HASH(), etc. At that point, all is lost* with respect to IDL "forgetting" about the compiled definitions.
>>>
>> Aah, that brings some light into the darkness. IDL 8.4 restores the
>> hash.sav file when compiling a routine with LIST() as a structure tag?
>> I didn't know this, since I'm still working with Version 8.0.1.* My
>> version is different. It does _not_ restore hash.sav or list.sav when
>> compiling a routine with a LIST() tag in the class structure
>> definition.
>>
>> Do you think, that it may be interesting for the news group, why this
>> change was done?
>>
>> Cheers, Heinz
>>
>> * My company does not have the money to upgrade. :-(
>
> I think I was caught by this issue, hoping to compile a .sav in 8.4 for a client to run in 8.3. Some problems were
> relieved by:
>
> 1. In my project's Build, replacing the standard RESOLVE_ALL with post-process command:
> RESOLVE_ALL, skip=['list','hash','orderedhash']
>
> 2. Avoiding the use of myList = List(myArray, /EXTRACT)
>
> But I'm seeing another problem I can't seem to avoid, where I'm adding an item to a list, and I get:
> IDL_CONTAINER::ADD: Object reference type required in this context: INDEXITEM.
> (INDEXITEM was my variable that I was Adding to the list)
>
> I confirm that neither IDL 8.3 nor 8.4 actually require this of a List() object.
>
> I tried adding 'idl_container' to the "skip" list in RESOLVE_ALL, to no avail.

>
> Any suggestions of how to get around this?
>
> --
>
> Cheers,
> -Dick
>
> Dick Jackson Software Consulting Inc.
> Victoria, BC, Canada
> www.d-jackson.com

Hi Dick,

Okay, this is a tricky one. The bottom line is you'd probably be better off compiling your app in 8.3, if you can. We don't really have forward compatibility in IDL, though you might get lucky on occasion.

There was considerable work performed in 8.4 to make it more efficient so some code formerly in .pro was moved down to C.

Compare and contrast the output from routine_info('list::add', /source) between the two versions. You'll see that in 8.4 it "doesn't exist". Instead search through the output from routine_info(/system).

If you perform a RESTORE, /VERBOSE on your compiled 8.4 file in 8.3, are you sure there are no references to the routines from the lib\datatypes .sav files? If there are references in your .sav file, then there's probably a stray list() or hash() in one or more routines, without a needed compile_opt idl2 (or strictarr).

Jim P.
"I presently work for Exelis"