Subject: Re: Scope_VarFetch Bug or Feature?
Posted by Robert Barnett on Tue, 14 Feb 2006 22:22:25 GMT
View Forum Message <> Reply to Message

Dear David,

I think that the trick is to test for the number of elements returned
from Scope_VarFetch before you assign it to a variable.

```
;*****************************************************
PRO test01

names = Scope_Varname(Level=1,Count=varcount)
FOR j=0,varcount-1 DO BEGIN
   s = n_elements(Scope_VarFetch(names[j], Level=1, /Enter))
   Print, 'Fetching variable ' + names[j] + ' with n_elements ' +
string(s)
   if (s gt 0) then begin
      mainvar = Scope_VarFetch(names[j], Level=1)
      Help, mainvar
   endif
ENDFOR

END
;*****************************************************
```

Cheers
Robbie

Subject: Re: Scope_VarFetch Bug or Feature?
Posted by Foldy Lajos on Tue, 14 Feb 2006 22:23:00 GMT
View Forum Message <> Reply to Message

Hi David,

well, you can check it first:

```
help, out=s, (Scope_VarFetch(names[j], Level=1, /Enter))
if strpos(s, 'UNDEFINED =') gt 0 then print, 'Undefined'
```

(you can't retrieve an undefined variable, but you can get a
reference to it :-)

And I think the ENTER keyword works, it just creates an undefined
variable :-) You can create a defined variable by

```
(Scope_VarFetch('something_new', Level=1, /Enter))=0
```

regards,
lajos

On Tue, 14 Feb 2006, David Fanning wrote:

```
> Folks,
>
> Here is a short test program and main-level program:
>
> ;***************************************************
> PRO test
>
>   names = Scope_Varname(Level=1,Count=varcount)
>   FOR j=0,varcount-1 DO BEGIN
>     Print, 'Fetching variable ' + names[j] + '...'
>     mainvar = Scope_VarFetch(names[j], Level=1, /Enter)
>     Help, mainvar
>   ENDFOR
>
> END
>
> a = 5
> b = c
> test
>
> END
> ;***************************************************
>
> Compile and run the main-level program. Yes, it will
> cause an error. Don't worry about it. :-)
>
> When the error occurs, you will have two variables at
> the main level (LEVEL=1 in SCOPE_ parlance), a and b.
> The variable a is equal to 5 and b is undefined.
>
> Now, just type "test" and the test program runs.
> It will find both variables at the main level.
> But, as far as I can tell there is no way to learn
> anything more about the variables unless I retrieve
> them. BUT, I can't retrieve an undefined variable.
> It causes an error. :-(
>
> (According to the documentation the ENTER keyword should
> cause a variable with the name I am trying to fetch (b, in
> this case) to be created at the main-level, but this
```

> doesn't seem to work either. At least I still get the error.
> This DOES seem like a bug to me.)
>
> This is all causing me some heartburn, because I need to
> retrieve variables at a particular level and save them.
> All works well unless I try to retrieve undefined variables,
> and in this particular application, I cannot assume there
> won't be undefined variables.
>
> Does this seem like a bug to you, or a feature? Is the
> only alternative to CATCH the error?
>
> Cheers,
>
> David
>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
>

Subject: Re: Scope_VarFetch Bug or Feature?
Posted by David Fanning on Tue, 14 Feb 2006 22:33:04 GMT
View Forum Message <> Reply to Message

=?ISO-8859-2?Q?F=D6LDY_Lajos?= writes:

> well, you can check it first:
>
> help, out=s, (Scope_VarFetch(names[j], Level=1, /Enter))
> if strpos(s, 'UNDEFINED =') gt 0 then print, 'Undefined'
>
> (you can't retrieve an undefined variable, but you can get a
> reference to it :-)
>
> And I think the ENTER keyword works, it just creates an undefined
> variable :-) You can create a defined variable by
>
> (Scope_VarFetch('something_new', Level=1, /Enter))=0

Ah, now all that screwy parentheses stuff I remember from
a couple of months ago when I first starting looking at
SCOPE_VARFETCH is starting to make some sense to me! :-)

Thanks guys.

Cheers,

David

P.S. Let's just say we were burning about $1000/hour with
a room full of Ph.D. physicists trying to figure it out from
the documentation earlier today, to no avail. :-(

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Scope_VarFetch Bug or Feature?
Posted by Mark Hadfield on Tue, 14 Feb 2006 22:46:29 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> P.S. Let's just say we were burning about $1000/hour with
> a room full of Ph.D. physicists trying to figure it out from
> the documentation earlier today, to no avail. :-(

Must be a big room!

--
Mark Hadfield "Kei puwaha te tai nei, Hoea tahi tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: Scope_VarFetch Bug or Feature?
Posted by David Fanning on Tue, 14 Feb 2006 22:51:15 GMT
View Forum Message <> Reply to Message

Mark Hadfield writes:

> David Fanning wrote:
>> P.S. Let's just say we were burning about $1000/hour with
>> a room full of Ph.D. physicists trying to figure it out from
>> the documentation earlier today, to no avail. :-(
>
> Must be a big room!

Well, it's government money. In this country it's easy to
raise more just by lowering taxes. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Scope_VarFetch Bug or Feature?
Posted by Craig Markwardt on Wed, 15 Feb 2006 00:31:38 GMT
View Forum Message <> Reply to Message

"Robbie Barnett" <retsil@zipworld.com.au> writes:
> Dear David,
>
> I think that the trick is to test for the number of elements returned
> from Scope_VarFetch before you assign it to a variable.

This is definitely how I did it in the past. Except I used
ROUTINE_NAMES(), and I used SIZE() instead of N_ELEMENTS().

But other than that, totally the same :-)
Craig

Subject: Re: Scope_VarFetch Bug or Feature?
Posted by retsil on Wed, 15 Feb 2006 01:52:05 GMT
View Forum Message <> Reply to Message

I think that Scope_VarFetch certainly opens up a lot of possibilities.
At first I thought that it was just a hack tool for IDL, but it
actually has some legitamite uses.

For example, I use Scope_VarFetch to bring DICOM images and object
properties into the context of a procedure.
https://www.rsinc.com/codebank/search.asp?FID=397

Robbie

Subject: Re: Scope_VarFetch Bug or Feature?
Posted by David Fanning on Wed, 15 Feb 2006 02:00:21 GMT
View Forum Message <> Reply to Message

retsil@iinet.net.au writes:

> I think that Scope_VarFetch certainly opens up a lot of possibilities.
> At first I thought that it was just a hack tool for IDL, but it
> actually has some legitamite uses.

I can't really believe (or discuss) what I am doing with
it. But, yes, I am very, very impressed. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Scope_VarFetch Bug or Feature?
Posted by JD Smith on Thu, 16 Feb 2006 22:41:33 GMT
View Forum Message <> Reply to Message

On Tue, 14 Feb 2006 19:00:21 -0700, David Fanning wrote:

> retsil@iinet.net.au writes:
>
>> I think that Scope_VarFetch certainly opens up a lot of possibilities.
>> At first I thought that it was just a hack tool for IDL, but it actually
>> has some legitamite uses.
>
> I can't really believe (or discuss) what I am doing with it. But, yes, I
> am very, very impressed. :-)

IDLWAVE uses the precursor to SCOPE_VARFETCH, the undocumented
ROUTINE_NAMES, to allow pulling and examining variables from other
calling stack levels while stopped deep in a calling sequence. I like
to think that that pushed RSI over the edge to bundle it into a
supported set of routines (that and I bugged one of the developers
about it mercilessly). It's very useful. In object/widget code, I
often have a "Send project to command line" choice, which lets people
play with the object directly.

JD