Subject: Re: ROUTINE_NAMES and other magic
Posted by Craig Markwardt on Mon, 17 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

"J.D. Smith" <jdsmith@astro.cornell.edu> writes:

> Craig Markwardt wrote:
>>
>> One thing that ROUTINE_NAMES() (**note) cannot do is *add* variables
>> to another level. If the variable exists, then you can muck as much
>> as you want with it, but if it doesn't exist, sorry.
>
> I have been able to create variables on the $MAIN$ level with no problem.
> Perhaps you mean adding to non-$MAIN$ levels?

No, I meant any level. Using the example from before:

```
IDL> deepstop, 1
% Stop encountered:  DEEPSTOP           2 /dev/tty
IDL> print, routine_names('deus_ex_machina', 1, store=1)
% ROUTINE_NAMES: Variable is undefined: deus_ex_machina.
% Execution halted at:  DEEPSTOP           2 /dev/tty
%                       DEEPSTOP           3 /dev/tty
%                       $MAIN$
```

Am I doing something wrong here?

Craig

--
-- --------------
Craig B. Markwardt, Ph.D.   EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
-- --------------

Subject: Re: ROUTINE_NAMES and other magic
Posted by John-David T. Smith on Mon, 17 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:
>
> One thing that ROUTINE_NAMES() (**note) cannot do is *add* variables
> to another level. If the variable exists, then you can muck as much
> as you want with it, but if it doesn't exist, sorry.

I have been able to create variables on the $MAIN$ level with no problem.

Perhaps you mean adding to non-$MAIN$ levels?

JD

```
--
 J.D. Smith                             |*|      WORK: (607) 255-5842
 Cornell University Dept. of Astronomy  |*|            (607) 255-6263
 304 Space Sciences Bldg.               |*|       FAX: (607) 255-5875
 Ithaca, NY 14853                       |*|
```

Subject: Re: ROUTINE_NAMES and other magic
Posted by Craig Markwardt on Tue, 18 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

```
"R.Bauer" <R.Bauer@fz-juelich.de> writes:
> Craig Markwardt wrote:
>
>> One thing that ROUTINE_NAMES() (**note) cannot do is *add* variables
>> to another level.  If the variable exists, then you can muck as much
>> as you want with it, but if it doesn't exist, sorry.
>
> You can add variables to another level.
> Try this!
>
>
>
> PRO DEEPSTOP, level
>    IF level EQ 1 THEN BEGIN
>       level = ROUTINE_NAMES(/LEVEL)
>       varName = 'A'
>       void = ROUTINE_NAMES(varName, STORE=(level+1), 8)
>    ENDIF
>    IF level EQ 2 THEN begin
>    print,a
>    STOP
>    end
>    deepstop, level + 1
> END
```

Ahhh, but I argue that your procedure works for the reasons I said
before. The variable A already existed in the procedure because you
used it in a statement ("print, a"). So you really were not *adding*
the variable to the procedure.

Try this one:

```
PRO DEEPSTOP2, level
```

```
  IF level EQ 1 THEN BEGIN
;     a = 0
      level = ROUTINE_NAMES(/LEVEL)
      varName = 'A'
      void = ROUTINE_NAMES(varName, STORE=(level+1), 8)
      help
  ENDIF
  IF level EQ 2 THEN begin
    help
    STOP
  end
  deepstop, level + 1
END
```

This procedure does not mention "A" explicitly anywhere, and I can't
get beyond the first IF clause.

```
IDL> deepstop2, 1
% ROUTINE_NAMES: Variable is undefined: A.
% Execution halted at:  DEEPSTOP2           5 /dev/tty
%                        $MAIN$
```

But, if you uncomment the "a=0" line above, then you can get further.
What I find is that the value of A is set at *both* levels!

I am using an older version of IDL, 5.2. This tells me that the
functionality of ROUTINE_NAMES continued to evolve between versions,
and that you can't be guaranteed to be able to add new variables in
older versions.

```
IDL> help, !version, /str
** Structure !VERSION, 5 tags, length=80:
   ARCH            STRING    'alpha'
   OS              STRING    'OSF'
   OS_FAMILY       STRING    'unix'
   RELEASE         STRING    '5.2'
   BUILD_DATE      STRING    'Oct 30 1998'
```

Craig

--
 -- --------------
Craig B. Markwardt, Ph.D.   EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 -- --------------

Subject: Re: ROUTINE_NAMES and other magic
Posted by R.Bauer on Tue, 18 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:

> One thing that ROUTINE_NAMES() (**note) cannot do is *add* variables
> to another level. If the variable exists, then you can muck as much
> as you want with it, but if it doesn't exist, sorry.

You can add variables to another level.
Try this!

```
PRO DEEPSTOP, level
  IF level EQ 1 THEN BEGIN
    level = ROUTINE_NAMES(/LEVEL)
    varName = 'A'
    void = ROUTINE_NAMES(varName, STORE=(level+1), 8)
  ENDIF
  IF level EQ 2 THEN begin
  print,a
  STOP
  end
  deepstop, level + 1
END
```

```
IDL> deepstop,1
IDL>       8
IDL> % Stop encountered:  DEEPSTOP          9
```

Subject: Re: ROUTINE_NAMES and other magic
Posted by R.Bauer on Tue, 18 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:

> One thing that ROUTINE_NAMES() (**note) cannot do is *add* variables
> to another level. If the variable exists, then you can muck as much
> as you want with it, but if it doesn't exist, sorry.
>
> Oh, another funny thing. Try this recursive procedure:
>
>   PRO DEEPSTOP, level

```
>    if level EQ 2 then stop
>    deepstop, level + 1
>    END
>
>  and then run it with
>
>    deepstop, 1
>
>  % Stop encountered:  DEEPSTOP            2 /dev/tty
>  IDL> help
>  % At  DEEPSTOP            2 /dev/tty
>  %     DEEPSTOP            3 /dev/tty
>  %     $MAIN$
>  LEVEL           INT       =        2
>  Compiled Procedures:
>     $MAIN$  DEEPSTOP
>  Compiled Functions:
>
>  Okay, this is fine.  We've stopped two recursive levels down.  But
>  then if we try to set a variable like this:
>
>  IDL> a = 1
>
>  a = 1
>      ^
>  % Unable to add local variable to recursively active program unit: DEEPSTOP
>
>
```

Dear Craig,

I have no problems.

```
help,!version,/str
** Structure !VERSION, 5 tags, length=40:
   ARCH            STRING    'x86'
   OS              STRING    'Win32'
   OS_FAMILY       STRING    'Windows'
   RELEASE         STRING    '5.3.1'
   BUILD_DATE      STRING    'Feb 23 2000'


deepstop,1

IDL> help
% At  DEEPSTOP            2 C:\home\icg105\idl\20000418\deepstop.pro
%     DEEPSTOP            3 C:\home\icg105\idl\20000418\deepstop.pro
%     $MAIN$
```

```
A               INT       =        1
LEVEL           INT       =        2
Compiled Procedures:
    $MAIN$  DEEPSTOP

Compiled Functions:

IDL> help,a
A               INT       =        1
```

R.Bauer

Subject: Re: ROUTINE_NAMES and other magic
Posted by R.Bauer on Wed, 19 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:

```
>  "R.Bauer" <R.Bauer@fz-juelich.de> writes:
>>  Craig Markwardt wrote:
>>
>>>  One thing that ROUTINE_NAMES() (**note) cannot do is *add* variables
>>>  to another level.  If the variable exists, then you can muck as much
>>>  as you want with it, but if it doesn't exist, sorry.
>>
>>  You can add variables to another level.
>>  Try this!
>>
>>
>>
>>  PRO DEEPSTOP, level
>>     IF level EQ 1 THEN BEGIN
>>        level = ROUTINE_NAMES(/LEVEL)
>>        varName = 'A'
>>        void = ROUTINE_NAMES(varName, STORE=(level+1), 8)
>>     ENDIF
>>     IF level EQ 2 THEN begin
>>     print,a
>>     STOP
>>     end
>>     deepstop, level + 1
>>  END
>
>  Ahhh, but I argue that your procedure works for the reasons I said
>  before.  The variable A already existed in the procedure because you
>  used it in a statement ("print, a").  So you really were not *adding*
> the variable to the procedure.
>
> Try this one:
>
> PRO DEEPSTOP2, level
>    IF level EQ 1 THEN BEGIN
> ;     a = 0
>       level = ROUTINE_NAMES(/LEVEL)
>       varName = 'A'
>       void = ROUTINE_NAMES(varName, STORE=(level+1), 8)
>       help
>    ENDIF
>    IF level EQ 2 THEN begin
>      help
>      STOP
>    end
>    deepstop, level + 1
> END
>
> This procedure does not mention "A" explicitly anywhere, and I can't
> get beyond the first IF clause.
>
> IDL> deepstop2, 1
> % ROUTINE_NAMES: Variable is undefined: A.
> % Execution halted at:  DEEPSTOP2           5 /dev/tty
> %                       $MAIN$
>
> But, if you uncomment the "a=0" line above, then you can get further.
> What I find is that the value of A is set at *both* levels!
>
> I am using an older version of IDL, 5.2.  This tells me that the
> functionality of ROUTINE_NAMES continued to evolve between versions,
> and that you can't be guaranteed to be able to add new variables in
> older versions.
>
```

Dear Craig,

I did yesterday a bad mistake in my script deepstop.
Today early in the morning I recognized what's my script really does.

I have overwritten the variable level, which is the counter too.

Sorry.

Reimar