Subject: Re: get variables name
Posted by davidf on Wed, 05 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

J.D. Smith (jdsmith@astro.cornell.edu) writes:

> By the way, I once (two years ago!) made the comment in this context:
>
> "This presumably is the very reason Insight chooses to do real
> importing/exporting from the $MAIN$ level: it is attempting to be a general
> purpose analysis tool that is not tied to any one specific format of the data it
> deals with. As David, I also don't know the details of how Insight was written,
> but it is apparently written in IDL (and is restored from a save file). I am
> uncertain how they could achieve this flexibility without special built-in
> functions which they're not telling us about."
>
> I took a look inside that .sav file, and what do I find??? "ROUTINE_NAMES", of
> course. Sometimes things just come together.

Oh, my Goodness! Was that Mark Goosman, the IDL Product
Manager, screaming!?

Can you imagine what this means? It might now be possible
to write an Insight-like tool that actually worked the way
people expect it to. *And* it might even look good. :-)

Cheers,

David

P.S. Let's just say I'm in the market for knowledgeable
IDL programmers who enjoy a challenge (as opposed to,
say, money) and are anxious to live in God's Country.
Object graphics experience is NOT required. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: get variables name
Posted by John-David T. Smith on Wed, 05 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

"J.D. Smith" wrote:

>
> David Fanning wrote:
>>
>> R.Bauer (R.Bauer@fz-juelich.de) writes:
>>
>>> I like to know the name of a variable which is submitted as input to a
>>> procedure
>>>
>>> e.g
>>>
>>> pro test,var
>>> help,var
>>> end
>>>
>>> a=1
>>> test,a
>>>
>>> Is it possible to get inside test the information that test was called
>>> with a.
>>
>> My life is totally out of control at the moment, so I haven't
>> had the opportunity to write this up for my web page, but
>> here is part of a correspondence I recently received from
>> Paul Woodford, an IDL user, concerning the undocumented
>> ROUTINE_NAMES function:
>>
>> routine_names(variables=0) - variable names at current level
>> routine_names(variables=1) - variable names at main level
>> routine_names(variables=2) - variable names at one level down from main
>> routine_names(variables=-1) - variable names at one level up from current
>> routine_names(variables=-2) - variable names at two levels up from current
>>
>> But wait, that's not all... Take a look at how routine_names is used in
>> gethelp - you can use routine_names to actually fetch a variable from a
>> different level. For instance, if you have a variable named 'blah' one
>> level up from your current level, you can get it using
>>
>> blah_here = routine_names('blah', fetch=-1)
>>
> < wise cautionary matter clipped>
>
> But wait... that'still not all!
>
> By poking around in the library, I found:
>
> pro foo,a
> print,routine_names(a,/ARG_NAME)
> end

>
> IDL> foo,x
> X
>
> pretty cool. And this is the coolest... I've been asking for this for years...
> easy to abuse, but oh so useful for sending variables to the command-line and
> widget program...
>
> Try this:
>
> pro foo,a
> g=12
> print,routine_names("gvar",g,STORE=1)
> end
>

By the way, I once (two years ago!) made the comment in this context:

"This presumably is the very reason Insight chooses to do real importing/exporting from the $MAIN$ level: it is attempting to be a general purpose analysis tool that is not tied to any one specific format of the data it deals with. As David, I also don't know the details of how Insight was written, but it is apparently written in IDL (and is restored from a save file). I am uncertain how they could achieve this flexibility without special built-in functions which they're not telling us about."

I took a look inside that .sav file, and what do I find??? "ROUTINE_NAMES", of course. Sometimes things just come together.

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: get variables name
Posted by davidf on Wed, 05 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

J.D. Smith (jdsmith@astro.cornell.edu) writes:

> Finally I can create variables for later storage on the command line
> within widget apps! Or what I mean to say... hmmph... I mean... don't play with
> matches... bad dog...

I hear that in IDL 5.4 RSI is changing this routine so that
only people who can give the secret handshake of an IDL
Expert Programmers Association member will have access. :-(

Cheers,

David

P.S. I've been thinking of nominating J.D. for President Elect.
Any objections?

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: get variables name
Posted by John-David T. Smith on Wed, 05 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
>
> R.Bauer (R.Bauer@fz-juelich.de) writes:
>
>> I like to know the name of a variable which is submitted as input to a
>> procedure
>>
>> e.g
>>
>> pro test,var
>> help,var
>> end
>>
>> a=1
>> test,a
>>
>> Is it possible to get inside test the information that test was called
>> with a.
>
> My life is totally out of control at the moment, so I haven't
> had the opportunity to write this up for my web page, but
> here is part of a correspondence I recently received from
> Paul Woodford, an IDL user, concerning the undocumented
> ROUTINE_NAMES function:
>

>   routine_names(variables=0) - variable names at current level
>   routine_names(variables=1) - variable names at main level
>   routine_names(variables=2) - variable names at one level down from main
>   routine_names(variables=-1) - variable names at one level up from current
>   routine_names(variables=-2) - variable names at two levels up from current
>
>   But wait, that's not all... Take a look at how routine_names is used in
>   gethelp - you can use routine_names to actually fetch a variable from a
>   different level. For instance, if you have a variable named 'blah' one
>   level up from your current level, you can get it using
>
>   blah_here = routine_names('blah', fetch=-1)
>
< wise cautionary matter clipped>

But wait... that'still not all!

By poking around in the library, I found:

```
pro foo,a
  print,routine_names(a,/ARG_NAME)
end

IDL> foo,x
X
```

pretty cool. And this is the coolest... I've been asking for this for years...
easy to abuse, but oh so useful for sending variables to the command-line and
widget program...

Try this:

```
pro foo,a
  g=12
  print,routine_names("gvar",g,STORE=1)
end

IDL> foo
      1
IDL> help
% At  $MAIN$
GV              INT       =       12
```

Cool! This is the symmetric form of fetch, and I guess just returns 1 on
success. Finally I can create variables for later storage on the command line
within widget apps! Or what I mean to say... hmmph... I mean... don't play with
matches... bad dog...

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: get variables name
Posted by davidf on Wed, 05 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

R.Bauer (R.Bauer@fz-juelich.de) writes:

> I like to know the name of a variable which is submitted as input to a
> procedure
>
> e.g
>
> pro test,var
> help,var
> end
>
> a=1
> test,a
>
> Is it possible to get inside test the information that test was called
> with a.

My life is totally out of control at the moment, so I haven't
had the opportunity to write this up for my web page, but
here is part of a correspondence I recently received from
Paul Woodford, an IDL user, concerning the undocumented
ROUTINE_NAMES function:

routine_names(variables=0) - variable names at current level
routine_names(variables=1) - variable names at main level
routine_names(variables=2) - variable names at one level down from main
routine_names(variables=-1) - variable names at one level up from current
routine_names(variables=-2) - variable names at two levels up from current

But wait, that's not all... Take a look at how routine_names is used in
gethelp - you can use routine_names to actually fetch a variable from a
different level. For instance, if you have a variable named 'blah' one
level up from your current level, you can get it using

```
blah_here = routine_names('blah', fetch=-1)
```

Now, I would add a STRONGLY worded caution here. This routine is COMPLETELY undocumented. I suspect there is a reason for that. When you write your program relying on this behavior, and RSI changes it in the next release (which is their prerogative with a COMPLETELY undocumented function), then I don't want to hear any whining! :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155