Subject: Re: ROUTINE_INFO problems
Posted by Ted Cary on Thu, 11 Apr 2002 00:47:29 GMT
View Forum Message <> Reply to Message

Mark Hadfield wrote:

> I think the answer is "Just because".

Man, that's always the answer. But thanks again for the help. At least I won't spend more time barking up the wrong tree. I had a feeling it was a longshot.

Ted Cary

Subject: Re: ROUTINE_INFO problems
Posted by Mark Hadfield on Thu, 11 Apr 2002 01:19:14 GMT
View Forum Message <> Reply to Message

"Ted Cary" <tedcary@yahoo.com> wrote in message
news:3CB4C6E7.657CD4A5@yahoo.com...

```
> Mark Hadfield wrote:
>
>> Try the ROUTINE_INFO function with keyword PARAMETERS set and
>> examine the KW_ARGS tag in the structure it returns. (I found this
>> in the docs & I haven't tried it myself.) Just out of curiosity,
>> why do you want to do this?
>
> Thanks for the tip. ROUTINE_INFO sounded like exactly what I want,
> but I played with it for a bit and it doesn't work as expected, at
> least not for system routines like PLOT. In fact, I can't imagine
> it does what anyone wants with system routines, since the PARAMETERS
> structure it returns is just wrong.
```

Well I *did* say I hadn't tried it myself.

```
> ...Here is the output for the 'PLOT' procedure.
>
> IDL> plotParams = ROUTINE_INFO('PLOT', /PARAMETERS)
> IDL> PRINT, plotParams.num_args
>           0
> IDL> PRINT, plotParams.num_kw_args
>           0
>
> That's not right. I tried ROUTINE_INFO on PTR_FREE and even on
> itself, but the results were the same. Probably this is all
> documented or I'm just messing up, but why does ROUTINE_INFO return
> the PARAMETERS structure for system routines if it's going to lie
```

> about it?

You don't really want an answer to that question, do you? Not from
anyone other than the author of the routine, anyway. Though David
could probably think up something pithy & relevant. To do with tennis,
probably...

I think the answer is "Just because".

> The parameter information it returns for non-system routines also
> could be more complete. If a routine uses keyword inheritance and
> passes along an _EXTRA structure to a subroutine, then the KW_ARGS
> field of the PARAMETERS structure returned by ROUTINE_INFO only
> contains the word '_EXTRA.' It would be more useful to know *all*
> the keywords that could be passed to the routine, including keywords
> of any subroutines called with _EXTRA .

I rather expected that one.

> I want this information because I'm toying around with an idea that
> will probably go nowhere. I'm trying to the use the keywords as
> Get/Set-able properties of an object class, if that makes any sense.

Not entirely, but it *would* be kind of cool to be able to query an
object to see what keywords its GetProperty and SeProperty methods
support.

> So is there any way to ascertain all the keywords accepted by any
> IDL routine, including keywords of system routines and including
> keywords inherited from subroutines?

Don't know, sorry (though I suspect not). JD is probably the expert on
this (as on many other things) because there is a routine-info
facility built into the IDLWAVE Emacs mode, which he currently
maintains. But it can't recurse into inheritance chains either.

--
Mark Hadfield
m.hadfield@niwa.co.nz   Ka puwaha et tai nei
http://katipo.niwa.co.nz/~hadfield   Hoea tatou
National Institute for Water and Atmospheric Research (NIWA)

My news host refused to accept this post the first time because of "more
included text than new". Aren't computers stupid. So here's some more new
text. So here's some more new text. So here's some more new text. So here's
some more new text. So here's some more new text. So here's some more new
text. So here's some more new text. So here's some more new text. So here's
some more new text. So here's some more new text. So here's some more new

text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text. So here's some more new text.

Subject: Re: ROUTINE_INFO problems
Posted by David Fanning on Thu, 11 Apr 2002 02:16:32 GMT

View Forum Message <> Reply to Message

Mark Hadfield (m.hadfield@niwa.co.nz) writes:

> You don't really want an answer to that question, do you? Not from
> anyone other than the author of the routine, anyway. Though David
> could probably think up something pithy & relevant. To do with tennis,
> probably...

Uh, I can't think of anything pithy, but I did want you
to know that I hit two big-time, Pete-Sampras forehands
in a row today, and my girl's team lost only two games
in 7 sets in the match today. Pretty good day, tennis-wise. :-)

Cheers,

David

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

Subject: Re: ROUTINE_INFO problems
Posted by David Burridge on Thu, 11 Apr 2002 15:52:17 GMT

View Forum Message <> Reply to Message

Hi Guys,

Having *just* finished a routine to do exactly this I stumbled on your
question. Glad I didn't see it before .... I think!-)

You need to use the /SYSTEM keyword to ROUTINE_INFO and, assuming you want

to resolve it first, the /IS_FUNCTION keyword to RESOLVE_ROUTINE ...... both
of which you need to know in advance. I managed to sidestep that by putting
a couple of catches in the code so that if it failed to find a procedure
routine, it searched the functions next.

Anyhow, the net result is a routine that'll tell you the positionals and
keywords for any named routine (I have a similar thing for objects). If
you're still interested I'd be happy to send the code off list. And the
keyword inheritance thingy .... not a clue:-(Maybe an answer lies around
in the _STRICT_EXTRA stuff, but solving this bit is enough for me!

Cheers,

Dave

"Ted Cary" <tedcary@yahoo.com> wrote in message
news:3CB4EB28.B1B5854E@yahoo.com...
>
>
> Mark Hadfield wrote:
>
>> I think the answer is "Just because".
>
> Man, that's always the answer. But thanks again for the help. At least I
> won't spend more time barking up the wrong tree. I had a feeling it was a
> longshot.
>
> Ted Cary
>
>

Subject: Re: ROUTINE_INFO problems
Posted by David Burridge on Thu, 11 Apr 2002 16:08:00 GMT
View Forum Message <> Reply to Message

Oops - sorry to reply to my own post, but I made an error:-(

It *still* doesn't work for system stuff (which was the point, I guess),
even though I'm checking for it. My test case, DIST, is of course source
code so it worked nicely. PLOT etc look pretty impossible. I'll let you know
if I find anything more.

Cheers,

Dave

"David Burridge" <davidb@clogic.f9.co.uk> wrote in message

news:Rmit8.13177$51.441837@wards...
> Hi Guys,
>
> Having *just* finished a routine to do exactly this I stumbled on your
> question. Glad I didn't see it before .... I think!-)
>
> You need to use the /SYSTEM keyword to ROUTINE_INFO and, assuming you want
> to resolve it first, the /IS_FUNCTION keyword to RESOLVE_ROUTINE ......
both
> of which you need to know in advance. I managed to sidestep that by
putting
> a couple of catches in the code so that if it failed to find a procedure
> routine, it searched the functions next.
>
> Anyhow, the net result is a routine that'll tell you the positionals and
> keywords for any named routine (I have a similar thing for objects). If
> you're still interested I'd be happy to send the code off list. And the
> keyword inheritance thingy .... not a clue:-(Maybe an answer lies
around
> in the _STRICT_EXTRA stuff, but solving this bit is enough for me!
>
> Cheers,
>
> Dave
>
> "Ted Cary" <tedcary@yahoo.com> wrote in message
> news:3CB4EB28.B1B5854E@yahoo.com...
>>
>>
>> Mark Hadfield wrote:
>>
>>> I think the answer is "Just because".
>>
>> Man, that's always the answer. But thanks again for the help. At least
I
>> won't spend more time barking up the wrong tree. I had a feeling it was
a
>> longshot.
>>
>> Ted Cary
>>
>>
>
>

Subject: Re: ROUTINE_INFO problems

Posted by JD Smith on Fri, 12 Apr 2002 17:44:33 GMT
View Forum Message <> Reply to Message

Mark Hadfield wrote:

```
>
> "Ted Cary" <tedcary@yahoo.com> wrote in message
> news:3CB4C6E7.657CD4A5@yahoo.com...
>> Mark Hadfield wrote:
>>
>>> Try the ROUTINE_INFO function with keyword PARAMETERS set and
>>> examine the KW_ARGS tag in the structure it returns. (I found this
>>> in the docs & I haven't tried it myself.) Just out of curiosity,
>>> why do you want to do this?
>>
>> Thanks for the tip.  ROUTINE_INFO sounded like exactly what I want,
>> but I played with it for a bit and it doesn't work as expected, at
>> least not for system routines like PLOT.  In fact, I can't imagine
>> it does what anyone wants with system routines, since the PARAMETERS
>> structure it returns is just wrong.
>
> Well I *did* say I hadn't tried it myself.
>
>> ...Here is the output for the 'PLOT' procedure.
>>
>> IDL> plotParams = ROUTINE_INFO('PLOT', /PARAMETERS)
>> IDL> PRINT, plotParams.num_args
>>           0
>> IDL> PRINT, plotParams.num_kw_args
>>           0
>>
>> That's not right.  I tried ROUTINE_INFO on PTR_FREE and even on
>> itself, but the results were the same.  Probably this is all
>> documented or I'm just messing up, but why does ROUTINE_INFO return
>> the PARAMETERS structure for system routines if it's going to lie
>> about it?
>
> You don't really want an answer to that question, do you? Not from
> anyone other than the author of the routine, anyway. Though David
> could probably think up something pithy & relevant. To do with tennis,
> probably...
>
> I think the answer is "Just because".
>
>> The parameter information it returns for non-system routines also
>> could be more complete.  If a routine uses keyword inheritance and
>> passes along an _EXTRA structure to a subroutine, then the KW_ARGS
>> field of the PARAMETERS structure returned by ROUTINE_INFO only
>> contains the word '_EXTRA.'  It would be more useful to know *all*
>> the keywords that could be passed to the routine, including keywords
```

>> of any subroutines called with _EXTRA .
>
> I rather expected that one.
>
>> I want this information because I'm toying around with an idea that
>> will probably go nowhere. I'm trying to the use the keywords as
>> Get/Set-able properties of an object class, if that makes any sense.
>
> Not entirely, but it *would* be kind of cool to be able to query an
> object to see what keywords its GetProperty and SeProperty methods
> support.
>
>> So is there any way to ascertain all the keywords accepted by any
>> IDL routine, including keywords of system routines and including
>> keywords inherited from subroutines?
>
> Don't know, sorry (though I suspect not). JD is probably the expert on
> this (as on many other things) because there is a routine-info
> facility built into the IDLWAVE Emacs mode, which he currently
> maintains. But it can't recurse into inheritance chains either.

Not entirely true... newer versions of IDLWAVE do indeed follow
inheritance chains for keyword information in association with object
inheritance, but they don't do this by default for all routines with
*_EXTRA. Why? Looking through the body of the code for calls with
_EXTRA is error-prone and time consuming, and often returns something
other than what you want.

That said, you should know that IDLWAVE is very devious in the way it
gets information, relying on text versions of the PDF manuals for the
bulk of the system info. That's why it knows all the keywords to PLOT.

JD