Subject: Re: translating an array name to a string
Posted by John-David T. Smith on Thu, 19 Oct 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Alex Schuster wrote:
>
> David Williams wrote:
>
>> Hi. This is probably a very basic IDL question, so apologies if that's
>> the case. I'm looking for a way to translate the name of an array (e.g.
>> "DATACUBE1") into a string that I can use in titles and/or in feedback
>> at the prompt. I want to make my routines more user-friendly, and I hate
>> forcing a title
>
> Have a look at the OUTPUT keyword to HELP:
>
> IDL> help, movie
> MOVIE BYTE = Array[256, 256, 64]
> IDL> help, movie, output=output
> IDL> print, output
> MOVIE BYTE = Array[256, 256, 64]
> CLU> print, (str_sep(output[0], " "))[0]
> MOVIE

Also try:

print,routine_names(movie,/ARG_NAME)

Don't try looking this one up in the manual. Neither of these operations are supported. RSI reserves the right to remove or retool routine_names() AND/OR the output format of help. That's the price you pay.

By the way, for those of you using routine_names for heavy magic... you might consider examining the following extra-cautions snippet to export a variable to the $MAIN$ level:

```
  var_free=0
  catch, err
  if err ne 0 then begin
    ;; An undefvar indicates routine_info ran and
    ;; the variable is free
    if !ERROR_STATE.NAME ne 'IDL_M_UNDEFVAR' then begin
      catch,/cancel
message,"Can't complete operation... Try obj=sp_sel()"
    endif
    var_free=1
```

```
  endif

  ;; If we need to check if the variable name is available, do so.
  if var_free eq 0 then $
    rn=call_function('routine_names',var_name,FETCH=1)

  if n_elements(rn) ne 0 then begin
 catch,/cancel
 message,'A variable named '+var_name+' already exists.'
  endif

  ;; Still here... we need to export ourself to the main level
  rn=call_function('routine_names',var_name,myvar,store=1)
```

basically the idea is to wrap routine_names as a string in
call_function, to allow your routine to compile even if RSI yanks or
renames it (it wouldn't compile if you tried to call it directly).
You'll get an error, of course, which will be caught. You have to
discriminate between errors caused by the successful operation of
routine_names(), and those caused by incorrect arguments, changed
keywords (IDL_M_KEYWORD_BAD), or other mutations routine_names() has
undergone (such as vanishing altogether -- IDL_M_UPRO_UNDEF). You
obviously have to have a backup plan too, to tell your users what to do
in case routine_names() has broken. But it's better than your program
not running at all though.

JD

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