Subject: Re: Keyword checking
Posted by R.Bauer on Wed, 29 May 2002 16:02:58 GMT
View Forum Message <> Reply to Message

Randall Skelton wrote:

Dear Randall,

keyword_set
 should only used to test variables to be true or false

arg_present
 is used to find out if a parameter is called by reference
   this gives true and by value gives false

n_elements
 returns the number of elements of a parameter

We have set up a test case in our lessons
I have posted some time ago.

I believe all is clear if you once filled out these forms

regards

Reimar

1)
There are three functions given

```
   FUNCTION test1,minimum=min_val
      IF KEYWORD_SET(min_val) THEN RETURN,1 ELSE RETURN,0
   END


   FUNCTION test2,minimum=min_val
      IF ARG_PRESENT(min_val) THEN RETURN,1 ELSE RETURN,0
   END
   FUNCTION test3,minimum=min_val
      IF N_ELEMENTS(min_val) GT 0 THEN RETURN,1 ELSE RETURN,0
   END
```

Fill out the form:

CALL	test1	test2	test3
PRINT, testX( )			
PRINT, testX(minimum=0)			
PRINT, testX(minimum=10)			
PRINT, testX(minimum=-10)			
mv=0 & PRINT, testX(minimum=mv)			
mv=10 & PRINT, testX(minimum=mv)			
PRINT, testX(minimum=mv2)			

2)
There are three functions given

```
   FUNCTION test1,min_val
      IF KEYWORD_SET(min_val) THEN RETURN,1 ELSE RETURN,0
   END


   FUNCTION test2,min_val
      IF ARG_PRESENT(min_val) THEN RETURN,1 ELSE RETURN,0
   END


   FUNCTION test3,min_val
```

```
    IF N_ELEMENTS(min_val) GT 0 THEN RETURN,1 ELSE RETURN,0
  END
```

Fill out the form:

CALL	test1	test2	test3
PRINT, testX( )			
PRINT, testX(0)			
PRINT, testX(10)			
PRINT, testX(-10)			
mv=0 & PRINT, testX(mv)			
mv=10 & PRINT, testX(mv)			
PRINT, testX(mv2)			

```
>
>  I have run into a great deal of trouble checking keywords in IDL and I
>  thought I would relay my thoughts and frustrations.  I am trying to
>  prevent a user from passing an undefined variable in a keyword...
>
>  Imagine,
>
>  pro test, A=a, B=b
>    print, n_params()
>    print, arg_present(a), keyword_set(a), n_elements(a), size(a,/type)
>    print, arg_present(b), keyword_set(b), n_elements(b), size(b,/type)
>  end
>
>  If I call this routine as:
>
```

```
> IDL> .reset
> IDL> test, a=1, b=undefvar
> % Compiled module: TEST.
>       0
>     0   1     1     2
>     1   0     0     0
>
> So, in order to prevent a user from passing junk in a keyword, I have:
>
> pro test, B=b
>  if n_elements(b) eq 0 and arg_present(b) eq 1 then $
>   message, 'You passed an undefined variable as a keyword'
>  if n_elements(b) gt 0 then b = 'passed'
> end
>
> Now,
>
> IDL> .reset
> IDL> test, b=junk
> % Compiled module: TEST.
> % TEST: You passed an undefined variable as a keyword
>
> Surely there must be something simpiler than my 2-step approach ;)
>
> Cheers,
> Randall
```

--
Reimar Bauer

Institut fuer Stratosphaerische Chemie (ICG-I)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de

```
 ------------------------------------------------------------ -------
       a IDL library at ForschungsZentrum Juelich
  http://www.fz-juelich.de/icg/icg1/idl_icglib/idl_lib_intro.h tml
 ============================================================ =======
```