Subject: Re: keyword_set bug or feature
Posted by zawodny on Fri, 12 May 1995 07:00:00 GMT
View Forum Message <> Reply to Message

In article <3p0030$d0s@post.gsfc.nasa.gov> thompson@orpheus.nascom.nasa.gov (William Thompson) writes:
>
> This is a very common error that people make. They want to use KEYWORD_SET()
> to test for the *existence* of keyword variables. KEYWORD_SET() is *only* for
> keywords which can take True or False values, and has the (quite correct in my
> opinion) property that not passing something is the same as passing it as
> False.
>
> To test for the *existence* of keywords that can take arbitrary values, use the
> function N_ELEMENTS() instead.

While Bill Thompson is certainly knowledgeble and we are for the most part better off because he takes the time to answer peoples questions, his answers sometimes have a tone of arrogance (or at least a bit of "If you do not do it my way, then you are wrong").

In this instance he is again basically correct. Keywords come in two flavors. Those that take the logical values True and False and those that do not. The former are used primarily for program control (do something special when a particular keyword is set). The latter is used to pass optional "valued" parameters in a function or proceedure.

In an environment where code is shared or codeveloped, it is important to define and obey certain rules or standard practices. When using keywords in such an environment it is best (not the *only* proper way) to set logical keywords using /. For example

```
 mypro,v1,v2,/logical_key
```

and to test for them being set by using the KEYWORD_SET() function. Non-logical keywords are best given a value (notice I did not say "best set") using the = operator and tested for by using the N_ELEMENTS() function.

It is not, however, an *error* to do otherwise since

```
 mypro,v1,v2,logical_key=1
```

does indeed produce the same behavior as above. Errors produce erroneous results or cause improper functionality. Similarly,

```
 mypro,v1,v2,/non-logical_key
```

is A way to set the value of non-logical_key equal to 1.

These are not wrong ways or *errors*, but merely bad habits. If you code by habit or your software users have bad habits then you must be careful. The only instance where a problem appears is when you use the logical test KEYWORD_SET() on a non-logical keyword which might have a valid value of zero. Otherwise, try to get into a better habit, but in a pinch use whatever works.

It is interesting to note that
KEYWORD_SET(0) = 0 ,
and KEYWORD_SET(2) = 1 as they obviously should be,
but KEYWORD_SET([0]) = 1 !

I'm sure this tells many of us just how KEYWORD_SET() is coded.

I think that what a number of folks here are asking for is a new function, call it KEYWORD_EXISTS(), which returns true if the keyword was used in the call to a proceedure or function. Such a function would be trivial to write,

```
function KEYWORD_EXISTS(key)
return,(n_elements(key) ne 0)
end
```

and might actually get some use if provided within IDL as a complement to the KEYWORD_SET() function. Seeing

```
if KEYWORD_EXISTS(key) then ...
```

in some foreign code is a bit more readable than

```
if(N_ELEMENTS(key) ne 0) then ...
```

,but not by much.

I hope that this explains the KEYWORD issues a bit for the newcommers.

--
Joseph M. Zawodny (KO4LW) NASA Langley Research Center
Internet: j.m.zawodny@larc.nasa.gov MS-475, Hampton VA, 23681-0001
TCP/IP: ko4lw@ko4lw.ampr.org Packet: ko4lw@n4hog.va.usa.na