Subject: Re: Probs w/ keyword_set
Posted by sjt on Wed, 03 Aug 1994 09:03:32 GMT
View Forum Message <> Reply to Message

Alan Tai (tai@rockola.larc.nasa.gov) wrote:
: This is probably old news, but in IDL 3.6 (and I think in 3.51), I
: realized early on that you run into problems by setting a keyword
: parameter to 0 and running it through keyword_set. I.e. keyword_set
: believes the parameter is unset:

```
: pro test, Key = Key
:    if (keyword_set(Key)) then print, 'Set' $
:    else print, 'Unset'
: end
: test, Key = 0
```

: Solution: use n_elements instead of keyword_set. I found this out
: from reading through some procedures from the JHUAPL library.

This has always been the case. The main prupose of keyword_set as I've
always seen it is coupled with /<key> constructs.

: Alan

--
James Tappin, School of Physics & Space Research
   University of Birmingham
sjt@xun8.sr.bham.ac.uk
   "If all else fails--read the instructions!"

P.S. Did you know that keyword_set's argument needn't be a keyword! it
actually just returns 0 if arg is 0, '' or undefined and 1 otherwise.

Subject: Re: Probs w/ keyword_set
Posted by thompson on Wed, 03 Aug 1994 14:09:52 GMT
View Forum Message <> Reply to Message

tai@rockola.larc.nasa.gov (Alan Tai) writes:

> This is probably old news, but in IDL 3.6 (and I think in 3.51), I
> realized early on that you run into problems by setting a keyword
> parameter to 0 and running it through keyword_set. I.e. keyword_set
> believes the parameter is unset:

```
> pro test, Key = Key
>    if (keyword_set(Key)) then print, 'Set' $
>    else print, 'Unset'
> end
> test, Key = 0
```

> Solution: use n_elements instead of keyword_set. I found this out
> from reading through some procedures from the JHUAPL library.

This is the way it's supposed to work. Why do you think it's a problem? It's standard for 0 to represent false and 1 to represent true. The beauty of the KEYWORD_SET function is that if the variable is undefined, then it assumes false by default.

It's possible you're trying to use KEYWORD_SET for keywords that aren't true/false switches. If a keyword can take other values than true/false, and you're trying to decide whether or not that keyword has been passed, then N_ELEMENTS is the correct way to test for this, as you point out.

For example, in the AXIS routine, the keyword XAXIS can take the values 1 or 0. However, in this case the values do not represent true or false. Instead, setting XAXIS=0 means draw an axis under the plot window, and XAXIS=1 means draw it over the plot window. Leaving it undefined means to not draw either. In a situation like this, using KEYWORD_SET to decide whether or not XAXIS was passed would be a mistake.

However, in the same routine, it would be correct to use KEYWORD_SET to determine the state of the /SAVE keyword.

I've noticed that sometimes people will use the /KEYWORD form to set a keyword that can take a variety of values to 1. This is bad programming practice. It confuses the distinction between keywords that make true/false distinctions, and keywords that take more traditional values.

Hope this clears up the confusion,

Bill Thompson