Subject: Re: passing multiple keywords to subroutines
Posted by Liam Gumley on Mon, 25 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

David Ritscher wrote:
> It's often desirable to redefine a function, and then have this new
> function behave almost like the original function. To cook up an
> example, let's say I need a new plot function, that plots the x-axis
> as .001 units:
>
> pro myplot, x
> plot, findgen(n_elements(x) * 0.001), x
> return
> end
>
> It would now be great to be able to pass any and all possible keywords
> into this new plotting function, so I might call it as:
> myplot, x, title='test', linestyle=3
>
> In IDL, this can be done by including all possible keywords (rather
> tedious!)

It can be done this way, and it is tedious indeed.

Alternatively, you can exploit the keyword inheritance features offered
by the _EXTRA keyword, e.g.

```
pro myplot, x, _extra = extra_keywords
plot, findgen(n_elements(x) * 0.001), x, _extra = extra_keywords
end
```

Now all keywords which go unrecognized by your routine MYPLOT are passed
into (but not out of) the PLOT procedure, so that

```
myplot, x, title='test', linestyle=3
```

will work the way you intended. In the online help, click the Contents
button, select 'Building IDL Applications', 'Defining Procedures and
Programs', 'Keyword Inheritance'.

Cheers,
Liam.

Liam E. Gumley
Space Science and Engineering Center, UW-Madison
1225 W. Dayton St., Madison WI 53706, USA
Phone (608) 265-5358, Fax (608) 262-5974

Subject: Re: passing multiple keywords to subroutines
Posted by J.D. Smith on Mon, 25 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

David Ritscher wrote:
>
> It's often desirable to redefine a function, and then have this new
> function behave almost like the original function. To cook up an
> example, let's say I need a new plot function, that plots the x-axis
> as .001 units:
>
> pro myplot, x
> plot, findgen(n_elements(x) * 0.001), x
> return
> end
>
> It would now be great to be able to pass any and all possible keywords
> into this new plotting function, so I might call it as:
> myplot, x, title='test', linestyle=3
>
> In IDL, this can be done by including all possible keywords (rather
> tedious!) In PV-Wave, it's almost impossible, since it is not
> possible to pass in an undefined keyword:
>
> (I have not defined the variable 'color' here:)
> WAVE> plot, [1,5], color=color
> % PLOT: Variable is undefined: COLOR.
> % Execution halted at $MAIN$ (PLOT).
>
> This same command works fine in IDL, and has the appropriate behavior
> (Here, with 'color' undefined, it behaves exactly the same as if the
> keyword was not specified).
>
> Thus in IDL it is possible to define my new function along these lines:
> pro myplot, x, color=color, linestyle=linestyle, ynozero=ynozero, $
> noclip=noclip, yrange=yrange,
> ...
> As I say, this becomes rather tedious.
>
> Does anyone see a simpler way to handle this sort of thing? It seems
> to me that that two things are needed:
> 1. both companies need to add a mechanism to handle keywords that will
> be passed on to subroutines, which might take on a form like the
> following:
> pro pass_keywords=pass_keywords myplot, x

> plot, x, pass_keywords=pass_keywords
>
> where I can now call 'myplot' with extra keywords:
> myplot, x, linestyle=3
>
> 2. The bugs in PV-Wave have to be fixed, such that undefined keywords
> can be handled by every function.
>
> Am I missing a better way to handle these issues?
>
> David Ritscher
>

uhhhhh....

Take a look at IDL's _EXTRA and _REF_EXTRA keyword inheritance mechanisms...

or am I missing something?

JD

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