
Subject: Re: `_extra` and `call_method`

Posted by [Benjamin Hornberger](#) on Fri, 28 Jan 2005 20:38:11 GMT

[View Forum Message](#) <> [Reply to Message](#)

David Fanning wrote:

> Benjamin Hornberger writes:

>

>

>> can't keyword passing by `_extra` be used with `call_method`? I am getting

>> an error message "Keyword parameters not allowed in call" if I try to do

>> so.

>>

>> I want to write a wrapper routine which first checks if an object is

>> valid, and if yes, calls methods on it. For positional parameters, I

>> have to use "case `n_params()` of", which is not very nice, but for

>> keywords at least I would like to use inheritance...

>

>

> Are you **sure** you are doing this correctly? I've never

> had the least bit of trouble with it. It sounds to me

> like the method you are calling does not have the particular

> keyword you are using defined for it. Is this a spelling

> problem, perhaps?

>

> Cheers,

>

> David

>

Well, I figured out what the problem is. It's not just about `call_method`, but keyword inheritance in general. If you call a procedure and pass `_extra` keywords to it even though it does not accept any keywords at all, the error occurs. Simple example:

PRO taco

 print, 'Hello, world!'

END

PRO call_taco, _extra=extra

 taco, _extra=extra

END

In this case, when running "call_taco", IDL issues an error because I call "taco" with `_extra` keywords even though it doesn't accept any.

In my case, I want to write a general wrapper routine for several methods of an object. For simplicity, let's say all object methods accept one positional parameter, but zero or more keywords. What I would like to do is:

```
PRO call_object, object, method, par, _extra=extra
```

```
;; method is passed as a string
```

```
IF ~obj_valid(object) THEN BEGIN
```

```
;; issue an error and more
```

```
ENDIF
```

```
call_method, method, object, par, _extra=extra
```

```
END
```

Now if there are any object methods which don't have any keywords defined, the error occurs as described.

Any more hints, except defining a dummy keyword for each method?

Thanks,
Benjamin
