Subject: Re: Generating keyword parameters from strings
Posted by Paul Van Delst[1] on Thu, 30 Jul 2009 15:55:14 GMT
View Forum Message <> Reply to Message

JDS wrote:
> On Jul 23, 3:48 pm, Paul van Delst <paul.vande...@noaa.gov> wrote:
>> David Fanning wrote:
>>> Paul van Delst writes:
>>>> I really don't understand why, in the SUBCLASS::Get_Property method, I can get away with
>>>> using the _EXTRA keyword to the SUPERCLASS::Get_Property method, but it works. Go figure.
>>> The rule is you use _REF_EXTRA (or _REF_STRICT_EXTRA, etc) on
>>> the procedure or function *definition* line, but *all* extra
>>> parameters are to be *passed* with _EXTRA.
>> Ah, o.k. A nice, simple rule to remember.
>>
>> Of course it would be nicer if I didn't have to remember it at all. Passing
>> arguments/keywords by reference or value is an implementation detail the user shouldn't be
>> concerned with. Having to consider it in IDL OO programming is mildly ironic.
>
> Nicety is in the eye of the beholder. At the time I proposed
> _REF_EXTRA and it was added to IDL 5.1, the IDL programmer who
> implemented it was very proud that it required only a single change to
> the routine definition line, not all the _EXTRA calls themselves. At
> the time, I objected to the duplicate interface. But I came to
> understand that it was an absolute requirement due to a simple but
> frustrating issue: various people (including those posting to this
> thread!) routinely build or augment their own _EXTRA structures.
>
> If you think about it, there is no way (without inventing a new type
> of IDL variable) to simultaneously change the semantics of _EXTRA to
> allow passing by reference, and *not* break all of those sneaky codes
> which are poking around in _EXTRA structure themselves. So we have
> only ourselves to blame.

Two things:

1) When has that ever stopped RSI/IITVIS in the past from changing things? (o.k., o.k. just kidding! :o)
2) Why is building and/or augmenting your own _EXTRA structure a bad thing? I'm lazy so I don't do it myself, but the IDL documentation provides information on how to do it and use the results. If the vendor itself exposes the flawed implementation interface and thus tacitly encourages its use, you can't blame the programmer for getting creative.

Given the following IDL code,

```
pro assign, x
  x = !PI
```

```
end

pro test_assign
  mystruct = {x:0.0, y:0.0}
  myarray = fltarr(4)
  assign, mystruct.x
  assign, myarray[3]
  help, mystruct, /struct
  print, myarray
end
```

until the output looks like

```
IDL> test_assign
** Structure <95c36f4>, 2 tags, length=8, data length=8, refs=1:
   X               FLOAT           3.14159
   Y               FLOAT           0.00000
     0.00000     0.00000     0.00000     3.14159
```

IDL will be a bit of an oddity (IMO :o).

cheers,

paulv
