Subject: Generating keyword parameters from strings
Posted by Eric Hudson on Wed, 22 Jul 2009 15:56:06 GMT
View Forum Message <> Reply to Message

Hi,

Does anyone know of a method of taking a string and turning it into a keyword parameter? As an example, say I want to write a function:

```
function GetProperty, object, propertyName
  object->GetProperty, 'propertyName'=value ; This is pseudocode --
this line is what I need
  return, value
end
```

Obviously I can write this using execute, but I'd prefer to avoid that if at all possible.
Also, although I use the GetProperty method as an example, this is a more generic question about generating keyword parameter calls (so, for example, David's nice discussion of a general GetProperty method http://www.dfanning.com/tips/getproperty.html doesn't help here).

If I were just setting a value this would be easy to do with _EXTRA by making my own structure. But for getting values it doesn't seem like there is an easy equivalent (it would be nice to just make a _REF_EXTRA structure if it behaved equivalent to _EXTRA but it seems it doesn't).

Thanks,
Eric

Subject: Re: Generating keyword parameters from strings
Posted by David Fanning on Thu, 30 Jul 2009 15:44:53 GMT
View Forum Message <> Reply to Message

JD writes:

> 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.

Instead of assigning blame, I prefer to think of it as a perfect example of the evolutionary principle in action. What can happen, will happen. :-)

Cheers,

David
--
David Fanning, Ph.D.
Coyote's Guide to IDL Programming (www.dfanning.com)
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

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

Subject: Re: Generating keyword parameters from strings
Posted by Kenneth P. Bowman on Thu, 30 Jul 2009 17:48:42 GMT
View Forum Message <> Reply to Message

In article <MPG.24db6dddeec3306f98983f@news.giganews.com>,
David Fanning <news@dfanning.com> wrote:

> Instead of assigning blame, I prefer to think of it
> as a perfect example of the evolutionary principle
> in action. What can happen, will happen. :-)
>
> Cheers,
>

> David

Like some genes with both beneficial and harmful effects (e.g., the gene that causes sickle cell anemia), _EXTRA must have some survival value for the IDL organism.

Ken

Subject: Re: Generating keyword parameters from strings
Posted by David Fanning on Thu, 30 Jul 2009 18:08:55 GMT
View Forum Message <> Reply to Message

Kenneth P. Bowman writes:

> Like some genes with both beneficial and harmful effects (e.g., the gene that
> causes sickle cell anemia), _EXTRA must have some survival value for the
> IDL organism.

Let's just say without it, writing object programs
would be almost exactly like writing iTools programs. :-(

Cheers,

David

--
David Fanning, Ph.D.
Coyote's Guide to IDL Programming (www.dfanning.com)
Sepore ma de ni thui. ("Perhaps thou speakest truth.")
