Subject: Re: Still missing features in IDL 8
Posted by Bob[4] on Thu, 04 Nov 2010 18:29:37 GMT
View Forum Message <> Reply to Message

On Nov 1, 9:30 am, Chris Torrence <gorth...@gmail.com> wrote:
> On Oct 31, 6:00 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
>
>
>
>> On Oct 13, 2:38 pm, Chris Torrence <gorth...@gmail.com> wrote:
>
>>> Regarding #2, what if you could use additional indices to access array
>>> elements within lists?
>
>>> For example:
>
>>> IDL> a = LIST(FINDGEN(10), BYTARR(5,3))
>>> IDL> help, a[0]
>>> <Expression> FLOAT = Array[10]
>>> IDL> help, a[0,3] ; currently throws an error in IDL8.0
>>> <Expression> FLOAT = 3.00000
>>> IDL> a[0,3] = !pi ; currently throws an error in IDL8.0
>
>>> IDL> help, a[1]
>>> <Expression> BYTE = Array[5, 3]
>>> IDL> help, a[1,4,2] ; currently throws an error in IDL8.0
>>> <Expression> BYTE = 0
>>> IDL> a[1,4,2] = 255 ; currently throws an error in IDL8.0
>
>>> So the first index would give the list element, and the remaining
>>> indices would index into the array itself. Obviously you could only
>>> have up to 7 dimensions in your contained array, but that probably
>>> isn't a huge limitation.
>
>> I was writing a class like that, inheriting from list, and that
>> brought me a question: Should the extra dimension (of the list index)
>> be on the left, as above, or on the right?
>
>> The notation (already valid for retrieving values) (a[1])[0] suggests
>> that the array index should come on the left. However, writing a[1,0]
>> suggests array dimensions, in which case the list index would make
>> more sense on the right, as the list dimension is the slowest-varying
>> one.
>
>> Tough it would be a bit incoherent with the array dimension order, it
>> seems to me that it is better to have the list index on the left. That
>> way,
>

>> print,(a[1])[0] ;already valid
>
>> would be the same as
>
>> print,a[1,0]
>
>> instead of the more confusing
>
>> print,a[0,1]
>
>> Any thoughts on that?
>
> Yes, that is exactly what I was thinking.
>
> Back to your original thread - if we added this way of subscripting,
> does that eliminate the need to convert a list to/from a pointer
> array? I'd rather not add more functionality if we don't have to.
>
> -Chris
> ITTVIS

It seems to me that adding array subscripting to the list object is
wrong since lists can take many more types of objects than arrays.
This seems to be a product of IDL think that "everything is an array",
which does not make sense anymore in 8.0. I think the proper way to
implement the feature Paulo is asking for is to have an array_list
object, which would only take arrays and could have overloaded
brackets to get the extra subscripting. I think this is what Paulo is
implementing already and it would seem it could be completely
implemented with no changes to the language. Perhaps if you wanted to
implement it in the language, an ARRAY keyword could be added to the
list::init to force the list to only accept arrays and only if that is
set then the bracket overloading could be added. It still seems
cleaner to have a subclass object, however.

Bob
