Subject: Re: objarr
Posted by J.D. Smith on Tue, 07 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

bjackel@phys.ucalgary.ca wrote:
>
> Say I have a single object called Vector3, that allows nice things like
>
> mag= Vector3->magnitude()
> dot= Vector3->dotproduct(secondvector)
>
> and so on. Then assume I've got a whole pile of vectors
>
> pile= OBJARR(100)
> FOR indx=0,99 DO pile[i]= Vector3
>
> and want to manipulate them all at once
>
> magpile= pile->magnitude()
> % Object reference must be scalar in this context: PILE
> % Execution halted at: $MAIN$
>
> One foremost advantages of IDL is that logical groups of
> vector and matrix operations can be carried out with a
> single command. Am I correct in fearing that the only
> way to get what I want is with something like this?
>
> magpile= DBLARR(N_ELEMENTS(pile))
> FOR indx=0,N_ELEMENTS(pile)-1 DO magpile[indx]=
> pile[indx]->magnitude()
>
> Any comments or suggestions would be most welcome.

Why not make your own array object as a container class for vector objects? You could hide all of this code behind an object interface, and also allow things like:

vec_pile->Rotate,!PI/2.

etc. And you could maintain the object list more consistently, ensuring it's integrity as Vector objects only.

While it might often be convenient for us to treat object arrays as traditional arrays, it's no more valid than expecting something like

```
a=[ptr_new(5),ptr_new('Foo')]
print,*a
```

to work. Objects have no single type or operational (binary,
arithmetic, etc.) defaults, just as pointers have none. It's up to you
to define these through the object oriented framework, which can be used
to create much more elaborate and useful operant constructs (as in your
example of dot products) than the traditional, static ones.

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 |*|