Subject: Re: Object Data and pointer assignments
Posted by davidf on Thu, 09 Mar 2000 08:00:00 GMT
View Forum Message <> Reply to Message

J.D. Smith (jdsmith@astro.cornell.edu) writes:

> Just to be clear... you are free to free self.inarray, and point it somewhere
> else, at any time. This can be useful if you have a list which is either empty
> (NULL pointer a.k.a. a dangling reference), or not (pointer to a list of finite
> size). If the list changes size, and becomes empty again, you can simply free
> it, which indicates its emptiness. If it then grows again, simply use ptr_new()
> to get another heap variable for it. So, while it might be easiest in some
> cases only to call ptr_new() once, in other cases it is useful to let a single
> member variable like self.inarray point to different heap variables over its
> life.

Lord knows I need more excitement in my life if I'm quibbling with
quibbles, but let me make one suggestion:

If I want to point to an "empty" variable, I prefer to
use a pointer to an undefined variable. The advantage
to me is that this is a VALID pointer, in contrast
to the NULL pointer, which is an invalid pointer.

Note:

```
IDL> a = Ptr_New()
IDL> Print, Ptr_Valid(a)
      0
IDL> *a = 5
   % Unable to dereference NULL pointer: A.

IDL> b = Ptr_New(/Allocate_Heap)
IDL> Print, Ptr_Valid(b)
      1
IDL> *b = 5
```

I like this because it fits into the programming style
I've developed. For example:

```
IF N_Elements(color) EQ 0 THEN color = 5
IF N_Elements(*b) EQ 0 THEN *b = 5
```

But again, you must *initialize* this pointer to an
undefined variable in the INIT method, NOT in the __DEFINE
module.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155