Subject: Using NO_COPY with pointers
Posted by David Fanning on Mon, 14 Apr 2003 15:22:40 GMT
View Forum Message <> Reply to Message

Folks,

This may be common knowledge, but I wasn't aware of it, and
it is one of those things that makes you feel all warm and
goose-pimply about IDL.

I was adding a "user value" to all of my objects
today, via a UVALUE field in the "atom" object that
is inherited by all objects in my library. This field
is, of course, a pointer.

Naturally enough, I want to be able to get and set
the "value" of this field sometimes without making
a copy of the data. This is no problem when I am
adding the information to the pointer, I simply use
the NO_COPY keyword on PTR_NEW:

```
  self.uvalue = Ptr_New(uvalue, /No_Copy)
```

But it is a bit of a problem when I want to "get"
the value back:

```
  uvalue = *self.uvalue
```

I was of the impression that pointer de-referencing
always made a copy of the data. But on a whim, I
tried this:

```
  IF Keyword_Set(no_copy) THEN uvalue = Temporary(*self.uvalue)
```

Lo and behold, this did *exactly* what I wanted it to do!!

```
  Help, *self.uvalue
    <PtrHeapVar2093>
    UNDEFINED = <Undefined>
```

Moreover,

```
   Print, Ptr_Valid(self.uvalue)
     1
```

Perfect! A valid pointer that points to an undefined variable.
Hooray! You gotta love IDL! :-)

Cheers,

David

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