Subject: Re: When Ptr_New doesn't work
Posted by John-David T. Smith on Tue, 22 Jan 2002 20:02:18 GMT
View Forum Message <> Reply to Message

Richard Younger wrote:
>
> Hi, Carles.
>
> There's something very fishy here. Your code seems perfectly correct to
> me. I found that when the outer set of parenthesis on the left hand
> side of the offending line of code is omitted, your program snippet
> seems to work fine. When I include the outer set of parenthesis, it
> gives me an error on versions Win 5.3-5.5, and possibly earlier. My
> test program is below. Either this is a bug or a very strange quirk of
> the language with obscure origins. Perhaps it should be reported?
>
> For anyone else out there running on different systems, do other
> versions have this problem?

>  ;Works:
>  (*rCoefficients[j])[k] = Ptr_New(FltArr(2), /NO_COPY)
>
>  ;Causes Error:
>  ;((*rCoefficients[j])[k]) = Ptr_New(FltArr(2), /NO_COPY)

There's nothing mysterious about this error:

```
IDL> a=fltarr(4)
IDL> a[1]=1
IDL> (a[1])=1
% Expression must be named variable in this context: <FLOAT    (
  1.00000)>.
% Execution halted at: $MAIN$
```

The problem is with IDL's somewhat strict definition of what can consititute a "left-hand", or assignable, value. It handles the simple case of:

```
IDL> ((((a))))=1
```

without choking, but throw anything more complicated at the LHS, and it often falls down. It's a known weakness of IDL, revealed in a complicated way. ;) One good rule of thumb is that, if the LHS contains subscripts, the last character before the "=" (modulo whitespace), should be "]" (ok Craig, or ")").

Here's another example:

```
IDL> a={a:1}
IDL> a.a=1
IDL> (a.a)=1
% Expression must be named variable in this context: <INT      (
1)>.
% Execution halted at: $MAIN$
```

The same rule applies.  If structure dereference occurs, the sequence prior to "=" must be an unadulterated tag (unless you're combining structure dereference and subscripting... it can get ugly).

I like to work from the inside out with long nested *[]() sequences.  A really nice feature to be released in an upcoming version of IDLWAVE lets you do in-place inspection of parts of the command line (or prior commands visible in the buffer) while your composing it, similar to the Shift-click and Shift-Drag printing available now in the buffer.  For instance, while composing:

```
(*rCoefficients[j])[k]=
```

you could key-drag various regions (or trust IDLWAVE's ability to pick the expression of the "right" length nearby), and inspect them with customized commands ("print,size(___,/DIMENSIONS)" being one of my current favorites).

This feature helps a lot with these kinds of deeply nested compositions (even in the buffer! -- just set a breakpoint and inspect away).  It would also help if RSI would generate a true operator precedence table including "[]" (array subscript) and "." (structure dereference).

Good luck,

JD

---