
Subject: Re: dereferencing pointer arrays.

Posted by [David Fanning](#) on Thu, 06 Feb 2003 03:51:01 GMT

[View Forum Message](#) <> [Reply to Message](#)

Rochelle Hatton (nrh@nospam.imag.wsahs.nsw.gov.au) writes:

> On following some useful tips from the newsgroup, I have
> referenced my assoc variables (multiple files) with pointers. I have then
> placed these pointers into an array. Not being used to using pointers,
> I checked the help on dereferencing my array (BTW I really dislike the pdf
> online help in 5.6!).
> So the help says, if you have a pointer array mm, you can dereference it using
> the index e.g. if I want the first pointer, I can type
> IDL> print, *mm[0]
> however I get this error-
> % PRINT: File expression not allowed in this context: <PtrHeapVar9>.
>
> Also, if I assign the index to a variable, it works e.g. xx=*mm[0]
> I guess this is a problem because of what I am storing in the pointer,
> namely a non-variable reference thingy which is
> like a pointer (the assoc variable).
> Is there any way I can dereference the assoc
> variable without having to assign it-the whole point of using the pointer??

Well, you can dereference the pointer if you are using
it in an appropriate associated variable context. For example:

```
IDL> filename = filepath(subdir=['examples','data'], 'abnorm.dat')
IDL> openr, lun, filename, /Get_Lun
IDL> b=assoc(lun, bytarr(64, 64))
IDL> ptr= ptr_New(b)
IDL> print, *ptr ; Inappropriate context.
% PRINT: File expression not allowed in this context:
<PtrHeapVar131736>.
% Execution halted at: $MAIN$
IDL> print, b ; Also inappropriate context.
% PRINT: File expression not allowed in this context: B.
% Execution halted at: $MAIN$
IDL> e = (*ptr)[4] ; Appropriate context
IDL> tv, e
```

Associated variables are **always** assigned to something, even if
indirectly:

```
IDL> tv, (*ptr)[3]
```

It doesn't even matter that you have pointer arrays:

```
IDL> ptrarray = ptrArr(2)
IDL> ptrarray[0] = ptr
IDL> bb = Assoc(lun, bytarr(64, 64, 15))
IDL> ptrarray[1] = ptr_new(bb)
IDL> vol = (*ptrarray[1])[0]
```

I think the whole point of using the pointer in the previous context is that it was the only way to pass an associated variable in an info structure between widget program modules. There was no way to store the associate variable in the structure otherwise. It was the sort of thing that was likely to impress the IDL EPA committee, but pretty much useless otherwise. :-)

Cheers,

David

P.S. The alternative is to store the lun in the structure, and make the associated variable (using the lun) whenever you need it.

--

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

Subject: Re: dereferencing pointer arrays.
Posted by [Rochelle Hatton](#) on Thu, 06 Feb 2003 04:33:52 GMT
[View Forum Message](#) <> [Reply to Message](#)

It's always an obvious thing isn't it.
That's exactly what I am missing- a few extra brackets!
Thanks.

David Fanning wrote:

```
>
> It doesn't even matter that you have pointer arrays:
>
> IDL> ptrarray = ptrArr(2)
> IDL> ptrarray[0] = ptr
> IDL> bb = Assoc(lun, bytarr(64, 64, 15))
> IDL> ptrarray[1] = ptr_new(bb)
> IDL> vol = (*ptrarray[1])[0]
>
```

> I think the whole point of using the pointer in the previous
> context is that it was the only way to pass an associated variable
> in an info structure between widget program modules. There
> was no way to store the associate variable in the structure
> otherwise. It was the sort of thing that was likely to impress
> the IDL EPA committee, but pretty much useless otherwise. :-)
>
> Cheers,
>
> David
>
> P.S. The alternative is to store the lun in the structure,
> and make the associated variable (using the lun) whenever you need it.
>
>

--

Rochelle Hatton
Medical Physicist
Department of Nuclear Medicine and Ultrasound
Westmead Hospital ph(02) 9845 7223
