Subject: Re: Everything I always wanted to know about IDL * but were afraid to ask. Chapter1
Posted by David Fanning on Fri, 14 Nov 2003 12:44:54 GMT
View Forum Message <> Reply to Message

df23775@hotmail.com writes:

> *A=[*A, next_value]
> As many times as the values I want to add to the pointer.
>
> I can print the whole pointer with PRINT, *A. But how can I get intermediate
> positions??.

You subscript like any normal IDL variable. The only
trick is knowing that pointer dereferencing has (generally)
lower precedence than almost everything else. So we throw
a lot of parentheses around when we are working with pointers.

```
values = (*A)[4:6]
*A = [(*A)[0:6], new_Value, (*A)[7:*]]
```

> As far as I can recall from my C classes I do believe that in C, A means the
> memory position pointed by A, in what we could call the first pointer
> positions and you could use a counter to get further position with a syntax
> like *(A+counter) In IDL *A means not only the memory position pointed by A
> but also the full contents of the pointer?

Forget what you learned about C pointers. IDL pointers
are NOT C pointers. Think of them as regular IDL variables
that live in global memory space and you will be MUCH
further ahead. :-)

Cheers,

David

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

Subject: Re: Everything I always wanted to know about IDL * but were afraid to ask. Chapter1
Posted by Nuno Oliveira on Fri, 14 Nov 2003 13:38:36 GMT

Hi, David, thanks again for your tip. It is clear to me that IDL is NOT C language. But also it is clear to me that I'm trying to be a good boy searching the tutorials (still the tutorials!). And again the information was not enough. Because I noticed some similar syntax between IDL and C language, I confess, I tried a C-like solution before I come to this newsgroup. :)

Cheers,

Nuno.

"David Fanning" <davidf@dfanning.com> wrote in message news:MPG.1a1e7c32da522d9a989684@news.frii.com...
> df23775@hotmail.com writes:
>
>
> Forget what you learned about C pointers. IDL pointers
> are NOT C pointers. Think of them as regular IDL variables
> that live in global memory space and you will be MUCH
> further ahead. :-)
>
> Cheers,
>
> David
>
> --
> David W. Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming: http:/www.dfanning.com/
> Phone: 970-221-0438, IDL Book Orders: 1-888-461-0155