Subject: Re: Arrays of Structures
Posted by Paul Van Delst[1] on Thu, 08 Feb 2007 19:02:08 GMT
View Forum Message <> Reply to Message

Mick Brooks wrote:
> On Feb 8, 5:41 pm, "Brian Larsen" <balar...@gmail.com> wrote:
>
>> This is what you did and it is true this is an array with 50 elements
>> IDL> HELP, structs.a
>> <Expression> INT = Array[50]
>>
>> but if you look at structs by itself not structs.a, it is the array,
>> NOT structs.a
>>
>> IDL> HELP, structs
>> STRUCTS STRUCT = -> <Anonymous> Array[50]
>>
>> So what is inside the structure?
>>
>> Here a is inside the struct and it is an int
>>
>> IDL> HELP, structs, /str
>> ** Structure <84bd40c>, 1 tags, length=2, data length=2, refs=1:
>> A INT 1
>
> (I think) I'm following you to here...
>
>> Meaning that structs.a[23] doesn't make sense because structs.a is an
>> int.
>
> ... but this is where you lose me. Doesn't the the first line that you
> showed say that structs.a is an array of 50 ints?
> Maybe I just can't read the output of HELP properly, but struts.a
> certainly sometimes behaves as an array. Try PRINTing it, for
> instance.
> Or let's compare the output of HELP on a real array of 50 ints:
>
> IDL> HELP, structs.a
> <Expression> INT = Array[50]
> IDL> HELP, indgen(50)
> <Expression> INT = Array[50]
>
>> While structs[23].a does make sense because structs is an array.
>
> Yes, I'm happy with why this is right, but still not clear why
> structs.a[23] is wrong.

You know, I'm confused now too. Check out the precedence in the IDL help:

Table 12-9: Operator Precedence

Priority	Operator
First (highest)	( ) (parentheses, to group expressions)
	[ ] (brackets, to concatenate arrays)
Second	. (structure field dereference)
	[ ] (brackets, to subscript an array)
	( ) (parentheses, used in a function call)
etc...	

So, you see that the the structure field dereference operator, ".", and the array subscript operator, "[]" , have the same precedence. Operators with equal precedence are evaluated from left to right.

So,

```
  structs.a[23]
```

means FIRST dereference the structure field (structs.a), THEN index the array ([23]). The way I see it,

```
  structs.a[23]
and
  (structs.a)[23]
```

should be equivalent.

Maybe the weirdness has something to do with where the result of the dereference "goes" ? Hence the "<No name>" in the error message?

Hmm.

paulv

--
Paul van Delst Ride lots.
CIMSS @ NOAA/NCEP/EMC Eddy Merckx