Subject: Re: update variable in structure
Posted by David Fanning on Tue, 16 Jun 2009 16:14:53 GMT
View Forum Message <> Reply to Message

M. Suklitsch writes:

> So far, so good. Now we do exactly the same, but this time the
> variable is embedded in a structure:
> IDL> my_struct = {my_value:8}
> IDL> update_value, my_value
> IDL> help, my_struct, /STRUC
> ** Structure <8220044>, 1 tags, length=2, data length=2, refs=1:
> MY_VALUE INT 8
>
> And now the rather simple question: how come this doesn't work?

I think you must have meant, why doesn't *this*
work:

IDL> udate_value, my_struck.my_value

And the reason is that structure dereferences, like
expressions, array subscripts, etc., etc. (in fact,
anything *except* a variable) are passed by *value*
and not by *reference*. Things that are passed by
value make a copy of themselves and pass that, rather
than passing the thing itself.

Cheers,

David

--
David Fanning, Ph.D.
Coyote's Guide to IDL Programming (www.dfanning.com)
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: update variable in structure
Posted by Michael Galloy on Tue, 16 Jun 2009 16:19:59 GMT
View Forum Message <> Reply to Message

M. Suklitsch wrote:
> Hi everybody!
>
>
> Today I have a question regarding the update of variables within a
> structure, which does not work as I would expect.

>
> Say we have a very simple program:
>
> ===============
> PRO update_value, input
>
> input = input MOD 5
>
> END
> ===============
>
> [In reality, this subroutine/program does some more sophisticated
> things, but this is sufficient to prove my point. ;-)]
>
> Okay, now we call this routine with a variable holding an integer
> value.
> IDL> my_value = 8
> IDL> update_value, my_value
> IDL> help, my_value
> MY_VALUE   INT   =   3
>
> So far, so good. Now we do exactly the same, but this time the
> variable is embedded in a structure:
> IDL> my_struct = {my_value:8}
> IDL> update_value, my_value
> IDL> help, my_struct, /STRUC
> ** Structure <8220044>, 1 tags, length=2, data length=2, refs=1:
>   MY_VALUE   INT   8
>
> And now the rather simple question: how come this doesn't work?
> Normally IDL is eager to overwrite variables of any kind. On some
> occasions, I've seen it overwriting the "parental" variable of a
> duplicated one. And more important: is there a way to get the above
> thing working?

Well, I assume you mean to refer to the field in the structure you just created, as in:

```
IDL> update_value, my_struct.my_value
IDL> help, my_struct.my_value
<Expression>    INT       =        8
```

The reason my_struct.my_value was not modified is that only "named variables" are passed by reference, so changes to them by the called routine will still be in effect at the caller level. The expression "my_struct.my_value" is not a named variable (named variables are just the name of a variable like "my_value" was in your previous examples), so modification to it inside update_value are only to a local variable.

> Maybe important, maybe not: I'm working with IDL 7.0 and have tried it
> on Solaris and Linux.

Should not matter for this.

Mike
--
www.michaelgalloy.com
Associate Research Scientist
Tech-X Corporation

Subject: Re: update variable in structure
Posted by M. Suklitsch on Tue, 16 Jun 2009 16:38:13 GMT
View Forum Message <> Reply to Message

On 16 Jun., 18:19, mgalloy <mgal...@gmail.com> wrote:
> M. Suklitsch wrote:
>> Hi everybody!
>
>> Today I have a question regarding the update of variables within a
>> structure, which does not work as I would expect.
>
>> Say we have a very simple program:
>
>> ===============
>> PRO update_value, input
>
>> input = input MOD 5
>
>> END
>> ===============
>
>> [In reality, this subroutine/program does some more sophisticated
>> things, but this is sufficient to prove my point. ;-)]
>
>> Okay, now we call this routine with a variable holding an integer
>> value.
>> IDL> my_value = 8
>> IDL> update_value, my_value
>> IDL> help, my_value
>> MY_VALUE INT = 3
>
>> So far, so good. Now we do exactly the same, but this time the
>> variable is embedded in a structure:
>> IDL> my_struct = {my_value:8}
>> IDL> update_value, my_value

>> IDL> help, my_struct, /STRUC
>> ** Structure <8220044>, 1 tags, length=2, data length=2, refs=1:
>> MY_VALUE INT 8
>
>> And now the rather simple question: how come this doesn't work?
>> Normally IDL is eager to overwrite variables of any kind. On some
>> occasions, I've seen it overwriting the "parental" variable of a
>> duplicated one. And more important: is there a way to get the above
>> thing working?
>
> Well, I assume you mean to refer to the field in the structure you just
> created, as in:
>
> IDL> update_value, my_struct.my_value
> IDL> help, my_struct.my_value
> <Expression> INT = 8
>
> The reason my_struct.my_value was not modified is that only "named
> variables" are passed by reference, so changes to them by the called
> routine will still be in effect at the caller level. The expression
> "my_struct.my_value" is not a named variable (named variables are just
> the name of a variable like "my_value" was in your previous examples),
> so modification to it inside update_value are only to a local variable.
>
>> Maybe important, maybe not: I'm working with IDL 7.0 and have tried it
>> on Solaris and Linux.
>
> Should not matter for this.
>
> Mike
> --www.michaelgalloy.com
> Associate Research Scientist
> Tech-X Corporation

Thanks for your quick replys!
In that case I've got a problem... or rather I've to find a neat
workaround for my own work. :)

Bye,
Martin

Subject: Re: update variable in structure
Posted by Jean H. on Tue, 16 Jun 2009 17:13:10 GMT
View Forum Message <> Reply to Message

> Thanks for your quick replys!

> In that case I've got a problem... or rather I've to find a neat
> workaround for my own work. :)
>
> Bye,
> Martin

make is a function

function xyz, arg

return, arg mod 5
end

myStruct.field = xyz(mystruct.field)

or copy the argument

tmp = mystruct.field
xyz,tmp
mystruct.field = tmp

Jean

Subject: Re: update variable in structure
Posted by R.G. Stockwell on Tue, 16 Jun 2009 23:55:06 GMT
View Forum Message <> Reply to Message

"M. Suklitsch" <martin@suklitsch.at> wrote in message
news:308ff1f9-67da-493e-bde1-46d29e3f63cf@a7g2000yqk.googleg roups.com...
> Hi everybody!
>
>
> Today I have a question regarding the update of variables within a
> structure, which does not work as I would expect.
>
> Say we have a very simple program:
>
> ===============
> PRO update_value, input
>
> input = input MOD 5
>
> END

> ==========
...
> And now the rather simple question: how come this doesn't work?

Passs the structure, then inside modify the field you need modified:

PRO update_value, input

input.test = 13

END

data = {test:2, str:'hello'}

update_value, data

print,data

end

Note: you can get very fancy if you want a general routine,
perhaps pass in the field name and use that string in an execute call,
or pass the field number you want to modify, and access the
structure like input.(0) = 2

cheers,
bob

Subject: Re: update variable in structure
Posted by R.Bauer on Wed, 17 Jun 2009 17:01:52 GMT
View Forum Message <> Reply to Message

R.G. Stockwell schrieb:
> "M. Suklitsch" <martin@suklitsch.at> wrote in message
> news:308ff1f9-67da-493e-bde1-46d29e3f63cf@a7g2000yqk.googleg roups.com...
>> Hi everybody!
>>
>>
>> Today I have a question regarding the update of variables within a
>> structure, which does not work as I would expect.
>>
>> Say we have a very simple program:
>>
>> ==========
>> PRO update_value, input

>>
>> input = input MOD 5
>>
>> END
>> ===============
> ...
>> And now the rather simple question: how come this doesn't work?
>
>
> Passs the structure, then inside modify the field you need modified:
>
> PRO update_value, input
>
> input.test = 13
>
> END
>
> data = {test:2, str:'hello'}
>
> update_value, data
>
> print,data
>
> end
>
>
> Note: you can get very fancy if you want a general routine,
> perhaps pass in the field name and use that string in an execute call,
> or pass the field number you want to modify, and access the
> structure like input.(0) = 2
>
> cheers,
> bob
>

Hi

the easiest thing is to convert the structure params to pointers
and afterwards back to a structure without pointers if you don't like
pointers.

 http://www.fz-juelich.de/icg/icg-1/idl_icglib/idl_source/idl
_html/dbase/struct2ptr_struct_dbase.pro.html

 http://www.fz-juelich.de/icg/icg-1/idl_icglib/idl_source/idl
_html/dbase/ptr_struct2struct_dbase.pro.html

struct={A:1,b:findgen(10)}

```
help,struct,/str
** Structure <1d5bbd8>, 2 tags, length=44, data length=42, refs=1:
   A               INT             1
   B               FLOAT     Array[10]

result=struct2ptr_struct(struct)
help,result,/str
** Structure <10551e8>, 2 tags, length=8, refs=1:
   A               POINTER   <PtrHeapVar1>
   B               POINTER   <PtrHeapVar2>

*result.b = "don't get fancy"

struct = ptr_struct2struct(result,/free)
help, struct ,/str
** Structure <1d5bd18>, 2 tags, length=24, data length=18, refs=1:
  A               INT             1
  B               STRING    'don't get fancy'


cheers
Reimar


>
>
>
>
>
```