
Subject: Static Variables in IDL

Posted by [santanu](#) on Wed, 13 Mar 1996 08:00:00 GMT

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

Hi,

Is there a way to statically allocate variables in IDL ? I would like to define a static variable in a .pro which is repeatedly called from an upper level. I am specifically looking for the IDL counterpart of

```
static int testint;
```

I would appreciate any help in this regard,

Thanks

Santanu

santanu@uiuc.edu

Subject: Re: Static Variables in IDL

Posted by [peter](#) on Thu, 14 Mar 1996 08:00:00 GMT

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

Santanu Bhattacharyya (santanu@ux7.cso.uiuc.edu) wrote:

: Hi,

: Is there a way to statically allocate variables in IDL ? I would
: like to define a static variable in a .pro which is repeatedly called
: from an upper level. I am specifically looking for the IDL counterpart
: of
: static int testint;

: I would appreciate any help in this regard,

:

: Thanks

: Santanu

: santanu@uiuc.edu

Place it in a common block. Declare the common block everywhere you want the variable to be in scope.

Peter

Peter Webb, HP Labs Medical Dept
E-Mail: peter_webb@hpl.hp.com
Phone: (415) 813-3756

Subject: Re: Static Variables in IDL

Posted by [rivers](#) on Thu, 14 Mar 1996 08:00:00 GMT

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

In article <4i8geg\$5fv@vixen.cso.uiuc.edu>, santanu@eehpx22.cen.uiuc.edu (S Bhattacharyya) writes:

> rivers@cars3.uchicago.edu (Mark Rivers) writes:

>

> Pardon me if I appear a bit obtuse, but I am still a little confused. I am

> under the impression that the common block declaration is equivalent to

> C's global declaration. What I would like to have is a bit of non re-entrant

> code in a standalone function (.pro). I want an IDL .pro that does the

> following::

>

> main()

> {

> for(;;) non_rEntrant();

> }

> non_rEntrant()

> {

> static int block=1;

>

> if (block == 1){

> puts("This is executed only once");

> block=0;

> }

> puts("And this is done over and over again");

> }

>

> The following does exactly the same thing -----

>

> pro test

> common SHARE,block

> block=1

>

> repeat begin

> call_procedure,'non_rEntrant'

> endrep until block eq 1

> end

>

> pro non_rEntrant

> common SHARE,block

> if block eq 1 then begin print,'This is executed only once' & block=0 & endif

> print,'And this is done over and over again'

> end

>

> But the two lower level function/pro's are not the same, the

> C version is standalone, how do I make the IDL pro non_rEntrant behave

> in the same way ?

Change your IDL example to the following:

```
pro test
while (1) call_procedure,'non_rEntrant'
end

pro non_rEntrant
common SHARE,block
if (n_elements(block) eq 0) then begin
    print,'This is executed only once'
    block=0
endif
print,'And this is done over and over again'
end
```

The key is `n_elements()`. If `non_rEntrant` has never been called then `block` is undefined and `n_elements(block)=0`.

Mark Rivers	(312) 702-2279 (office)
CARS	(312) 702-9951 (secretary)
Univ. of Chicago	(312) 702-5454 (FAX)
5640 S. Ellis Ave.	(708) 922-0499 (home)
Chicago, IL 60637	rivers@cars3.uchicago.edu (Internet)

Subject: Re: Static Variables in IDL
Posted by [santanu](#) on Thu, 14 Mar 1996 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

rivers@cars3.uchicago.edu (Mark Rivers) writes:

> In article <4i7ara\$606@vixen.cso.uiuc.edu>, santanu@ux7.cso.uiuc.edu (Santanu Bhattacharyya) writes:

>> Hi,

>> Is there a way to statically allocate variables in IDL ? I would

>> like to define a static variable in a .pro which is repeatedly called

>> from an upper level. I am specifically looking for the IDL counterpart

>> of

>> static int testint;

>>

> Yes, it is easy. Use a common block. Make the name of the common block unique
> enough that you can be sure it won't conflict with common blocks used in other
> procedures.

> common unique_name, testint

> testint = 14

Pardon me if I appear a bit obtuse, but I am still a little confused. I am under the impression that the common block declaration is equivalent to C's global declaration. What I would like to have is a bit of non re-entrant code in a standalone function (.pro). I want an IDL .pro that does the following::

```
main()
{
  for(;;) non_rEntrant();
}
non_rEntrant()
{
  static int block=1;

  if (block == 1){
    puts("This is executed only once");
    block=0;
  }
  puts("And this is done over and over again");
}
```

The following does exactly the same thing -----

```
pro test
common SHARE,block
block=1

repeat begin
  call_procedure,'non_rEntrant'
endrep until block eq 1
end

pro non_rEntrant
common SHARE,block
if block eq 1 then begin print,'This is executed only once' & block=0 & endif
print,'And this is done over and over again'
end
```

But the two lower level function/pro's are not the same, the C version is standalone, how do I make the IDL pro non_rEntrant behave in the same way ?

I really appreciate the help,

Thanks,

Santanu

santanu@uiuc.edu

> testint will have the value 14 the next time the .pro file is called.

>

> Mark Rivers	(312) 702-2279 (office)
> CARS	(312) 702-9951 (secretary)
> Univ. of Chicago	(312) 702-5454 (FAX)
> 5640 S. Ellis Ave.	(708) 922-0499 (home)
> Chicago, IL 60637	rivers@cars3.uchicago.edu (Internet)
