Subject: Re: Static Variables in IDL
Posted by steinhh 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:

```
|>
|>
|> 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");
|> }
```

```
pro main
  while 1 do non_rEntrant
end

pro non_rEntrant
common non_rEntrant_private_others_keep_off,block

  if N_elements(block) eq 0 then block=1   ;; Instead of "static .... =1"

  if block eq 1 then begin
     print,"This is executed only once"
     block = 0
  end
  print,"And this is done over and over again"
end
```

This will do what you want. Common blocks are not quite like
global declarations. I can have e.g., a variable called block
in the main program without having a conflict. I can even

have a variable called block at the interactive prompt level,
and still have no conflict.

|> call_procedure,'non_rEntrant'

You'll get speedier programs by using just

```
   non_rEntrant
```

as in the example above 8-)

Stein Vidar