
Subject: Best way to share 'enum' between various functions

Posted by [Robin Wilson](#) on Sat, 24 Sep 2011 08:24:41 GMT

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

Hi all,

I have a number of constant definitions that I'm using to do what an enum would do in other languages:

MARITIME = 1

CLOUD = 2

URBAN = 3

...

I want these definitions to be available to a lot of different functions and procedures (all in different files) in my code - they are used a lot in IF statements and various assignments.

Obviously I don't want to replicate the definitions in loads of different files, as if I ever change them (or add extra ones) then it will get **very** confusing.

What's the best way to do this?

Cheers,

Robin

Robin Wilson

A PhD student studying complexity in remote sensing

www.rtwilson.com/academic

Subject: Re: Best way to share 'enum' between various functions

Posted by [SonickKenking](#) on Tue, 27 Sep 2011 00:28:40 GMT

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

On Sep 24, 6:24 pm, Robin Wilson <ro...@rtwilson.com> wrote:

> Hi all,

>

> I have a number of constant definitions that I'm using to do what an

> enum would do in other languages:

>

> MARITIME = 1

> CLOUD = 2

> URBAN = 3

> ...

>
> I want these definitions to be available to a lot of different functions
> and procedures (all in different files) in my code - they are used a lot
> in IF statements and various assignments.
>
> Obviously I don't want to replicate the definitions in loads of
> different files, as if I ever change them (or add extra ones) then it
> will get *very* confusing.
>
> What's the best way to do this?
>
> Cheers,
>
> Robin
>
> -----
> Robin Wilson
> A PhD student studying complexity in remote sensing www.rtwilson.com/academic

Define the variables in it's own file, say the file is "myconst.pro".
In other routines, use @myconst to include the variables, similar to C
header files.

Subject: Re: Best way to share 'enum' between various functions
Posted by [Paul Van Delst\[1\]](#) on Tue, 27 Sep 2011 14:53:10 GMT
[View Forum Message](#) <> [Reply to Message](#)

SonicKenking wrote:

>
> Define the variables in it's own file, say the file is "myconst.pro".
> In other routines, use @myconst to include the variables, similar to C
> header files.

Seconded. This is what I do with a lot of stuff. For example, I have a single file containing all of the
fundamental
constants I use:

```
idl/Constants : more fundamental_constants.pro
; PI
PI          = 3.141592653589793238462643d0
PI_RECIPROCAL = 0.318309886183790671537767d0
PI_SQUARED   = 9.869604401089358618834491d0
PI_SQUARE_ROOT = 1.772453850905516027298167d0
PI_LN        = 1.144729885849400174143427d0
PI_LOG10     = 0.497149872694133854351268d0

; E
```

```

E          = 2.718281828459045235360287d0
E_RECIPROCAL = 0.367879441171442321595523d0
E_SQUARED    = 7.389056098930650227230427d0
E_LOG10      = 0.434294481903251827651129d0
....etc....
SPEED_OF_LIGHT = 2.99792458d+08
PERMEABILITY = PI * 4.0d-07
PERMITTIVITY = ONE / ( PERMEABILITY * SPEED_OF_LIGHT^2 )
PLANCK_CONSTANT = 6.62606876d-34
GRAVITATIONAL_CONSTANT = 6.673d-11
....etc....

```

And I use this file all over the place, e.g.

```

idl/Planck/define : dir
total 68K
-rw-r--r-- 1 wd20pd wd4 8.7K Apr 28 2009 planck_dbdt.pro
-rw-r--r-- 1 wd20pd wd4 9.5K Apr 28 2009 planck_dtdb.pro
-rw-r--r-- 1 wd20pd wd4 8.0K Apr 28 2009 planck_radiance.pro
-rw-r--r-- 1 wd20pd wd4 8.8K Nov 9 2010 planck_temperature.pro
idl/Planck/define : grep fundamental_constants *.pro
planck_dbdt.pro: @fundamental_constants
planck_dtdb.pro: @fundamental_constants
planck_radiance.pro: @fundamental_constants
planck_temperature.pro: @fundamental_constants

```

```

idl/Units_Conversion : dir
total 48K
-rw-r--r-- 1 wd20pd wd4 214 May 14 2009 db_to_response.pro
-rw-r--r-- 1 wd20pd wd4 2.6K May 14 2009 ghz_to_inverse_cm.pro
-rw-r--r-- 1 wd20pd wd4 150 May 14 2009 ghz_to_mhz.pro
-rw-r--r-- 1 wd20pd wd4 2.3K May 14 2009 inverse_cm_to_ghz.pro
-rw-r--r-- 1 wd20pd wd4 150 May 14 2009 mhz_to_ghz.pro
-rw-r--r-- 1 wd20pd wd4 178 May 14 2009 response_to_db.pro
idl/Units_Conversion : grep fundamental_constants *.pro
ghz_to_inverse_cm.pro: @fundamental_constants
inverse_cm_to_ghz.pro: @fundamental_constants

```

I also use the same method for common code "snippets", e.g. error handlers

```

idl/osRF/lib : more osrf_err_handler.pro
; OSRF function message handler
@error_codes
IF ( KEYWORD_SET(Debug) ) THEN BEGIN

```

```
MESSAGE, '--> Entered.', /INFORMATIONAL
MsgSwitch = 0
ENDIF ELSE BEGIN
  CATCH, Error_Status
  IF ( Error_Status NE 0 ) THEN BEGIN
    CATCH, /CANCEL
    MESSAGE, !ERROR_STATE.MSG
  ENDIF
  MsgSwitch = 1
ENDELSE
```

which I include in any needed routines via a

```
@osrf_err_handler
```

It all works quite well.

cheers,

paulv
