Subject: Re: Consensus on error handling with DLMs
Posted by Paul van Delst on Fri, 27 Apr 2001 13:23:12 GMT
View Forum Message <> Reply to Message

Randall Skelton wrote:
>
> Hi all,
>
> I've more or less finished writing an IDL interface to Postgres and I'm
> now in the debugging stage. I thought I'd take a poll to see what people
> think of appropriate error returns. In this library I have a variety of
> function returns... integers, floats, doubles, strings, complex structures
> and so forth. For integer returns, I usually default to giving the user a
> message with the handle IDL_MSG_INFO in IDL_Message and returning -1 on
> failure. Is there a good protocol for signifying an error in strings,
> structures and arrays? Some of my default string returns are themselves
> null strings (indicating that no data or message was found) so it wouldn't
> be wise to simply return a null string on error. I am also very reluctant
> to return a float -1.0000 as testing for this can lead to problems with
> IEEE number definitions in C. For the moment, I am using the
> IDL_MSG_LONGJMP to signal an error in all routines that don't return an
> IDL integer. It stops the interpreter immediately (which isn't
> necessarily bad) as it signifies a major fault. Comments?
>
> Thanks,
> Randall

Your situation is a lot more complicated than mine, but I return integer status variables
in functions. Any actual data is returned in the argument list., e.g.:

```
FUNCTION blah, x1, x2, x3
  @error_codes

  CATCH, error_status
  IF ( error_status NE 0 ) THEN BEGIN
    CATCH, /CANCEL
    MESSAGE, !ERROR_STATE.MSG, /CONTINUE
    RETURN, FAILURE
  ENDIF

  .... here do some interesting stuff. If an error occurs
  .... anywhere in the code I use a :

  MESSAGE, 'An error occurred counting the number of numbats', $
         /NONAME, /NOPRINT


  ... At the end of the code I have a :
```

```
  CATCH, /CANCEL
  RETURN, SUCCESS

END
```

In testing the result I sim0ply do a :

```
  @error_codes
  IF ( result NE SUCCESS ) THEN.....
```

where SUCCESS and FAILURE are defined in error_codes.pro (there are other definitions as well like WARNING and INFORMATION, UNDEFINED, etc). Like I said, the above is a very simple approach but it's worked pretty well for me - and I like the fact that there are only two exits from the code, one for success and one for a failure. I used to have returns peppered through code which sorta sucked.

paulv

p.s. I only turn this error checking on *after* debugging though - if code crashes I want to know the line number i crashed at rather than always returning with a failure.

--
Paul van Delst A little learning is a dangerous thing;
CIMSS @ NOAA/NCEP Drink deep, or taste not the Pierian spring;
Ph: (301)763-8000 x7274 There shallow draughts intoxicate the brain,
Fax:(301)763-8545 And drinking largely sobers us again.
paul.vandelst@noaa.gov Alexander Pope.