
Subject: Re: Error Handling Advice

Posted by [David Fanning](#) on Mon, 14 Jan 2013 18:44:08 GMT

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

Matthew Argall writes:

> Ok, I think I have reasoned with myself enough to know when to use ON_ERROR, 2 and when to use CATCH, but most CATCH blocks that I see just print an error message and return to the calling program, anyway, without doing anything special.

>

> For example, take the following

>

> -----%<-----

> CATCH, Error_status

>

> IF Error_status NE 0 THEN BEGIN

> CATCH, /CANCEL

> PRINT, 'Error index: ', Error_status

> PRINT, 'Error message: ', !ERROR_STATE.MSG

> RETURN

> ENDIF

> -----%<-----

>

> Is there any reason for not using ON_ERROR, 2 in this case?

In this case, probably not. That CATCH error handler sucks. :-)

What you normally want to do is print out some kind of useful information about where the error occurred in the program (what line number) and what the error was. Otherwise, you will have a devil of a time fixing it. My standard error handler looks like this:

```
Catch, theError
IF theError NE 0 THEN BEGIN
    Catch, /CANCEL
    void = Error_Message()
    RETURN
ENDIF
```

Error_Message is a Coyote Library routine that will display a pop-up dialog (if appropriate) and write the error traceback out to the command line where you can read the message and do something about the error. The Catch error handler also clears the error status, so you return to the caller with a clear slate. This is especially important in widget programs, where you often like to keep the programs running, even if errors occur.

> From what I gather, the CATCH blocks are normally used to make a program "fail gracefully", in

that it resets the device and color table, closes files, frees pointers, etc. But without anything to reset, is ON_ERROR, 2 sufficient?

I'm afraid the CATCH block doesn't do ANY of these things, unless you explicitly do it yourself in the CATCH block. Freeing up your pointers is a good idea!

> I guess I will start with some with trial and ::ehem:: error ;-)

Yep. :-)

Cheers,

David

--

David Fanning, Ph.D.

Fanning Software Consulting, Inc.

Coyote's Guide to IDL Programming: <http://www.idlcoyote.com/>

Sepore ma de ni thue. ("Perhaps thou speakest truth.")
