Subject: Re: error reading file question
Posted by David Fanning on Tue, 07 Nov 2006 14:12:54 GMT

View Forum Message <> Reply to Message

David Groff writes:

> Is there a way in IDL to check for problems when reading a data file?
> For example in Fortran you would do something like
> Error_Status = Read (filename)
> IF (Error_Status /= Success) THEN
> --------------------------(etc.)
>
> If something is wrong with the data file the problem would be discovered
> via error_status.

Typically, one uses a CATCH error handler for this,
although ON_IOERROR is still sometimes used. Here is
a fairly old article that explains some of the details:

http//www.dfanning.com/misc_tips/conversion_errors.html

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: error reading file question
Posted by Paul Van Delst[1] on Tue, 07 Nov 2006 19:08:53 GMT

View Forum Message <> Reply to Message

David Groff wrote:
> Is there a way in IDL to check for problems when reading a data file?
> For example in Fortran you would do something like
> Error_Status = Read (filename)
> IF (Error_Status /= Success) THEN
> --------------------------(etc.)
>
> If something is wrong with the data file the problem would be discovered
> via error_status.
>
> Thanks

Hello again Dave! :o)

This is how I do it (you can check the netCDF TauProfile read function for more details):

```
pro test
   ; Setup error handler
   catch, ierr
   if(ierr ne 0)then begin
     catch, /cancel
     message, !error_state.msg + '; Closing file...', /continue
     if(n_elements(fileid) ne 0) then free_lun, fileid
     return
   endif
   ; Read non-existant file
   openr, fileid, 'newfile.dat', /get_lun
   x=0.0
   readu, fileid, x
   free_lun, fileid
   print, x
   catch,/cancel  ; <--- don't forget this
end
```

```
IDL> .run test
% Compiled module: TEST.
IDL> test
% TEST: OPENR: Error opening file. Unit: 100, File: newfile.dat; Closing file...
```

You can also use ON_IOERROR, but that's basically a fancy way of saying GOTO. I reckon the CATCH method is cleaner.

BTW, this should work for any error. If you want to flag your own errors you can just do something like:

```
   if( ...some error condition...) then $
     message, 'Some error condition occurred', /noname, /noprint
```

and control will be sent to the CATCH handler.

cheers,

paulv

--
Paul van Delst   Ride lots.
CIMSS @ NOAA/NCEP/EMC   Eddy Merckx
Ph: (301)763-8000 x7748
Fax:(301)763-8545

Subject: Re: error reading file question
Posted by David Fanning on Tue, 07 Nov 2006 19:34:35 GMT
View Forum Message <> Reply to Message

Paul van Delst writes:

```
>  pro test
>     ; Setup error handler
>     catch, ierr
>     if(ierr ne 0)then begin
>       catch, /cancel
>       message, !error_state.msg + '; Closing file...', /continue
>       if(n_elements(fileid) ne 0) then free_lun, fileid
>       return
>     endif
>     ; Read non-existant file
>     openr, fileid, 'newfile.dat', /get_lun
>     x=0.0
>     readu, fileid, x
>     free_lun, fileid
>     print, x
>     catch,/cancel  ; <--- don't forget this
>  end
```

I wouldn't worry about forgetting to cancel
the Catch in the next to last line. CATCH
is only in effect for that program module
anyway, and you are, uh, about to leave
there. :-)

Cheers,

David

P.S. You *should* cancel the CATCH in the first
line of the error handling code, though, because--
Heaven forbid!--if you have an error in your error
handling code you are going to notice!! :-)
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: error reading file question
Posted by Paul Van Delst[1] on Tue, 07 Nov 2006 20:36:21 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> Paul van Delst writes:
>
```
>>  pro test
>>     ; Setup error handler
>>     catch, ierr
>>     if(ierr ne 0)then begin
>>       catch, /cancel
>>       message, !error_state.msg + '; Closing file...', /continue
>>       if(n_elements(fileid) ne 0) then free_lun, fileid
>>       return
>>     endif
>>     ; Read non-existant file
>>     openr, fileid, 'newfile.dat', /get_lun
>>     x=0.0
>>     readu, fileid, x
>>     free_lun, fileid
>>     print, x
>>     catch,/cancel  ; <--- don't forget this
>>  end
```
>
> I wouldn't worry about forgetting to cancel
> the Catch in the next to last line. CATCH
> is only in effect for that program module
> anyway, and you are, uh, about to leave
> there. :-)

Is that true? I could be wrong, but I recall some sort of issue with not canceling the CATCH before leaving the routine...somehow it was still active. Or something. You know.... ehem.

> Cheers,
>
> David
>
> P.S. You *should* cancel the CATCH in the first
> line of the error handling code, though, because--
> Heaven forbid!--if you have an error in your error
> handling code you are going to notice!! :-)

Yes. Hello infinite loopage. Modifiying the above to:

```
pro test
   catch, ierr
   if(ierr ne 0)then begin
     catch, /cancel
     message, !error_state.msg + '; Closing file...', /continue
     free_lun, fileid
```

```
    return
  endif
  message, 'a pretend error!',/noname,/noprint
  openr, fileid, 'newfile.dat', /get_lun
  x=0.0
  readu, fileid, x
  free_lun, fileid
  print, x
end
```

gives me:

```
IDL> .run test
% Compiled module: TEST.
IDL> test
% TEST: a pretend error!; Closing file...
% FREE_LUN: Variable is undefined: FILEID.
% Execution halted at: TEST               6 /export/lnx374/wd20pd/scratch/test.pro
%                       $MAIN$
```

Commenting out the catch,/cancel in the error handler portion gives:

```
IDL> .run test
% Compiled module: TEST.
IDL> test
% TEST: a pretend error!; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
% TEST: FREE_LUN: Variable is undefined: FILEID.; Closing file...
....etc. ad infinitum.
```

paulv

--

Paul van Delst Ride lots.
CIMSS @ NOAA/NCEP/EMC Eddy Merckx
Ph: (301)763-8000 x7748
Fax:(301)763-8545

Subject: Re: error reading file question
Posted by David Fanning on Tue, 07 Nov 2006 21:02:03 GMT
View Forum Message <> Reply to Message

Paul van Delst writes:

> Is that true? I could be wrong, but I recall some sort of issue with not canceling the
> CATCH before leaving the routine...somehow it was still active. Or something. You know....
> ehem.

Well, I wouldn't say "wrong". Perhaps "misinformed." :-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")