Subject: Re: programmatically detect problems in idl code
Posted by Benjamin Hornberger on Mon, 01 Aug 2005 15:23:42 GMT
View Forum Message <> Reply to Message

Mathieu Malaterre wrote:
>
> Is there a way to compile this program so that idl actually return
> to the parent process an error occur ?
>

on_error, 2

at the beginning of the routine. Have a look at "on_error", "on_ioerror"
and "catch" in the IDL help.

Good luck,
Benjamin

Subject: Re: programmatically detect problems in idl code
Posted by Mathieu Malaterre on Mon, 01 Aug 2005 15:32:05 GMT
View Forum Message <> Reply to Message

Benjamin Hornberger wrote:
> Mathieu Malaterre wrote:
>
>>
>> Is there a way to compile this program so that idl actually return
>> to the parent process an error occur ?
>>
>
> on_error, 2
>
> at the beginning of the routine. Have a look at "on_error", "on_ioerror"
> and "catch" in the IDL help.

Thanks Benjamin,

But I am looking for a command line option. I don't know ahead there is
a mistake in my idl code. I want to write a daemon that would check for
me all the code I am writting.

For a typical usage, let say I am writting code on win32 system, but
at the same time a linux system is running which observe the code I am
writting. All I need to do is run some command on the linux system to
check the code is ok: WITHOUT changing the code.

The analogy for C/C++, is that when the compiler find an error it

returns to the parent process with a value different from 0.

Or the worst case is there is a missing file, I want idl to tell me it could not find the missing file:

$ echo inexistant_file | idl
$ echo $?
0

If there is no such thing I'll have to parse idl output and check for strings like: 'Syntax Error', 'Execution halted' ... which is kind of a pain.

I hope to be clear this time, thanks again
Mathieu

Subject: Re: programmatically detect problems in idl code
Posted by b_efremova@yahoo.com on Mon, 01 Aug 2005 16:50:42 GMT
View Forum Message <> Reply to Message

Hi,
There's a system variable !error_state
You can see the help for it, and you can also type

help,!error_state,/structure

to see its fields.
Cheers
Boryana

Subject: Re: programmatically detect problems in idl code
Posted by Michael Wallace on Mon, 01 Aug 2005 17:57:38 GMT
View Forum Message <> Reply to Message

If I understand what you're asking, you'd like an appropriate exit status returned to the operating system such that you can check that number and determine if you IDL code executed successfully. Is this what you're after?

By default, IDL will always return an exit status of 0 regardless of whether IDL was successful or not. You can set your own exit status numbers with the exit command. For example,

exit, STATUS = 1

will immediately exit your program and the operating system will see a value of 1 returned from the process. You can simply put a line like this at the end of your main program. If all of your code was successful, the OS will see the value 1. If there was an error anywhere along the way, you'll see the value 0.

If you want to get fancier, you can use the on_error, on_ioerror, and catch commands as mentioned in a previous message. These commands will enable you to catch certain errors and then exit your program with a custom status code for each error type. One word of warning however -- make sure you only put the exit codes in your main program and not in any of your subroutines. Doing so can lead to code that is very difficult to debug and can be very error prone. It's just good practice to throw the error back out to the main level before exiting.

-Mike

Subject: Re: programmatically detect problems in idl code
Posted by Mathieu Malaterre on Mon, 01 Aug 2005 19:09:10 GMT
View Forum Message <> Reply to Message

b_efremova@yahoo.com wrote:
> Hi,
> There's a system variable !error_state
> You can see the help for it, and you can also type
>
> help,!error_state,/structure
>
> to see its fields.

This is actually pretty good. I can run command like:

echo ".compile file.pro \n print,\!ERROR_STATE" | idl

and all I need to check is the output. Not fancy but should be ok.

thanks
Mathieu
