Subject: Re: subverting IDL builtin variables !FORMYOWNPURPOSES
Posted by Michael Galloy on Sat, 19 Feb 2011 03:34:56 GMT
View Forum Message <> Reply to Message

On 2/18/11 5:41 PM, Ed Hyer wrote:
> This is a terrible idea, but I'll feel better about doing it one of
> the Hard Ways once I've committed this awful cheat to posterity.
>
> The package has ~100 subroutines, and each subroutine has various
> types of output. All of this output ends up in logs, but now that the
> whole creature is built, it's time to set different levels of
> verbosity. My simple scheme is like this:
>   DEBUG=0; put nothing in the log except fatal errors;
>   DEBUG=1; include warnings and limited diagnostics;
>   DEBUG=2; include full diagnostics, performance-related
> information, the kitchen sink.
>
> Now, I can think of three ways to do this:
> 1) Pass a VERBOSITY keyword from the top level through all of the
> subroutines. I'm not going to change 100 headers to add this (though I
> am going to change ~200 PRINT statements to IF(VERBOSITY gt XX) THEN
> PRINT).
> 2) Create a common block for the VERBOSITY level. I've never done
> this, but it seems like the right solution for this problem.
> 3) Put the VERBOSITY into a !VARIABLE that isn't being used for
> anything else. There are plenty to choose from, especially since this
> package doesn't actually generate any graphics, etc.
>
> Solution #3 is so easy... so wrong... so easy. Oh well.
> Have a great weekend, everybody.

I ended up using common blocks for my logging framework, but system variables would work just as well.

See MG_LOG and MGffLogger in the dist_tools for the way I did it:

http://docs.idldev.com/dist_tools/

By the way, you don't have to subvert a pre-existing system variable, you can create your own with DEFSYSV.

Mike
--
www.michaelgalloy.com
Research Mathematician
Tech-X Corporation