Subject: Re: compilation of subroutines without resetting calling sequence
Posted by Kenneth P. Bowman on Mon, 05 Jul 2010 20:21:34 GMT
View Forum Message <> Reply to Message

In article
<5a442a1b-8386-4d8e-807e-cb2890f75907@7g2000prh.googlegroups.com>,
 Ed Hyer <ejhyer@gmail.com> wrote:

> OK, compiled procedure 'A' calls function 'B', but spends most of its
> time in functions 'C-Z'.
>
> Function 'B' has a math error somewhere, which I am trying various
> things to get rid of. By setting !EXCEPT=2, I have lots of information
> on where the screwup is occurring.
>
> Once I get the spew from !EXCEPT=2, I manually interrupt the program,
> which is at that point working through functions 'C-Z'. I make changes
> to subroutine 'B', recompile it, and then '.continue' to see if the
> changes worked next time through subroutine 'B'.
>
> Except that doesn't work. '.compile' returns no exception, and
> '.continue' works like one expects, but it won't actually recompile
> the routine.
>
> So, evidently it's the top-level routine that must be recompiled in
> order to get changes to subroutines into effect, but if someone wants
> to provide a more coherent explanation of what IDL is doing here, I'd
> love to hear it.

Instead of just setting !EXCEPT (which is often all that is needed
in simple cases), use CHECK_MATH in B to check the math error status and
stop execution inside function B. (Don't forget to read the notes
at the bottom or the CHECK_MATH help page.)

Then, if you recompile B, IDL will return to the top level and you
can run the program again.

Ken Bowman