Subject: Re: main or procedure
Posted by Conor on Wed, 20 Jun 2007 12:39:14 GMT
View Forum Message <> Reply to Message

On Jun 20, 8:20 am, David Fanning <n...@dfanning.com> wrote:
> suj...@gmail.com writes:
>> When I .compile a procedure, it says,
>> You compiled a main program while inside a procedure. Returning.
>
>> And all the information before the compilation was lost.
>
>> How to fix it?
>
> You need to do two things: (1) Add more error handling
> to your code, so that when a program crashes it goes
> back to the main program level (ON_ERROR, 1), and (2)
> learn to type RETALL *immediately* when your program
> crashes, and certainly before you do any more compiling.
>
> 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.")

What's happening is the effects of variable scope. IDL has various levels of "scope" for every variable it creates. When you initially launch IDL, you are on the "MAIN" level. Everytime you run a program or function, that program runs in its own separate scope, that is below the "level" of the main level. Most importantly, it is entirely separate from the main level with an independent set of variables. So, if you have a variable named 'asdf' in your main level, and then you launch a program, the program in question won't be able to access that same 'asdf' variable, because your program runs in a different scope. By default, the variables used in a program can't be used in the main level, and vice-versa. Now, when a program encounters an error and quits it returns you to the idl prompt, but it returns you at the level of the program that had the error, not at the main level. What this means is that when your program encounters an error you won't immediately have access to the variables you declared initially in the main level. In your case, another important point is that when IDL is inside the scope of a program that had an error, and that program is re-compiled, then IDL automatically returns you to the main level and you automatically lose any variables that had existed in the program's scope. That's what is happening to you. At some

point you are running a routine that gives an error. Then, you are
continuing to use the IDL command line, creating variables and doing
whatever it is you do. When you re-compile the program that had the
original error, IDL automatically returns you to the main level, and
all those variables you created go out of scope and are deleted. The
solution is quite simple. Just do what David pointed out. If you set
the ON_ERROR command inside your program like he noted, then idl will
return you to the main level when the program encounters an error,
rather than to the program level. This isn't the best solution during
development though, since returning to the main level means you lose
all your program variables and therefore you can't examine the state
of your program to see where the problem was. The other solution is
to simply type 'retall' before you resume working with the IDL command
line. 'retall' will return you to the main programming level, once
again giving you access to any variables you had created previously.