Subject: Re: Moving back a level...
Posted by JD Smith on Thu, 24 Feb 2005 01:55:51 GMT
View Forum Message <> Reply to Message

On Tue, 22 Feb 2005 21:48:37 +0100, Ben Panter wrote:

> Hi Tim,
>
>> Here goes. If I put a STOP in a subroutine (or if I've encountered an
>> error and I'm stuck inside one) and I'd like to see what the variables
>> look like in the calling routine using HELP, is there any way to move
>> back to the calling routine but remain "stopped"? I can get back to
>> the calling routine by entering a return (sometimes I need a few
>> .skip's) but then it's off and running. This all sounds like I should
>> learn to make use of some type of debugging software associated with
>> IDL but I've never looked into any such thing. Any advice on where to
>> start would be hugely, hugely appreciated.
>
> I have an answer to a question that you might be asking, but I'm not
> entirely sure if I understand you!
>
> My answer would be to fire up idlde instead of idl, and run your code.
> Then there are a few buttons at the top which allow you to move up and
> down the calling stack from where you stopped. The variable watch area
> then changes to the variables in that level.
>
> Access to the variable watch window is one thing that I miss when I use
> idlwave...

Well, you shouldn't miss it, because IDLWAVE can traverse the calling
stack quite easily when you're stopped (at a breakpoint, error, etc.).

By default C-C C-d C-Up and Down will move you anywhere up and down
the stack, and you'll see your location in the minibar: [0:MyPro],
[-1:CallingPro] etc. As you navigate the stack, IDLWAVE brings you to
the exact line of source code through which the stack threads. Then
you can use the examine commands (a fun one is C-S-middle-click) which
will let you see variables at any place in the stack.

Suppose you do a lot of widget programming, and you'd like, at a
click, to know if a widget id is valid any longer. Just
C-S-middle-click it and select "Widget Valid" from the examine menu.
And the sky's the limit for your own custom examine commands: just
customize Idlwave Shell Command Setup->Idlwave Shell Examine Alist,
and make up your own crazy examine commands, like:

```
 IsOdd
 print,___ & 1L?"Odd":"Even"
```

If you have electric debug mode active (which happens by default on breakpoints, and can be enabled on errors as well), all these features are a single keystroke away. And you're not limited to examining single variables: you can examine arbitrary expressions, function results, essentially anything that IDL can evaluate.

I use this all the time when a routine stops at an error. Examine a few variables in the current routine to see that all is well, and then jump up the stack one level, examine some variables there to see if the inputs were kosher, etc. A real time saver.

JD