Subject: Re: directing control after program interruption
Posted by Craig Markwardt on Thu, 18 May 2000 07:00:00 GMT
View Forum Message <> Reply to Message

"D. Mattes" <dmattes@u.washington.edu> writes:

> hello IDL power users: i have a long minimization process running under
> IDL. i would like the user to be able to interrupt the process at any
> time. once the user interrupts by typing ctrl-c, i would like to direct
> program control to a different part of the routine. the only way i know
> to resume the program run is the .continue executive command, but that
> just continues on with the next statement, whereas i would like to branch
> to the end of the routine. is there a system variable that is set when a
> user interrupts program flow? or does this fall into exception handling?

I do not think IDL is capable of what you are asking. On the other hand, you can check for another control character to use for a termination criterium. The GET_KBRD() function will poll the keyboard without pausing, so you can do this periodically (i.e., every iteration). Unfortunately, it will consume the keyboard buffer contents, but that's the price you pay.

Here is how I do it in MPFIT, which looks for control-G (= ASCII 7):

```
k = get_kbrd(0)
if k EQ string(byte(7)) then begin
   message, 'WARNING: minimization not complete', /info
   print, 'Do you want to terminate this procedure? (y/n)', $
     format='(A,$)'
   k = ''
   read, k
   if strupcase(strmid(k,0,1)) EQ 'Y' then begin
      message, 'WARNING: Procedure is terminating.', /info
      mperr = -1
   endif
endif
```

MPFIT is a least squares fitter available here:

http://cow.physics.wisc.edu/~craigm/idl/idl.html

Good luck!
Craig

--
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu

Subject: Re: directing control after program interruption
Posted by Mark Hadfield on Fri, 19 May 2000 07:00:00 GMT
View Forum Message <> Reply to Message

"D. Mattes" <dmattes@u.washington.edu> wrote in message
news:Pine.A41.4.21.0005181339010.116032-100000@dante27.u.was hington.edu...
> hello IDL power users: i have a long minimization process running under
> IDL. i would like the user to be able to interrupt the process at any
> time. once the user interrupts by typing ctrl-c, i would like to direct
> program control to a different part of the routine. the only way i know
> to resume the program run is the .continue executive command, but that
> just continues on with the next statement, whereas i would like to branch
> to the end of the routine. is there a system variable that is set when a
> user interrupts program flow? or does this fall into exception handling?

I have a class called MGHwaiter, see

http://katipo.niwa.cri.nz/~hadfield/gust/software/idl/mghwai ter__define.pro

The idea is that, before a long calculation in IDL code, you create an MGHwaiter object, then in the main loop of the code you add calls to the object's Yield method. (Each call takes very little time so you can afford to have a lot of them.) Afterwards you destroy the object. Of course you can't add calls to Yield inside IDL internal procedures so there are some time-consuming operations it is not useful for.

Each MGHwaiter object pops up a widget base with 3 buttons: Suspend, Resume & Abort. Suspend suspends execution (goes into a loop), Resume reverses Suspend, and Abort calls the MESSAGE procedure to throw an exception. I've always just cleaned up manually at this point, but you could surround the Yield calls with a CATCH handler and do whatever you want.

It's all rather clunky, really, but it might be useful for you.

BTW, The MGHwaiter class descends from a procedure that I wrote in 1993. At that time, IDL on Windows took over the machine completely on long calculations and the main function of the procedure was to call a Windows API function called Yield and so allow other applications to get a look-in. That isn't necessary any more, but the MGHwaiter still has the welcome side-effect of letting IDL update its display, thus avoiding the big-blank-window effect on Windows NT.

A detail: MGHwaiter has a rather old-fashioned structure. It calls WIDGET_EVENT directly & doesn't use XMANAGER. It requires the XMENU routine,

which is obsolete, I think. I think it should be possible to modernise it,
though some calls to WIDGET_EVENT would still be needed.

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield/
National Institute for Water and Atmospheric Research
PO Box 14-901, Wellington, New Zealand