Subject: Re: endless loops suck
Posted by thompson on Tue, 04 Feb 2003 17:11:21 GMT
View Forum Message <> Reply to Message

It must be platform-specific. On my Alpha workstation, running IDL/v5.4, I can break out of all of these loops with control-C. On the other hand, with the same version of IDL under Windows 95, I could break out of happy_to_break with Control-Break, but with doomed_to_continue, I had to use Control-Alt-Delete to end the IDL task!

On a somewhat related note, I was reminded of another program where putting in a short call to wait turned out to be beneficial. A telemetry processing program I had written was eating up all the CPU on the machine, so that other processes were not getting their fair share. I'm not sure why this was; perhaps because it was so I/O intensive the computer was boosting its priority. In any case, simply adding a "WAIT, 0.005" statement for each packet solved the problem, without significantly impacting the speed of the program itself. The program ran just as fast as before when it had the machine to itself, but allowed other processes to also get their fair share of the clock cycles.

Bill Thompson

"Pete" <peterscarth@despammed.com> writes:

> Hi,
> I've come across this problem before when updating plots.

> This will not not break:

```
> PRO doomed_to_continue
> a=0L
>    While (1) DO BEGIN
>    a=a+1
>     ENDWHILE
> END
```

> This will break:

```
> PRO happy_to_break
> a=0L
>    While (1) DO BEGIN
>    a=a+1
>    wait,0.0001
>    ENDWHILE
> END
```

> as will this:

> PRO happy_to_break_2
> a=0L
>   While (1) DO BEGIN
>   a=a+1
>   print,''
>   ENDWHILE
> END

> At times, I've had to put in a wait statement after a plot statement to make
> it display, if the plot statement is in a loop and if there is no other way
> to interrupt the excecution of the loop. Weird!

> Peter

> "Thomas Gutzler" <tgutzler@ee.uwa.edu.au> wrote in message
> news:3E3DDF77.3040908@ee.uwa.edu.au...
>> good morning,
>>
>> David Fanning wrote:
>>>
>>> So I admit, "infinite" WHILE loops *can* be interrupted, as long as
>>> they use BEGIN ... END statement blocks. Try this:
>>>
>>> PRO Test
>>> While 1 DO BEGIN
>>>   Print, 'Test before you Post!'
>>> ENDWHILE
>>> END
>>
>> I thought about this and figured out that I have a loop with begin
>> and end. It looks like that:
>>
>> PRO SnakeInterp,xi,yi,dmax,dmin
>>   ; x,y are arrays of coordinates
>>   ; dmin, dmax are preferred min and max distance between coordinates
>>   [...]
>>   ; WHILE (MAX(d) GT dmax) DO BEGIN ; this would be correct
>>   WHILE (1) DO BEGIN ; this obviously loops to dead
>>   idx0 = (d GT dmax)
>> => z = SnakeIndex(idx0) ; debugger is here
>>   p = INDGEN(N+1)+1
>>   xi = InterPol([xi,xi[0]],p,z)
>>   yi = InterPol([yi,yi[0]],p,z)
>>   N = N_ELEMENTS(xi)
>>   d = abs([xi[1:N-1],xi[0]] - xi) + abs([yi[1:N-1],yi[0]] - yi)

```
>>    ENDWHILE
>>  END
>>
>>  FUNCTION snakeindex, idx
>>    ; SNAKEINDEX  Create index for adpative interpolating the snake
>>    arr = idx
>>    N = N_ELEMENTS(xi)
>>    arr = (TRANSPOSE([[arr],[INTARR(N)+1]]))[0:2*N-2]
>>    y = DOUBLE(WHERE([1,arr] EQ 1))/2+1
>>    RETURN, y
>>  END
>>
>>  I pressed "Step out", then CTRL+BREAK (10 times) then CTRL+c (another 10
>>  times) then I made myself a tea and read some news. After half an hour
>>  taskmanager was my friend again :(
>>  I thought maybe the compiler needs a little bit of time to finish
>>  currently running processes before it polls the keyboard again and breaks.
>>
>>>  On my Windows machine I can interrupt this program (after I've
>>>  learned my lesson) by doing a CNTL-Break.
>>
>>  I could interrupt the Print, 'Test before you Post!' loop with this, yes.
>>
>>  Perhaps I just should pay more attention to my sourcecode.
>>
>>  Tom
>>
```
