Subject: Re: endless loops suck
Posted by Timm Weitkamp on Fri, 31 Jan 2003 13:02:30 GMT
View Forum Message <> Reply to Message

Today at 10:54 +0800, Thomas Gutzler wrote:

> Is there any way to stop this while debuging ?
>
> pro loop_it
> a = 1
> while (1) do a = a + 1
> end

Thomas,

I don't know if this helps a lot, but one way that will allow you
to stop it with CTRL-c is of course to modify it into

```
pro loop_it
  a = 1
  while (1) do begin
    a = a + 1
  endwhile
end
```

Cheers,
Timm

--
Timm Weitkamp <http://people.web.psi.ch/weitkamp>

Subject: Re: endless loops suck
Posted by Pavel A. Romashkin on Fri, 31 Jan 2003 19:11:33 GMT
View Forum Message <> Reply to Message

I think you can stop a loop if you use BEGIN ... END syntax. Without it,
the loop is compiled into one command and is not interrupted until
finished; with it, each item inside is a separate command, so IDL will
check for breaks every time. I am not sure that it affects the speed at all.
Cheers,
Pavel

Thomas Gutzler wrote:
>
> Hey :)
>
> Is there any way to stop this while debuging ?

```
>  I tried CTRL+C and CTRL+BREAK which sometimes helps but not in this case.
>
>     pro loop_it
>       a = 1
>       while (1) do a = a + 1
>     end
>
>     pro kill_idl
>       print, 'start'
>       ; debugger is here and you pressed accidently 'F10'
>  =>   loop_it
>       print, 'stop'
>     end
>
>  *loop*,
>    Tom
```

Subject: Re: endless loops suck
Posted by David Fanning on Fri, 31 Jan 2003 19:38:22 GMT
View Forum Message <> Reply to Message

Pavel A. Romashkin (pavel_romashkin@hotmail.com) writes:

> I think you can stop a loop if you use BEGIN ... END syntax. Without it,
> the loop is compiled into one command and is not interrupted until
> finished; with it, each item inside is a separate command, so IDL will
> check for breaks every time. I am not sure that it affects the speed at all.

I don't think so. The whole point of the BEGIN ... END syntax
is to make the statements inside appear to be a single statement
to the compiler. The only recourse is to let the loop go to
completion. With an infinite loop, well, they don't call
them "infinite" for no reason. :-)

Cheers,

David
--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: endless loops suck

Posted by David Fanning on Sat, 01 Feb 2003 16:14:08 GMT
View Forum Message <> Reply to Message

I wrote yesterday:

> I don't think so. The whole point of the BEGIN ... END syntax
> is to make the statements inside appear to be a single statement
> to the compiler. The only recourse is to let the loop go to
> completion. With an infinite loop, well, they don't call
> them "infinite" for no reason. :-)

Apparently I owe an apology to Pavel. It has been
pointed out to me by a long-time IDL newsgroup reader,
who wishes not to embarrass me, that I don't know what
the hell I'm talking about when it comes to this topic.
(Have you noticed that IDL newsgroup readers are exceptionally
kind to one another? It's one of the things that makes
this newsgroup so much fun to hang out in. :-)

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
```

On my Windows machine I can interrupt this program (after
I've learned my lesson) by doing a CNTL-Break.

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: endless loops suck
Posted by Thomas Gutzler on Mon, 03 Feb 2003 03:18:15 GMT
View Forum Message <> Reply to Message

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

Subject: Re: endless loops suck
Posted by Pete[2] on Mon, 03 Feb 2003 23:36:58 GMT
View Forum Message <> Reply to Message

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
>

Subject: Re: endless loops suck
Posted by Thomas Gutzler on Tue, 04 Feb 2003 01:57:33 GMT
View Forum Message <> Reply to Message

Pete wrote:
> 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
```

Weird!
I'm starting to put some waits meta-info-prints in my loops. This spawns a new question:
How is this possible ?

```
FOR i=0,4 DO PRINT, i, ' ', /NONEWLINE
                    ^^^^^^^^^^ this may be a problem
```

Result:
0 1 2 3 4

I couldn't find anything usefull in the help. If this isn't possible I have to use the wait :/

thx,
  Tom

Subject: Re: endless loops suck
Posted by Timm Weitkamp on Tue, 04 Feb 2003 09:14:07 GMT
View Forum Message <> Reply to Message

Today at 09:57 +0800, Thomas Gutzler wrote:

```
>  How is this possible ?
>
>  FOR i=0,4 DO PRINT, i, ' ', /NONEWLINE
>                      ^^^^^^^^^^ this may be a problem
>  Result:
>  0 1 2 3 4
```

Use a format string (see also "Format codes" in the online help), like this:

```
  FOR i=0,4 DO PRINT, i, FORMAT='($,I1," ")'
  PRINT
```

Cheers,
Timm

--
Timm Weitkamp <http://people.web.psi.ch/weitkamp>

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
>>

Subject: Re: endless loops suck
Posted by thompson on Tue, 04 Feb 2003 17:13:17 GMT
View Forum Message <> Reply to Message

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

> 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!

Have you tried using EMPTY instead?

Bill Thompson
