Subject: Re: GUI interface update issues
Posted by Benjamin Luethi on Tue, 21 Jun 2005 16:59:44 GMT
View Forum Message <> Reply to Message

Hi,

No clue what is causing this, but a simple workaround might be to introduce
a (very short) pause after you update the progress bar:

wait, 0.01

Hope this helps,
Ben

On Tue, 21 Jun 2005 18:23:39 +0200, Rick Towler
<rick.towler@nomail.noaa.gov> wrote:

> Hello group,
>
> I have an interesting problem with gui redraw on WinXP which maybe
> someone can shed some light on.
>
> I have an application that uses David's progress bar object. We have
> some potentially long calculation times and it is important to give a
> little feedback. It worked great until I added the ability to call a
> new calculation routine written in C as a dlm. If I run the application
> the old way (all functions written in IDL) the progress bar and GUI
> interface function normally. But if I run the application using the
> external routine eventually the GUI interface stops redrawing, the drop
> down menu text disappears, and the progress bar fails to update. The
> application runs and when calculations are finished it returns to normal
> (progress bar is destroyed and interface works as expected) but there is
> no feedback while running. A problem since the new calculations can
> take hours and hours and it is nice to see where it is in the process.
>
> The program is structured like so:
>
> ;-------------
> setup stuff
>
> for loop begin
>
> progressBar->update
>
> if (use_dlm) then begin
> myDLM_proc, param1, param2, OUT1=out1, OUT2=out2...
> endif else begin
> IDLbased_proc, param1, param2, OUT1=out1, OUT2=out2...

>   endelse
>
>   for loop begin
>
>   if (use_dlm) then begin
>   anotherDLM_proc, param1, param2, OUT1=out1,OUT2=out2...
>   endif else begin
>   anotherIDLbased_proc, param1, param2, OUT1=out1,OUT2=out2...
>   endelse
>
>   endfor
>
> endfor
> ;------------
>
> Actually a lot more is going on but you get the idea.
>
> And to elaborate on what I mean by "eventually the gui stops
> responding". If I start the application and run a short calculation (~2
> minutes) the first time the gui functions normally. But with every
> subsequent run the progress bar moves maybe 20% of the way then I lose
> the gui. If I close and restart the application the same thing happens,
> first one works, then problems). If I run a long calculation (hours)
> the progress bar never moves past the first tick (maybe 1%).
>
> Why does IDL stop updating the gui? Any ideas? While the functions in
> the dlm are compute intensive, they aren't particularly complicated and
> only rely on some simple macros in an .h file. Just a *lot* of looping
> over a moderate amount of data.
>
> -Rick

--
Using Opera's revolutionary e-mail client: http://www.opera.com/m2/

Subject: Re: GUI interface update issues
Posted by Haje Korth on Tue, 21 Jun 2005 17:05:26 GMT
View Forum Message <> Reply to Message

Rick,
my question would be whether the same things happens on UNIX OS. On Windows
I have found IDL has the terrible habbit of taking over with unpredictable
results such as GUIs not reacting at all. If the UNIX version works fine,
which I suspect it will, your only hope is some input from the developers.

Haje

"Rick Towler" <rick.towler@nomail.noaa.gov> wrote in message
news:d99f2fohk1@news.nems.noaa.gov...
> Hello group,
>
> I have an interesting problem with gui redraw on WinXP which maybe someone
> can shed some light on.
>
> I have an application that uses David's progress bar object. We have some
> potentially long calculation times and it is important to give a little
> feedback. It worked great until I added the ability to call a new
> calculation routine written in C as a dlm. If I run the application the
> old way (all functions written in IDL) the progress bar and GUI interface
> function normally. But if I run the application using the external
> routine eventually the GUI interface stops redrawing, the drop down menu
> text disappears, and the progress bar fails to update. The application
> runs and when calculations are finished it returns to normal (progress bar
> is destroyed and interface works as expected) but there is no feedback
> while running. A problem since the new calculations can take hours and
> hours and it is nice to see where it is in the process.
>
> The program is structured like so:
>
> ;-------------
> setup stuff
>
> for loop begin
>
> progressBar->update
>
> if (use_dlm) then begin
> myDLM_proc, param1, param2, OUT1=out1, OUT2=out2...
> endif else begin
> IDLbased_proc, param1, param2, OUT1=out1, OUT2=out2...
> endelse
>
> for loop begin
>
> if (use_dlm) then begin
> anotherDLM_proc, param1, param2, OUT1=out1,OUT2=out2...
> endif else begin
> anotherIDLbased_proc, param1, param2, OUT1=out1,OUT2=out2...
> endelse
>
> endfor
>

> endfor
> ;------------
>
> Actually a lot more is going on but you get the idea.
>
> And to elaborate on what I mean by "eventually the gui stops responding".
> If I start the application and run a short calculation (~2 minutes) the
> first time the gui functions normally. But with every subsequent run the
> progress bar moves maybe 20% of the way then I lose the gui. If I close
> and restart the application the same thing happens, first one works, then
> problems). If I run a long calculation (hours) the progress bar never
> moves past the first tick (maybe 1%).
>
> Why does IDL stop updating the gui? Any ideas? While the functions in
> the dlm are compute intensive, they aren't particularly complicated and
> only rely on some simple macros in an .h file. Just a *lot* of looping
> over a moderate amount of data.
>
> -Rick

Subject: Re: GUI interface update issues
Posted by Benjamin Hornberger on Tue, 21 Jun 2005 18:01:56 GMT
View Forum Message <> Reply to Message

I know nothing about DLMs, but we also had a problem with David's progressbar recently. We had a continuous timer running in the background, with the consequence that the progressbar wouldn't recognize that the user clicked the cancel button. The problem could be solved by making progressbar__define.pro use xmanager rather than widget_event. If your program also has events going on in the background, the problem might be similar???

Benjamin

Rick Towler wrote:
> Hello group,
>
> I have an interesting problem with gui redraw on WinXP which maybe
> someone can shed some light on.
>
> I have an application that uses David's progress bar object. We have
> some potentially long calculation times and it is important to give a
> little feedback. It worked great until I added the ability to call a
> new calculation routine written in C as a dlm. If I run the application
> the old way (all functions written in IDL) the progress bar and GUI
> interface function normally. But if I run the application using the

> external routine eventually the GUI interface stops redrawing, the drop
> down menu text disappears, and the progress bar fails to update. The
> application runs and when calculations are finished it returns to normal
> (progress bar is destroyed and interface works as expected) but there is
> no feedback while running. A problem since the new calculations can
> take hours and hours and it is nice to see where it is in the process.
>
> The program is structured like so:
>
> ;----------
> setup stuff
>
> for loop begin
>
> progressBar->update
>
> if (use_dlm) then begin
> myDLM_proc, param1, param2, OUT1=out1, OUT2=out2...
> endif else begin
> IDLbased_proc, param1, param2, OUT1=out1, OUT2=out2...
> endelse
>
> for loop begin
>
> if (use_dlm) then begin
> anotherDLM_proc, param1, param2, OUT1=out1,OUT2=out2...
> endif else begin
> anotherIDLbased_proc, param1, param2, OUT1=out1,OUT2=out2...
> endelse
>
> endfor
>
> endfor
> ;----------
>
> Actually a lot more is going on but you get the idea.
>
> And to elaborate on what I mean by "eventually the gui stops
> responding". If I start the application and run a short calculation (~2
> minutes) the first time the gui functions normally. But with every
> subsequent run the progress bar moves maybe 20% of the way then I lose
> the gui. If I close and restart the application the same thing happens,
> first one works, then problems). If I run a long calculation (hours)
> the progress bar never moves past the first tick (maybe 1%).
>
> Why does IDL stop updating the gui? Any ideas? While the functions in
> the dlm are compute intensive, they aren't particularly complicated and
> only rely on some simple macros in an .h file. Just a *lot* of looping

> over a moderate amount of data.
>
> -Rick

Subject: Re: GUI interface update issues
Posted by Rick Towler on Tue, 21 Jun 2005 18:07:03 GMT
View Forum Message <> Reply to Message

Haje Korth wrote:
> Rick,
> my question would be whether the same things happens on UNIX OS. On Windows
> I have found IDL has the terrible habbit of taking over with unpredictable
> results such as GUIs not reacting at all. If the UNIX version works fine,
> which I suspect it will, your only hope is some input from the developers.

Hey Haje,

Well, I suppose I could install the IDL VM on a linux box and compile everything... Doesn't help me too much though if I find it's o.k. on linux. I too have had issues with the windows GUI but usually it is that IDLDE becomes unresponsive but my applications are o.k.

Upon further investigation I have determined that the dlm angle is wrong. The issue occurs with or without calling the dlm but it only occurs with one of the many models in the application. While the calculations are basically the same, that model has an additional nested for loop in the IDL code.

<sigh>

-Rick

>
> "Rick Towler" wrote in message
> news:d99f2fohk1@news.nems.noaa.gov...
>
>> Hello group,
>>
>> I have an interesting problem with gui redraw on WinXP which maybe someone
>> can shed some light on.
>>
>> I have an application that uses David's progress bar object. We have some
>> potentially long calculation times and it is important to give a little
>> feedback. It worked great until I added the ability to call a new
>> calculation routine written in C as a dlm. If I run the application the

>> old way (all functions written in IDL) the progress bar and GUI interface
>> function normally. But if I run the application using the external
>> routine eventually the GUI interface stops redrawing, the drop down menu
>> text disappears, and the progress bar fails to update. The application
>> runs and when calculations are finished it returns to normal (progress bar
>> is destroyed and interface works as expected) but there is no feedback
>> while running. A problem since the new calculations can take hours and
>> hours and it is nice to see where it is in the process.
>>
>> The program is structured like so:
>>
>> ;-------------
>> setup stuff
>>
>> for loop begin
>>
>>   progressBar->update
>>
>>   if (use_dlm) then begin
>>   myDLM_proc, param1, param2, OUT1=out1, OUT2=out2...
>>   endif else begin
>>   IDLbased_proc, param1, param2, OUT1=out1, OUT2=out2...
>>   endelse
>>
>>   for loop begin
>>
>>   if (use_dlm) then begin
>>   anotherDLM_proc, param1, param2, OUT1=out1,OUT2=out2...
>>   endif else begin
>>   anotherIDLbased_proc, param1, param2, OUT1=out1,OUT2=out2...
>>   endelse
>>
>>   endfor
>>
>> endfor
>> ;------------
>>
>> Actually a lot more is going on but you get the idea.
>>
>> And to elaborate on what I mean by "eventually the gui stops responding".
>> If I start the application and run a short calculation (~2 minutes) the
>> first time the gui functions normally. But with every subsequent run the
>> progress bar moves maybe 20% of the way then I lose the gui. If I close
>> and restart the application the same thing happens, first one works, then
>> problems). If I run a long calculation (hours) the progress bar never
>> moves past the first tick (maybe 1%).
>>
>> Why does IDL stop updating the gui? Any ideas? While the functions in

>> the dlm are compute intensive, they aren't particularly complicated and
>> only rely on some simple macros in an .h file. Just a *lot* of looping
>> over a moderate amount of data.
>>
>> -Rick
>
>
>

Subject: Re: GUI interface update issues
Posted by Haje Korth on Tue, 21 Jun 2005 18:23:20 GMT
View Forum Message <> Reply to Message

Rick,
Yes trying everything on unix takes time and all you will get out of it is a little more understanding on how IDL is coded internally. I thought the same spook that causes the IDLDE to be unresponsive is the origin of your update problem. (I still want the unix-like command line version in Windows, and damn it, I forgot to put that on the final beta questionaire.)

Haje

"Rick Towler" <rick.towler@nomail.noaa.gov> wrote in message news:d99l4c$ua5$1@news.nems.noaa.gov...
>
>
> Haje Korth wrote:
>> Rick,
>> my question would be whether the same things happens on UNIX OS. On
>> Windows I have found IDL has the terrible habbit of taking over with
>> unpredictable results such as GUIs not reacting at all. If the UNIX
>> version works fine, which I suspect it will, your only hope is some input
>> from the developers.
>
> Hey Haje,
>
> Well, I suppose I could install the IDL VM on a linux box and compile
> everything... Doesn't help me too much though if I find it's o.k. on
> linux. I too have had issues with the windows GUI but usually it is that
> IDLDE becomes unresponsive but my applications are o.k.
>
> Upon further investigation I have determined that the dlm angle is wrong.
> The issue occurs with or without calling the dlm but it only occurs with
> one of the many models in the application. While the calculations are
> basically the same, that model has an additional nested for loop in the
> IDL code.

>
> <sigh>
>
> -Rick
>
>
>
>>
>> "Rick Towler" wrote in message news:d99f2fohk1@news.nems.noaa.gov...
>>
>>> Hello group,
>>>
>>> I have an interesting problem with gui redraw on WinXP which maybe
>>> someone can shed some light on.
>>>
>>> I have an application that uses David's progress bar object. We have
>>> some potentially long calculation times and it is important to give a
>>> little feedback. It worked great until I added the ability to call a new
>>> calculation routine written in C as a dlm. If I run the application the
>>> old way (all functions written in IDL) the progress bar and GUI interface
>>> function normally. But if I run the application using the external
>>> routine eventually the GUI interface stops redrawing, the drop down menu
>>> text disappears, and the progress bar fails to update. The application
>>> runs and when calculations are finished it returns to normal (progress
>>> bar is destroyed and interface works as expected) but there is no
>>> feedback while running. A problem since the new calculations can take
>>> hours and hours and it is nice to see where it is in the process.
>>>
>>> The program is structured like so:
>>>
>>> ;----------
>>> setup stuff
>>>
>>> for loop begin
>>>
>>> progressBar->update
>>>
>>> if (use_dlm) then begin
>>> myDLM_proc, param1, param2, OUT1=out1, OUT2=out2...
>>> endif else begin
>>> IDLbased_proc, param1, param2, OUT1=out1, OUT2=out2...
>>> endelse
>>>
>>> for loop begin
>>>
>>> if (use_dlm) then begin
>>> anotherDLM_proc, param1, param2, OUT1=out1,OUT2=out2...
>>> endif else begin

>>> anotherIDLbased_proc, param1, param2, OUT1=out1,OUT2=out2...
>>> endelse
>>>
>>> endfor
>>>
>>> endfor
>>> ;------------
>>>
>>> Actually a lot more is going on but you get the idea.
>>>
>>> And to elaborate on what I mean by "eventually the gui stops responding".
>>> If I start the application and run a short calculation (~2 minutes) the
>>> first time the gui functions normally. But with every subsequent run the
>>> progress bar moves maybe 20% of the way then I lose the gui. If I close
>>> and restart the application the same thing happens, first one works, then
>>> problems). If I run a long calculation (hours) the progress bar never
>>> moves past the first tick (maybe 1%).
>>>
>>> Why does IDL stop updating the gui? Any ideas? While the functions in
>>> the dlm are compute intensive, they aren't particularly complicated and
>>> only rely on some simple macros in an .h file. Just a *lot* of looping
>>> over a moderate amount of data.
>>>
>>> -Rick
>>
>>

Subject: Re: GUI interface update issues
Posted by Karl Schultz on Tue, 21 Jun 2005 20:04:07 GMT
View Forum Message <> Reply to Message

On Tue, 21 Jun 2005 09:23:39 -0700, Rick Towler wrote:

> Hello group,
>
> I have an interesting problem with gui redraw on WinXP which maybe
> someone can shed some light on.

snip

> Why does IDL stop updating the gui? Any ideas? While the functions in
> the dlm are compute intensive, they aren't particularly complicated and
> only rely on some simple macros in an .h file. Just a *lot* of looping
> over a moderate amount of data.

You might try calling the (documented) IDL_BailOut function. One purpose
for this function is to let users have the opportunity to interrupt IDL

during a long-running calculation. Many of the IDL internal system routines call IDL_BailOut as they make progress through a calculation. It is probably good form to code your C DLM's to call this function every so often so that users can break out of it.

One side effect of IDL_BailOut is to flush window events, which depending on how your app is designed, might help keep the GUI's and progress bars fresh during the long-running operation.

I'm not completely certain this will help you, but it is worth a try. It is very easy to give it a shot. If it helps, then spend a little more time to figure out a good policy for calling BailOut so that you don't call it too often or too infrequently. You don't want to slow down your function TOO much.

Karl

Subject: Re: GUI interface update issues
Posted by Rick Towler on Tue, 21 Jun 2005 22:16:19 GMT
View Forum Message <> Reply to Message

Karl Schultz wrote:
> On Tue, 21 Jun 2005 09:23:39 -0700, Rick Towler wrote:
>
>
>> Hello group,
>>
>> I have an interesting problem with gui redraw on WinXP which maybe
>> someone can shed some light on.
>
>
> snip
>
>
>> Why does IDL stop updating the gui? Any ideas? While the functions in
>> the dlm are compute intensive, they aren't particularly complicated and
>> only rely on some simple macros in an .h file. Just a *lot* of looping
>> over a moderate amount of data.
>
>
> You might try calling the (documented) IDL_BailOut function. One purpose
> for this function is to let users have the opportunity to interrupt IDL
> during a long-running calculation. Many of the IDL internal system
> routines call IDL_BailOut as they make progress through a calculation. It
> is probably good form to code your C DLM's to call this function every so
> often so that users can break out of it.
>

> One side effect of IDL_BailOut is to flush window events, which depending
> on how your app is designed, might help keep the GUI's and progress bars
> fresh during the long-running operation.
>
> I'm not completely certain this will help you, but it is worth a try. It
> is very easy to give it a shot. If it helps, then spend a little more
> time to figure out a good policy for calling BailOut so that you don't
> call it too often or too infrequently. You don't want to slow down your
> function TOO much.

Thanks for the tip.

It didn't help with the GUI updating issue but it does allow the user to
"bail out" if needed. Which raises an interesting question... How
does IDL *really* handle interrupts.

The application structure is pretty basic. There is a main procedure
which sets up the gui (buttons, menus, windows). The buttons trigger
the main event handler which calls a function to display a modal dialog
where params are gathered. Clicking O.K. will return these params to
the main event procedure where they are passed to the appropriate model
procedure where the actual calculations are performed. After the model
procedure runs it returns to the event handler which returns to the main
routine to wait for the next event. There are no timers. The only
events are button clicks.

```
main ->
   event_handler ->
      gather_params modal dialog
      model procedure ->
         some stuff
         for loop
            main calculation here
            lots of IDL code + dlm calls
         endfor
         some more stuff
   event_handler
main
```

Reading the IDL_BailOut docs leads me to believe that IDL faithfully
captures interrupts and will break before the next IDL command is
executed and that the reason for IDL_BailOut is to provide an exit from
external routines.

But it seems that this is overly simplified. If I try to break my (pure
IDL) program's execution once it has entered the model procedure it will
continue to run thru the loop and thru 50 or so more lines of IDL code

after the loop before it stops at a call to SAVE.

So IDL is capturing the interrupt signal, but there seems to be limits on when it will stop execution based on where IDL is in the call stack. I would guess that SAVE has a call to IDL_BailOut to which is why my program breaks on SAVE. If I didn't have that call to SAVE execution would continue back up the call stack to some point where the interrupt would be handled?

What's the logic? Is this an issue of how I have structured my program or is this how IDL always handles interrupts and I haven't noticed until now?

-Rick

Subject: Re: GUI interface update issues
Posted by Haje Korth on Wed, 22 Jun 2005 12:16:14 GMT
View Forum Message <> Reply to Message

Karl,
thanks for introducing me to the IDL_BailOut function, which may be very helpful in my dlm loops. While it may be documented, the IDL documentation consists of several thousand pages now, and it is impossible (for me at least) to remember every entry in it. Rick raised some interesting question that I would be curious to hear about, too. Can you ellaborate on the interrupt issue?

Thanks,
Haje

"Karl Schultz" <k____schultz@rsinc.com> wrote in message news:pan.2005.06.21.20.04.05.703000@rsinc.com...
> On Tue, 21 Jun 2005 09:23:39 -0700, Rick Towler wrote:
>
>> Hello group,
>>
>> I have an interesting problem with gui redraw on WinXP which maybe
>> someone can shed some light on.
>
> snip
>
>> Why does IDL stop updating the gui? Any ideas? While the functions in
>> the dlm are compute intensive, they aren't particularly complicated and
>> only rely on some simple macros in an .h file. Just a *lot* of looping
>> over a moderate amount of data.
>

> You might try calling the (documented) IDL_BailOut function. One purpose
> for this function is to let users have the opportunity to interrupt IDL
> during a long-running calculation. Many of the IDL internal system
> routines call IDL_BailOut as they make progress through a calculation. It
> is probably good form to code your C DLM's to call this function every so
> often so that users can break out of it.
>
> One side effect of IDL_BailOut is to flush window events, which depending
> on how your app is designed, might help keep the GUI's and progress bars
> fresh during the long-running operation.
>
> I'm not completely certain this will help you, but it is worth a try. It
> is very easy to give it a shot. If it helps, then spend a little more
> time to figure out a good policy for calling BailOut so that you don't
> call it too often or too infrequently. You don't want to slow down your
> function TOO much.
>
> Karl
>

Subject: Re: GUI interface update issues
Posted by Rick Towler on Wed, 22 Jun 2005 15:57:42 GMT
View Forum Message <> Reply to Message

Clearly I am learning this as I go and I probably posted a little too
early... I never have had issues with GUI interface drawing and I never
really cared about interrupting program execution. Let the users wait.
My code is pure genius at work! :) lol

> Reading the IDL_BailOut docs leads me to believe that IDL faithfully
> captures interrupts and will break before the next IDL command is
> executed and that the reason for IDL_BailOut is to provide an exit
> from external routines.
>
> But it seems that this is overly simplified. If I try to break
> my (pure IDL) program's execution once it has entered the model
> procedure it will continue to run thru the loop and thru 50 or
> so more lines of IDL code after the loop before it stops at a
> call to SAVE.
>
> So IDL is capturing the interrupt signal, but there seems to be
> limits on when it will stop execution based on where IDL is in
> the call stack. I would guess that SAVE has a call to
> IDL_BailOut to which is why my program breaks on SAVE. If I
> didn't have that call to SAVE execution would continue back up
> the call stack to some point where the interrupt would be handled?
>

> What's the logic? Is this an issue of how I have structured
> my program or is this how IDL always handles interrupts and I
> haven't noticed until now?

I'll add that the GUI gets updated when program execution returns to the main program event handler. For example, I batched 3 model runs where the 3 runs are executed via a loop in the main event handler. GUI drawing is suspended while within the model procedure, but when control returns to the main event handler the GUI is updated right before the program enters another model procedure (where GUI updating is again suspended).

The thing that confounds this issue is that when the application enters a model procedure the GUI is updated for a few seconds, maybe 20 seconds. *Then* the GUI fails to update. The reason why I never noticed this until now is that all of our models ran in less than 20 seconds. Now with models that take 2 minutes to 4 hours the lack of updating is apparent.

So now I am really curious how IDL handles this on other OSes. And it is starting to smell fishy (that's funny. These are acoustic backscattering models of fish. Get it? Fishy? <nudge> <nudge> <wink> <wink>)

<sigh> I need to make a pot of coffee.

-Rick

Subject: Re: GUI interface update issues
Posted by Karl Schultz on Wed, 22 Jun 2005 16:22:13 GMT
View Forum Message <> Reply to Message

On Tue, 21 Jun 2005 15:16:19 -0700, Rick Towler wrote:

>
>
> Karl Schultz wrote:
>> On Tue, 21 Jun 2005 09:23:39 -0700, Rick Towler wrote:
>>
>>
>>> Hello group,
>>>
>>> I have an interesting problem with gui redraw on WinXP which maybe
>>> someone can shed some light on.
>>
>>
>> snip

>>
>>
>>> Why does IDL stop updating the gui? Any ideas? While the functions in
>>> the dlm are compute intensive, they aren't particularly complicated and
>>> only rely on some simple macros in an .h file. Just a *lot* of looping
>>> over a moderate amount of data.
>>
>>
>> You might try calling the (documented) IDL_BailOut function. One purpose
>> for this function is to let users have the opportunity to interrupt IDL
>> during a long-running calculation. Many of the IDL internal system
>> routines call IDL_BailOut as they make progress through a calculation. It
>> is probably good form to code your C DLM's to call this function every so
>> often so that users can break out of it.
>>
>> One side effect of IDL_BailOut is to flush window events, which depending
>> on how your app is designed, might help keep the GUI's and progress bars
>> fresh during the long-running operation.
>>
>> I'm not completely certain this will help you, but it is worth a try. It
>> is very easy to give it a shot. If it helps, then spend a little more
>> time to figure out a good policy for calling BailOut so that you don't
>> call it too often or too infrequently. You don't want to slow down your
>> function TOO much.
>
> Thanks for the tip.
>
> It didn't help with the GUI updating issue but it does allow the user to
>   "bail out" if needed. Which raises an interesting question... How
> does IDL *really* handle interrupts.
>
> The application structure is pretty basic. There is a main procedure
> which sets up the gui (buttons, menus, windows). The buttons trigger
> the main event handler which calls a function to display a modal dialog
> where params are gathered. Clicking O.K. will return these params to
> the main event procedure where they are passed to the appropriate model
> procedure where the actual calculations are performed. After the model
> procedure runs it returns to the event handler which returns to the main
> routine to wait for the next event. There are no timers. The only
> events are button clicks.
>
> main ->
>   event_handler ->
>   gather_params modal dialog
>   model procedure ->
>   some stuff
>   for loop
>   main calculation here

> lots of IDL code + dlm calls
> endfor
> some more stuff
> event_handler
> main
>
>
> Reading the IDL_BailOut docs leads me to believe that IDL faithfully
> captures interrupts and will break before the next IDL command is
> executed and that the reason for IDL_BailOut is to provide an exit from
> external routines.
>
> But it seems that this is overly simplified. If I try to break my (pure
> IDL) program's execution once it has entered the model procedure it will
> continue to run thru the loop and thru 50 or so more lines of IDL code
> after the loop before it stops at a call to SAVE.
>
> So IDL is capturing the interrupt signal, but there seems to be limits
> on when it will stop execution based on where IDL is in the call stack.
> I would guess that SAVE has a call to IDL_BailOut to which is why my
> program breaks on SAVE. If I didn't have that call to SAVE execution
> would continue back up the call stack to some point where the interrupt
> would be handled?
>
> What's the logic? Is this an issue of how I have structured my program
> or is this how IDL always handles interrupts and I haven't noticed until
> now?
>

IDL doesn't check for pending interrupts like this one when running sequences of non-interactive code or when not doing I/O. I think that the rationale is that if you interrupt the execution of a sequence of non-interactive code, you don't really care about the exact place that sequence gets interrupted, mainly because you have no control at all over it. You have no way of reliably keeping any particular statement from executing when you hit the break key. So, there's no checking during this sequence, which may help interpreter efficiency a bit. And that explains why it didn't stop until you tried to do some I/O.

Back to the original topic: I was probably wrong about BailOut processing widget events. One thing you might want to try is a call to WIDGET_EVENT() with no widget id and /NOWAIT. You can add this call to your for loop in the model procedure. It will cause pending events to be processed and is sort of like waking up XMANAGER for a moment explicitly while the application is too busy to return to XMANAGER widget processing.

Karl

Subject: Re: GUI interface update issues
Posted by Rick Towler on Wed, 22 Jun 2005 16:37:48 GMT
View Forum Message <> Reply to Message

> Back to the original topic: I was probably wrong about BailOut processing
> widget events. One thing you might want to try is a call to
> WIDGET_EVENT() with no widget id and /NOWAIT. You can add this call to
> your for loop in the model procedure. It will cause pending events to be
> processed and is sort of like waking up XMANAGER for a moment explicitly
> while the application is too busy to return to XMANAGER widget processing.

Thanks Karl, that did the trick. I guess I should have connected XMANAGER/event processing to this issue but I didn't think that GUI interface updates were handled similarly to IDL application events. I thought it was an IDL internals/OS issue.

Actually, this will help me fix another nagging issue in another application...

-Rick