Subject: IDL Runtime
Posted by wonko on Sat, 31 May 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Hi there!

Is there anyone using IDL runtime licenses? I guess this is a RTFM, we probably have some information about it somehere, but I didn't find any yet. The online help does not mention them, 'idl -h' tells nothing about command line parameters.

What I found out is: IDL started with the command line parameter '-rt' looks for the file 'RUNTIME.SAV', restores it, and executes the procedure MAIN. But there must be other parameters, at least for specifying the filename, idl -rt -file ~/idl/myprogram.sav or something like this.

Thanks for any insights!

Alex
--
Alex Schuster Wonko@weird.cologne.de
alex@pet.mpin-koeln.mpg.de

Subject: Re: IDL RunTime
Posted by davidf on Sun, 06 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Kelly Dean (krdean@lamar.colostate.edu) writes:

> Does a Runtime version of IDL come with the CD distribution?
>
> I am making RecCDs with GOES satellite data and I would like to add an
> IDL sav file with one of my routines to display these images. Incase a
> borrower of these CDs does not have IDL on their machine, I would like
> to allow them to use the "runtime" version of IDL to use my routine to
> display the imagery.

I'm not sure about UNIX installations, but every Windows CD-ROM I've ever installed has installed a run-time version of IDL along with the regular version. But since you *have* to install IDL to have either version available, the question seems moot to me. What difference would it make which version they ran your application in?

Also realize that your IDL save file is going to be version specific. That will easily cut out a large

faction of your potential users unless you provide
save files for *all* expected IDL versions.

Cheers,

David

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

Subject: Re: IDL RunTime
Posted by wbiagiot on Mon, 07 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

In article <MPG.130788bbb3a619e3989a11@news.frii.com>,
  davidf@dfanning.com (David Fanning) wrote:
> Kelly Dean (krdean@lamar.colostate.edu) writes:

>> Incase a borrower of these CDs does not have IDL on their machine, I
>> would like to allow them to use the "runtime" version of IDL to use
>> my routine to display the imagery.

I think someone is confused (maybe me). You *need* to have IDL
installed on your machine (Runtime or Development version) in order to
execute an IDL sav file. It is not a standalone executable. If a
user has "runtime" IDL installed, then they have "IDL on their machine".
Development licenses allow either the development or runtime
configuration, whereas runtime licenses are strictly runtime
configuration. Hope this helps.

--
"They don't think it be like it is, but it do."

Oscar Gamble, NY Yankees

Subject: Re: IDL RunTime
Posted by davidf on Mon, 07 Feb 2000 08:00:00 GMT

View Forum Message <> Reply to Message

Kelly Dean (krdean@lamar.colostate.edu) writes:

> If ENVI, RiverTools, ION, etc... can be issued with "runtime" IDL, I was
> just wondering if any of my tools could be issued with "runtime" IDL.

You can issue any application you like, including your own,
with IDL runtime licenses. (I do this all the time for clients.)
You just have to be willing to pay for each one of them. :-)

Cheers,

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

Subject: Re: IDL RunTime
Posted by Kelly Dean on Mon, 07 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

I could be mistaken.....

If ENVI, RiverTools, ION, etc... can be issued with "runtime" IDL, I was
just wondering if any of my tools could be issued with "runtime" IDL.

I thought I seen this done with an application developed under IDL 4.0 by
one of my colleaques a long time ago.

Kelly

Kelly Dean wrote:

> Does a Runtime version of IDL come with the CD distribution?
>
> I am making RecCDs with GOES satellite data and I would like to add an
> IDL sav file with one of my routines to display these images. Incase a
> borrower of these CDs does not have IDL on their machine, I would like
> to allow them to use the "runtime" version of IDL to use my routine to
> display the imagery.
>
> Kelly Dean
> CSU/CIRA

Subject: Re: IDL Runtime
Posted by David Fanning on Wed, 14 Nov 2001 14:01:30 GMT
View Forum Message <> Reply to Message

Andre Kyme (nak@imag.wsahs.nsw.gov.au) writes:

> I've developed a widget program which must now be run on IDL runtime.
> Problem is that whenever a "Cancel" button is selected (which
> should return execution to the initial menu options) the entire program
> exits and IDL too! Does anyone know why this is? I'm using the RETALL
> command in the program to return to the top level (which is XMANAGING
> the menu) when "cancel" is pressed. It seems to be working fine with
> the normal development license??

I think your main problem is that this program works
at all, anywhere. Using RETALL to negotiate a widget
program is one strange way to write a widget program!

But RETALL will get you all the way back to the
main program level (command line), which clearly
doesn't exist in a run-time environment. I think
I would be thinking about writing IDL programs in
more conventional ways. :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
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: IDL Runtime
Posted by David Fanning on Wed, 14 Nov 2001 23:31:44 GMT
View Forum Message <> Reply to Message

Andre Kyme (nak@imag.wsahs.nsw.gov.au) writes:

> Thanks for the replies re RUNTIME and RETALL.

Oh, oh. I was afraid of this. :-(

> Perhaps my problem is some confusion relating to how XMANAGER works. As
> far as I can understand if I create a bunch of buttons under a TL base,

> then call XMANAGER, the program is 'hung' in a loop checking whether
> events are being processed. So far OK?

Well, not exactly. There isn't any "loop" to get
hung up in, really. I prefer to think that
an officer of the court has been assigned to
to "pay attention", but it's probably a matter
of semantics, and it doesn't really matter which
metaphor you prefer to use.

> If one of the buttons is clicked,
> then we jump to the event handler for that button; but suppose that the
> event handler calls an event-processing routine that calls another, that
> calls another etc. - so we end up a few levels deep in the processing of
> the event. I have used modal prompts along the way in this event
> processing so that the user can decide "Yes" "No" "Maybe" and so on. One
> option I want to give them is "Cancel back to the bunch of buttons."
> Retall seemed a useful way of getting back to main program level, which am
> I right in saying will enter us back into the XMANAGER loop ready for the
> next event??

No, I don't think so. I really think it is a minor miracle
that your program works at all, but I believe it is because
widget programs don't really need to use XMANAGER. (And
I really don't want to get into any of this at all.) I think your
program works, but for the wrong reason, I guess is what I
am trying to say.

> David says, "Try a more conventional way than using retall" -
> what is the conventional way of returning from deep in event-processing
> programs back to the menu buttons?

The "conventional" way of writing widget programs is
to get into and out of event handlers as fast as
possible, not to spend the rest of the afternoon
in there trundling around. But I guess I can see
some sort of hierarchical data structure that might
require something like you describe. Maybe you want
to build a data structure based on the user's answers
to questions, or something like that.

If that is the case, then I think you don't have
to do anything special. It doesn't matter how deep you go
in an event handler, you don't get out of the event
handler until your return from all those levels. I guess
what you are trying to do is short-circuit all those
intervening levels.

I write most of my modal widgets with CANCEL
keywords attached to a Cancel button so I can
check to see if the user wanted out of a dialog
without being penalized:

```
  dataStruct = AskQuestion('Want a foo field?', Cancel=cancelled)
  IF cancelled THEN RETURN ELSE $
    dataStruct = Create_Struct(dataStruct, foo, Fltarr(12))
```

My dialogs are set up so that if the user kills the dialog
with the mouse, it is the same as hitting a cancel button.

Perhaps if these functions were a set of nested functions
something like this would be able to set the cancel flag
all the way back to the original event handler level.

(I don't know. It is hard for me to envision exactly
what you have. And I certainly don't want to write
an example program.)

Maybe you just need to re-think your program design,
although I'm sure you don't want to hear that after
writing what is, I am sure, a complicated program.

Good luck with this.

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
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: IDL Runtime
Posted by James Kuyper on Wed, 14 Nov 2001 23:49:20 GMT
View Forum Message <> Reply to Message

Andre Kyme wrote:
...
> Retall seemed a useful way of getting back to main program level, which am
> I right in saying will enter us back into the XMANAGER loop ready for the
> next event?? ...

I believe it will cause XMANAGER itself to exit.

> ...David says, "Try a more conventional way than using retall" -
> what is the conventional way of returning from deep in event-processing
> programs back to the menu buttons? ...

Use 'return'. If the wrong things happen when you return from the
procedure or function, then there's a defect somewhere in your code;
either in the procedure/function itself, or in one of the things that
directly or indirectly calls it.

> ... There are no widgets to destroy, I just
> want to get back to the state the application was in when first loaded.

That's the problem. RETALL returns to the state before the application
was loaded. i.e.: no application.

--
James Kuyper
MODIS Level 1 Lead
Science Data Support Team
(301) 352-2150

Subject: Re: IDL Runtime
Posted by Andre Kyme on Thu, 15 Nov 2001 00:22:24 GMT
View Forum Message <> Reply to Message

Hi David,

Appreciate the reply, but as yet nobody has given me a good explanation of how
XMANAGER works and I don't find the documentation provided by RSI insightful at
all. I'm quite happy to adjust my program design - but I'm not sure exactly how
to do this. People keep replying that "Your program must be so complicated..."
but it is really quite simple! Or at least the IDEA is! I expect the idea would
be a very typical widget programming endeavour!

To briefly describe what my design structure is:

About 5 widget buttons get created and realized to form a menu
Each has an event handler
I call XMANAGER to manage the created widgets
I wait...

Oh, button number 1 was pressed!
Well we better do what event handler number 1 tells us to do.

The event-handler is just a couple of programs that follow one after the other

- is that complicated!! Some of them call other programs - But execution is just moving sequentially through the event handler routine. Pretty standard I think.
The actual programs in the event handler involve the user drawing a region of interest (I don't use draw widgets, just a standard graphics window and a routine involving the cursor command) and involve moving it around, and also confirming they are happy with its location.
Now suppose they get fed up with the whole business (like I'm feeling right now) and want to go and have a coffee - so when they are asked to confirm the region (I use dialogue_message with the /cancel keyword), they click "CANCEL" - Why should it be so complicated to just return to your menu options when I detect the cancellation? This would have to be a pretty standard widget application surely - menu, option, back to menu, option, back to menu...? What is the conventional philosophy for doing 'this'?

So again, the design involves having options initially (menu). Some options aren't relevant initially such as "Save" and "Print", so if the user clicks on these I issue messages for the user telling them there's no point clicking these ones yet. The most likely thing they will want to do initially is "Analysis". "Analysis" is the button with the event handler I described above: programs that just simply get executed one after the other. During execution of the event-handler programs, the menu buttons can't be clicked, mainly due to the fact that the region drawing/moving uses the CURSOR command - so the computer is waiting for mouse clicks inside the graphics window. (You might prefer draw widgets w/o CURSOR etc etc, but there's really no problem I can see with doing it the way I have. It essentially creates MODAL functionality which ensures the user gets to the end. Anyway I think this is beside the point). But after this part is done and the results have been displayed, that's the end of "Analysis". Now they might want to print the results... or do another analysis...etc.

Also, exactly what DOES happen when instead, the user doesn't get fed up, finishes the whole event handling routine, and we arrive at the "END" command at the conclusion of the event-handler? Where is program execution? What is happening? I'm thinking that we're ready for another menu-button press??

Would appreciate any guidance on a structure,
Thanks,
Andre Kyme

Subject: Re: IDL Runtime
Posted by Kristine Hensel on Thu, 15 Nov 2001 01:20:01 GMT
View Forum Message <> Reply to Message

Andre Kyme wrote:
> The event-handler is just a couple of programs that follow one after the other
> - is that complicated!! Some of them call other programs - But execution is

> just moving sequentially through the event handler routine. Pretty standard I
> think.
> The actual programs in the event handler involve the user drawing a region of
> interest (I don't use draw widgets, just a standard graphics window and a
> routine involving the cursor command) and involve moving it around, and also
> confirming they are happy with its location.
> Now suppose they get fed up with the whole business (like I'm feeling right
> now) and want to go and have a coffee - so when they are asked to confirm the
> region (I use dialogue_message with the /cancel keyword), they click "CANCEL" -
> Why should it be so complicated to just return to your menu options when I
> detect the cancellation? This would have to be a pretty standard widget
> application surely - menu, option, back to menu, option, back to menu...? What
> is the conventional philosophy for doing 'this'?

Are you saying that you have *one* event handler that gets the user to
draw a ROI, move it around, confirm that they're happy with it? If so,
I would consider breaking these up into separate events. If you put all
the code that handles button events in one routine, it could look
something like this (I've assumed that you use a draw widget, even
though you don't):

```
 case which_button of
  'draw': begin
    ; make the draw widget start returning events:
      widget_control, info.draw_widget, /motion_events,$
   /button_events
    ; (an event handler for the draw widget could
    ; store how the region of interest has been      ; defined in the
info structure)
          widget_control, info.confirm_button, sensitive=1
      widget_control, info.cancel_button, sensitive=1
  end ; draw
  'confirm': begin
    ; don't pay attention to any more cursor
    ; events:
      widget_control, info.draw_widget, motion_events=0, $
   button_events=0
      widget_control, info.confirm_button, sensitive=0
      widget_control, info.cancel_button, sensitive=0
    ; pass the region of interest, stored in the
    ; info structure, to your analysis routine:
      start_analysis, info
    ; make the Save and Print buttons available,
    ; now that some analysis has been done:
      widget_control, info.save_button, sensitive=1
      widget_control, info.print_button, sensitive=1
  end ; confirm
  'cancel': begin
```

```
    ; don't pay attention to any more cursor
    ; events:
      widget_control, info.draw_widget, motion_events=0, $
   button_events=0
      widget_control, info.confirm_button, sensitive=0
      widget_control, info.cancel_button, sensitive=0
  end ;  cancel
```

> So again, the design involves having options initially (menu). Some options
> aren't relevant initially such as "Save" and "Print", so if the user clicks on
> these I issue messages for the user telling them there's no point clicking
> these ones yet.

A more straightforward way is to just make sure the user can't press
irrelevant buttons in the first place, by making them insensitive when
they can't do anything useful.

> The most likely thing they will want to do initially is
> "Analysis". "Analysis" is the button with the event handler I described above:
> programs that just simply get executed one after the other. During execution of
> the event-handler programs, the menu buttons can't be clicked, mainly due to
> the fact that the region drawing/moving uses the CURSOR command - so the
> computer is waiting for mouse clicks inside the graphics window.

The way I've written it above, the draw widget keeps on accepting events
until the user presses the confirm or cancel button.

If you're using CURSOR (which I've never used, so forgive me), I presume
you use a certain mouse click (e.g. right mouse button) to signify that
the user has finished defining the region? The event-handling code that
calls the CURSOR function should just store the ROI once the user has
said that they're done defining. XMANAGER can then wait until the
confirm or cancel button is pressed.

This is all to do than to explain, of course, so I apologize if I've
confused your further. I suggest you buy David's book. :)

Kristine
--
Kristine Hensel e-mail: kristine@esands.com
Environmental Systems & Services phone: +61-(0)3-9835-7901
20 Council St., Level 3 fax: +61-(0)3-9835-7900
Hawthorn East, VIC, Australia 3124

Subject: Re: IDL Runtime
Posted by David Fanning on Thu, 15 Nov 2001 01:21:32 GMT

Andre Kyme (nak@imag.wsahs.nsw.gov.au) writes:

> Appreciate the reply, but as yet nobody has given me a good explanation of how
> XMANAGER works and I don't find the documentation provided by RSI insightful at
> all.

Yes, well, no one wants to commit themselves
because no one is really sure *how* it works.
The documentation was written in 1946. I think
it's safe to say, however, that most of us are
pretty sure it doesn't work the way you seem
to think it works. :-)

> I'm quite happy to adjust my program design - but I'm not sure exactly how
> to do this. People keep replying that "Your program must be so complicated..."
> but it is really quite simple! Or at least the IDEA is! I expect the idea would
> be a very typical widget programming endeavour!

It's the experience of this newsgroup (or, at the
very least, this correspondent) that the hardest
questions come from the simplest things.

> To briefly describe what my design structure is:
>
> About 5 widget buttons get created and realized to form a menu
> Each has an event handler
> I call XMANAGER to manage the created widgets
> I wait...
>
> Oh, button number 1 was pressed!
> Well we better do what event handler number 1 tells us to do.
>
> The event-handler is just a couple of programs that follow one after the other
> - is that complicated!! Some of them call other programs - But execution is
> just moving sequentially through the event handler routine. Pretty standard I
> think.

Then, no problem. When you get to the end of all the
sequential programs, you exit the event handler and
you are ready to process the next event. (Which has
already queued up, no doubt, because users are
murder on slow event handlers. But that's another
story.)

> The actual programs in the event handler involve the user drawing a region of
> interest (I don't use draw widgets, just a standard graphics window and a
> routine involving the cursor command) and involve moving it around, and also

> confirming they are happy with its location.

Well, IMHO, this is a huge mistake. And this
part is going to have to be re-written if you
want to run this as a run-time program, I think.
Run-time IDL will not work with "hybrid" programs
like this, it seems to me, although I haven't tried
it.

> Now suppose they get fed up with the whole business (like I'm feeling right
> now) and want to go and have a coffee - so when they are asked to confirm the
> region (I use dialogue_message with the /cancel keyword), they click "CANCEL" -

Here is the problem. How do you know when they
are finished? What they are doing is not in your
"widget space", so you can't get any feedback from
them. What should be happening is that the user is
generating events that you are handling in your widget
program. Here they are working outside the box, and you
can't get any feedback from them.

> Why should it be so complicated to just return to your menu options when I
> detect the cancellation? This would have to be a pretty standard widget
> application surely - menu, option, back to menu, option, back to menu...? What
> is the conventional philosophy for doing 'this'?

RETURN, simple as that. The code would probably look
like this:

```
ans = Dialog_Message('Are you finished?', /Question, /Cancel)
IF Strupcase(ans) EQ 'CANCEL' THEN RETURN
```

> So again, the design involves having options initially (menu). Some options
> aren't relevant initially such as "Save" and "Print", so if the user clicks on
> these I issue messages for the user telling them there's no point clicking
> these ones yet. The most likely thing they will want to do initially is
> "Analysis". "Analysis" is the button with the event handler I described above:
> programs that just simply get executed one after the other. During execution of
> the event-handler programs, the menu buttons can't be clicked, mainly due to
> the fact that the region drawing/moving uses the CURSOR command - so the
> computer is waiting for mouse clicks inside the graphics window. (You might
> prefer draw widgets w/o CURSOR etc etc, but there's really no problem I can see
> with doing it the way I have.

Well, I can see problems upon problems. But I'll
let it go here.

> It essentially creates MODAL functionality which

> ensures the user gets to the end. Anyway I think this is beside the point).

I'm not sure it's beside the point. I'm more
inclined to think it is *exactly* the point.
It is certainly what makes your program NOT a
widget program. It is hard to make the point
that widgets don't *work* right, when what you
use as an example is a program that is not
written as a widget program. What you have,
I think, is a program that uses some widget
functionality, and not always in the appropriate
way, even then.

> Also, exactly what DOES happen when instead, the user doesn't get fed up,
> finishes the whole event handling routine, and we arrive at the "END" command
> at the conclusion of the event-handler? Where is program execution? What is
> happening? I'm thinking that we're ready for another menu-button press??

You are indeed ready for another button press.
It is the "where am I part" that confuses me.
I guess it depends on whether you used the
NO_BLOCK keyword when you called XManager.

I'd guess from what you say happens that you
are probably at the stop where the interpreter
is looking for something to happen. That is
probably the main-level, or something very
much like it.

I'd be curious to know if your program still
works *without* the XManager call.

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
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: IDL Runtime
Posted by Andre Kyme on Thu, 15 Nov 2001 01:58:13 GMT

Hi David,

I wrote:

> The actual programs in the event handler involve the user drawing a region of
> interest (I don't use draw widgets, just a standard graphics window and a
> routine involving the cursor command) and involve moving it around, and also
> confirming they are happy with its location.

You wrote:

Well, IMHO, this is a huge mistake. And this
part is going to have to be re-written if you
want to run this as a run-time program, I think.
Run-time IDL will not work with "hybrid" programs
like this, it seems to me, although I haven't tried
it.

What do you mean a "hybrid program" ?

You write:

Here is the problem. How do you know when they
are finished? What they are doing is not in your
"widget space", so you can't get any feedback from
them. What should be happening is that the user is
generating events that you are handling in your widget
program. Here they are working outside the box, and you
can't get any feedback from them.

I reply:

The user clicks the "Analysis" button
They have a group of images displayed before them
They are asked to choose which one they want to draw an ROI on (dialogue_message)
As soon as the dialogue is OK'd the cursor command is waiting
Cursor waits until something in the graphics window is selected - clicks outside the window have no effect.
They are asked for confirmation of the image (dialogue_message)
They are then prompted to draw an ROI (dialogue_message)
Cursor is waiting for the first point
A right mouse click finishes the region
They are presented with the next image and asked to move the ROI (that they just drew) over the new image (dialogue_message)
Cursor is waiting for the first position to move it to
And so on for the remaining images
Eventually they are all done, but in the meantime the user has only had limited

responses and I think I've been prepared for all of them??
Event-handler ends
Ready for next event from button menu

I can see that draw widgets and motion events might be 'nicer' but is it necessary.
I'm just using cursor as part of my event-handling.

You must be getting tired of this nutter on the other end of your replies David!
Thankyou,
Andre

Subject: Re: IDL Runtime
Posted by Andre Kyme on Thu, 15 Nov 2001 02:45:46 GMT
View Forum Message <> Reply to Message

How come cursor doesn't work with runtime?

Andre

Subject: Re: IDL Runtime
Posted by David Fanning on Thu, 15 Nov 2001 02:49:31 GMT
View Forum Message <> Reply to Message

Andre Kyme (nak@imag.wsahs.nsw.gov.au) writes:

> What do you mean a "hybrid program" ?

I mean a program that is half-widget and half
something else. You haven't indicated whether
your program runs without the XManager call,
but I suspect you are in the nether world
between a widget event loop (whatever the hell
that means) and the interpreter.

(And where are all you guys (Pavel!) who are
suppose to know something about this? I'm
on my tippy toes here and I still feel like
I'm drowning.)

I think this ... whatever ... is what is causing
your problem. I can assure you that if this
program was a proper widget program (without
any RETALLs, for sure) it would run as a run-time
program.

> The user clicks the "Analysis" button
> They have a group of images displayed before them
> They are asked to choose which one they want to draw an ROI on (dialogue_message)
> As soon as the dialogue is OK'd the cursor command is waiting

How does dialog_message return to you the
window they want to draw in?

> Cursor waits until something in the graphics window is selected - clicks outside the
> window have no effect.
> They are asked for confirmation of the image (dialogue_message)
> They are then prompted to draw an ROI (dialogue_message)
> Cursor is waiting for the first point
> A right mouse click finishes the region
> They are presented with the next image and asked to move the ROI (that they just
> drew) over the new image (dialogue_message)
> Cursor is waiting for the first position to move it to
> And so on for the remaining images
> Eventually they are all done, but in the meantime the user has only had limited
> responses and I think I've been prepared for all of them??
> Event-handler ends
> Ready for next event from button menu

If this works, what is the problem?

> I can see that draw widgets and motion events might be 'nicer' but is it necessary.
> I'm just using cursor as part of my event-handling.

Cursor has nothing whatsoever to do with event handling,
at least not in any widget sense. I think, in fact, that
this must be handled by the command interpreter, which is
not available in run-time programs.

To know for sure, I need to see your program (which I
really don't want to do), or I need to build an example
program (which I really, really don't want to do). :-)

> You must be getting tired of this nutter on the other end of your replies David!

Are you talking about Logan? Oh ... you!

No, not at all. The alternative is puzzling over a program
I'm trying to write that I think I already wrote as an example
program. The problem is, I can't seem to figure out how it works,
and it *is* a widget program! :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
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: IDL Runtime
Posted by Andre Kyme on Thu, 15 Nov 2001 02:53:22 GMT
View Forum Message <> Reply to Message

Fancy that! You're right about cursor not working on runtime! Here I was thinking I was talking to some maniac bent on rubbisging a perfectly good routine. :)

Just kidding.
Well, it looks like it's back to the drawing board.

Thankyou to all those who have made some contribution to this very depressing of conclusions.

Andre

Subject: Re: IDL Runtime
Posted by David Fanning on Thu, 15 Nov 2001 03:03:13 GMT
View Forum Message <> Reply to Message

Andre Kyme (nak@imag.wsahs.nsw.gov.au) writes:

> Fancy that! You're right about cursor not working on runtime! Here I was thinking I was
> talking to some maniac bent on rubbisging a perfectly good routine. :)

Whew! And I really was thinking I was just
getting in deeper and deeper. :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: IDL Runtime
Posted by Logan Lindquist on Thu, 15 Nov 2001 13:27:12 GMT
View Forum Message <> Reply to Message

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.165cdb22e9d9fcf4989778@news.frii.com...
> Andre Kyme (nak@imag.wsahs.nsw.gov.au) writes:

>> You must be getting tired of this nutter on the other end of your
replies David!
>
> Are you talking about Logan? Oh ... you!

I am not nuts and I am not tired of David's replies. Hope this clears things up for everyone.

-L

Subject: Re: IDL Runtime
Posted by R.Bauer on Thu, 06 Dec 2001 17:49:29 GMT
View Forum Message <> Reply to Message

Dear Andre, dear David

I am sorry that's I have not spent time last three
weeks to read more detailed this issue

Attached is a small example from our IDL widget lessons which shows
a very nice functionality about starting an event while an event
is already running.

You have not to wait until UP is finished, you can directly press DOWN
or
QUIT.

The trick for this is to call always as event_pro set by the
widget definition "wid1_event".

In the cases of 'wid1_UP' and 'wid1_DOWN' always a second own event
manager
"wid1_own_event" is started which did nothing else as asking for events
from the buttons.

If an event by a button is returned the event procedure "wid1_event"
is runnind independent from the XMANAGER.

This means the "wid1_event" runs many times started like child-processes
and only
if you have used pointers you know the variables in each process and
they are
the same.
If you press "Quit" all of this child-processes of "wid1_event" are
stopped at
once.

I have many of my widgets programmed in this way, but it is hard to
reprogramm an existing one. Sometimes it is enough to give only
a few buttons these functionality as an addition.

If I should explain something in more detail please give me a note.

David if you like you can add this widget to your tips page.

regards

Reimar

>
> So again, the design involves having options initially (menu). Some options
> aren't relevant initially such as "Save" and "Print", so if the user clicks on
> these I issue messages for the user telling them there's no point clicking
> these ones yet. The most likely thing they will want to do initially is
> "Analysis". "Analysis" is the button with the event handler I described above:
> programs that just simply get executed one after the other. During execution of
> the event-handler programs, the menu buttons can't be clicked, mainly due to
> the fact that the region drawing/moving uses the CURSOR command - so the
> computer is waiting for mouse clicks inside the graphics window. (You might
> prefer draw widgets w/o CURSOR etc etc, but there's really no problem I can see
> with doing it the way I have. It essentially creates MODAL functionality which
> ensures the user gets to the end. Anyway I think this is beside the point). But
> after this part is done and the results have been displayed, that's the end of
> "Analysis". Now they might want to print the results... or do another
> analysis...etc.
>
> Also, exactly what DOES happen when instead, the user doesn't get fed up,
> finishes the whole event handling routine, and we arrive at the "END" command
> at the conclusion of the event-handler? Where is program execution? What is

> happening? I'm thinking that we're ready for another menu-button press??
>
> Would appreciate any guidance on a structure,
> Thanks,
> Andre Kyme

--
Reimar Bauer

Institut fuer Stratosphaerische Chemie (ICG-1)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de
http://www.fz-juelich.de/icg/icg1/
 ========== ======
a IDL library at ForschungsZentrum Juelich
 http://www.fz-juelich.de/icg/icg1/idl_icglib/idl_lib_intro.h tml

http://www.fz-juelich.de/zb/text/publikation/juel3786.html
 ========== ======

read something about linux / windows
http://www.suse.de/de/news/hotnews/MS.html
PRO wid1_own_event,event

  IF WIDGET_INFO(event.top,/active) THEN $
   a=WIDGET_EVENT(WIDGET_INFO(event.top,find_by_uname='wid1_DOW
N'),/nowait,bad_id=bad)
  IF WIDGET_INFO(event.top,/active) THEN $
   a=WIDGET_EVENT(WIDGET_INFO(event.top,find_by_uname='wid1_UP'
),/nowait,bad_id=bad)
  IF WIDGET_INFO(event.top,/active) THEN $
   a=WIDGET_EVENT(WIDGET_INFO(event.top,find_by_uname='wid1_QUI
T'),/nowait,bad_id=bad)

END

PRO wid1_event, Event
  WIDGET_CONTROL,event.top,get_uvalue=values
  CASE event.id OF
    WIDGET_INFO(event.top,find_by_uname='wid1_QUIT'): BEGIN
      (*values).is_up =0
      (*values).is_down =0
      WIDGET_CONTROL, event.top,/destroy
    END
    WIDGET_INFO(event.top,find_by_uname='wid1_UP'): BEGIN
      (*values).is_up =1
      (*values).is_down =0

```
        IF event.select EQ 1 THEN BEGIN
           WIDGET_CONTROL,WIDGET_INFO(event.top,find_by_uname='wid1_DRA
W'),get_uvalue=WINDOW
          WSET,WINDOW
          WHILE (*values).counter LE (*values).max AND $
          (*values).is_up EQ 1 DO BEGIN
          ERASE
          XYOUTS,0.5,0.5,/NORM,'UP',alignment=0.5, $
                charsize=(*values).counter/100.
          (*values).counter=(*values).counter+1
          wid1_own_event,event
        ENDWHILE
      ENDIF
    END

    WIDGET_INFO(event.top,find_by_uname='wid1_DOWN'): BEGIN
      (*values).is_up =0
      (*values).is_down =1
      IF event.select EQ 1 THEN BEGIN
         WIDGET_CONTROL,WIDGET_INFO(event.top,find_by_uname='wid1_DRA
W'),get_uvalue=WINDOW
        WSET,WINDOW
        WHILE (*values).counter GT (*values).min AND $
        (*values).is_down EQ 1 DO BEGIN
        ERASE
        XYOUTS,0.5,0.5,/NORM,'DOWN',alignment=0.5, $
              charsize=(*values).counter/100.
        (*values).counter=(*values).counter-1
        wid1_own_event,event
      ENDWHILE
    ENDIF
END
ELSE:
ENDCASE
END

PRO wid1, GROUP_LEADER=wGroup, _EXTRA=_VWBExtra_
    ID_BASE_0=WIDGET_BASE( GROUP_LEADER=wGroup,$
           SCR_XSIZE=300 ,SCR_YSIZE=200,TITLE='IDL',$
           SPACE=3 ,XPAD=3,YPAD=3)


    ID_DRAW_0=WIDGET_DRAW(id_base_0,$
           XOFFSET=100 ,YOFFSET=7, SCR_XSIZE=185 ,$
           SCR_YSIZE=160,uname='wid1_DRAW')


    ID_UP=WIDGET_BUTTON(id_base_0, $
```

```
      XOFFSET=44 ,YOFFSET=60,XSIZE=50,/ALIGN_CENTER ,$
      VALUE='UP',event_pro='wid1_event',uname='wid1_UP')


  ID_DOWN=WIDGET_BUTTON(id_base_0, $
        XOFFSET=44, YOFFSET=83,  XSIZE=50, $
        /ALIGN_CENTER, VALUE='DOWN',$
        event_pro='wid1_event',uname='wid1_DOWN')


  ID_QUIT=WIDGET_BUTTON(id_base_0, $
        /ALIGN_CENTER ,VALUE='QUIT', $
        event_pro='wid1_event',uname='wid1_QUIT')

  WIDGET_CONTROL, /REALIZE, id_BASE_0

  WIDGET_CONTROL,id_draw_0,get_value=window_no
  WIDGET_CONTROL,id_draw_0,set_uvalue=window_no
  WSET,window_no
  ERASE

   values=PTR_NEW(CREATE_STRUCT('counter',0L,'min',1L,'max',100 00L,'is_up',0,$
       'is_down',0))


  WIDGET_CONTROL, /REALIZE, id_BASE_0,set_uvalue=values

  XMANAGER, 'wid1', id_BASE_0
  PTR_FREE,values
END
```

File Attachments

1) wid1.pro, downloaded 158 times

Subject: Re: IDL runtime
Posted by Jon Sattelberger on Wed, 20 Oct 2010 14:14:47 GMT
View Forum Message <> Reply to Message

Wox,

Thank you for your reply. Hopefully I answer all of your questions.

> What keywords are you using in calling LMGR?

I started with DEMO, then the rest below:

```
 IF LMGR(/DEMO) GT 0 THEN PRINT, "This runtime is in demo mode.
```

```
Please contact xyz."
  IF LMGR(/CLIENTSERVER) GT 0 THEN PRINT, "Obtained client server."
  IF LMGR(/EMBEDDED) GT 0 THEN PRINT, "This runtime is in EMBEDDED
mode. Please contact xyz."
  IF LMGR(/RUNTIME) GT 0 THEN PRINT, "This runtime is in RUNTIME
mode."
  IF LMGR(/STUDENT) GT 0 THEN PRINT, "This runtime is in STUDENT mode.
Please contact xyz."
  IF LMGR(/VM) GT 0 THEN PRINT, "This runtime is in VM mode. Please
contact xyz."

  R = LMGR(LMHOSTID = HOSTID)
  PRINT, "Host ID: " + strtrim(HOSTID,2)
  R = LMGR(INSTALL_NUM=INSTALL)
  PRINT, "Install number: " + strtrim(INSTALL,2)
  R = LMGR(SITE_NOTICE=SITE)
  PRINT, "Site notice: " + strtrim(SITE,2)
```

>
> From the help:
> "The fact that the IDL session is running in runtime mode does not
> imply that a runtime license is in use."
>
> How did you check whether a license was found when running in runtime
> mode?

When I would execute the runtime, I would output the information from the code above first. It would then go about, and start it's image processing. Then I compared the runtime license number and site notice (for what it's worth) to the actual runtime license contract on file with ITT. I then verified it on the license server using, lmstat, that it is indeed using a runtime license. Everything seemed valid. I suppose it's just one of those oddities.

Thanks again.

Subject: Re: IDL runtime
Posted by Gray on Wed, 20 Oct 2010 20:19:32 GMT
View Forum Message <> Reply to Message

On Oct 20, 10:14 am, Jon Sattelberger <jon.sattelber...@gmail.com>
wrote:
> Wox,
>
> Thank you for your reply. Hopefully I answer all of your questions.
>
>> What keywords are you using in calling LMGR?

>
> I started with DEMO, then the rest below:
>
> IF LMGR(/DEMO) GT 0 THEN PRINT, "This runtime is in demo mode.
> Please contact xyz."
> IF LMGR(/CLIENTSERVER) GT 0 THEN PRINT, "Obtained client server."
> IF LMGR(/EMBEDDED) GT 0 THEN PRINT, "This runtime is in EMBEDDED
> mode. Please contact xyz."
> IF LMGR(/RUNTIME) GT 0 THEN PRINT, "This runtime is in RUNTIME
> mode."
> IF LMGR(/STUDENT) GT 0 THEN PRINT, "This runtime is in STUDENT mode.
> Please contact xyz."
> IF LMGR(/VM) GT 0 THEN PRINT, "This runtime is in VM mode. Please
> contact xyz."
>
> R = LMGR(LMHOSTID = HOSTID)
> PRINT, "Host ID: " + strtrim(HOSTID,2)
> R = LMGR(INSTALL_NUM=INSTALL)
> PRINT, "Install number: " + strtrim(INSTALL,2)
> R = LMGR(SITE_NOTICE=SITE)
> PRINT, "Site notice: " + strtrim(SITE,2)
>
>
>
>> From the help:
>> "The fact that the IDL session is running in runtime mode does not
>> imply that a runtime license is in use."
>
>> How did you check whether a license was found when running in runtime
>> mode?
>
> When I would execute the runtime, I would output the information from
> the code above first. It would then go about, and start it's image
> processing. Then I compared the runtime license number and site notice
> (for what it's worth) to the actual runtime license contract on file
> with ITT. I then verified it on the license server using, lmstat, that
> it is indeed using a runtime license. Everything seemed valid. I
> suppose it's just one of those oddities.
>
> Thanks again.

Hi Jon!!

--Gray

Subject: Re: IDL runtime

Posted by Wout De Nolf on Fri, 22 Oct 2010 09:22:05 GMT
View Forum Message <> Reply to Message

On Wed, 20 Oct 2010 07:14:47 -0700 (PDT), Jon Sattelberger
<jon.sattelberger@gmail.com> wrote:

> I suppose it's just one of those oddities.

Hmmm, not sure what's happening there. After all, you don't need a
license to run a SAV file under IDLVM.