Subject: Re: passing parameters from base to base
Posted by Pavel A. Romashkin on Thu, 21 Nov 2002 20:52:46 GMT
View Forum Message <> Reply to Message

The simpliest way is to pass a few parameters to SetParams that reside
in the Mainbase STATE structure. I personally, for simplicity sake, pass
the entire STATE structure (which gets passed by reference and is easy
to update in that case).
Of course, it is cleaner if you only pass fields of STATE (that are
pointers because otherwise you'd be passing state.fields by value but if
you alter *state.fields, these will get updated).
If you want to update, say, labels on Mainbase, you can simply call
WIDGET_CONTROL on the State.MainBaseID inside SetParams.
Or, if you want to really do something intricate with the results
returned by SetParams, you could do something like depicted on
http://www.ainaco.com/idl/idl_library/export_to_main.pro
which will interrupt the main event handler until dismissed.
Cheers,
Pavel

Gert wrote:
>
> Hi,
>
> I've been trying to figure this one out for a while. I have 2 bases. If in
> Mainbase the button Set is pushed, a second base SetParams is called. Stuff
> happens there and the idea is that if SetParams is killed, a series of
> numbers go back to Mainbase. Now how can you write this neatly, so that the
> code for the SetParams can easily be used in other progs?
> These are my thoughts:
> I could pass a pointer to SetParams that keeps the desired data, but how
> does Mainbase knows that
> SetParams is killed and that it needs to update its fields?
> I looked at the examples for compound widgets (e.g. cw_defroi) and these do
> the trick but they do not use xmanager. Is this the only way.
>
> thx for any help,
>
> Gert

Subject: Re: passing parameters from base to base
Posted by David Fanning on Thu, 21 Nov 2002 23:46:23 GMT
View Forum Message <> Reply to Message

Gert (Gert.Van.de.Wouwer@NOS_PAMpandora.be) writes:

> I've been trying to figure this one out for a while. I have 2 bases. If in

> Mainbase the button Set is pushed, a second base SetParams is called. Stuff
> happens there and the idea is that if SetParams is killed, a series of
> numbers go back to Mainbase. Now how can you write this neatly, so that the
> code for the SetParams can easily be used in other progs?
> These are my thoughts:
> I could pass a pointer to SetParams that keeps the desired data, but how
> does Mainbase knows that
> SetParams is killed and that it needs to update its fields?

If these are widget programs we are talking about, the simplest
way of communicating between them is to send events back and forth.
Typically, when program 1 calls program 2 it "registers" that it would
like to receive updates if "something important" goes on. In my XColors
program, for example, any program that wants to know about XColors
loading a color table will register with either NOTIFYPRO, NOTIFYOBJ,
or NOTIFYID keywords. Then, depending upon whether it is a procedure,
object, or widget that you want to receive the notification, XCOLORS
will use Call_Procedure, Call_Method, or SEND_EVENT (via Widget_Control)
to notify program 1.

http://www.dfanning.com/programs/xcolors.pro

If you want to know if program 2 is killed, you can do the notification
in its CLEANUP routine.

> I looked at the examples for compound widgets (e.g. cw_defroi) and these do
> the trick but they do not use xmanager. Is this the only way.

Oh, goodness me. Don't be learning your widget programming from
examples that come with IDL! Life is complicated enough already.
Get a good book instead. :-)

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: passing parameters from base to base
Posted by Gert Van de Wouwer on Fri, 22 Nov 2002 13:58:51 GMT
View Forum Message <> Reply to Message

"David Fanning" <david@dfanning.com> wrote in message

news:MPG.18471e3f72f19a6d989a3f@news.frii.com...
>
> Oh, goodness me. Don't be learning your widget programming from
> examples that come with IDL! Life is complicated enough already.
> Get a good book instead. :-)
>
No! Off course you can't expect a decent manual with a 4000 EUR software
package, now do you...

thanks for the replies Pavel and David

Subject: Re: passing parameters from base to base
Posted by R.Bauer on Fri, 22 Nov 2002 15:37:31 GMT

View Forum Message <> Reply to Message

Gert wrote:
> Hi,
>
> I've been trying to figure this one out for a while. I have 2 bases. If in
> Mainbase the button Set is pushed, a second base SetParams is called. Stuff
> happens there and the idea is that if SetParams is killed, a series of
> numbers go back to Mainbase. Now how can you write this neatly, so that the
> code for the SetParams can easily be used in other progs?
> These are my thoughts:
> I could pass a pointer to SetParams that keeps the desired data, but how
> does Mainbase knows that
> SetParams is killed and that it needs to update its fields?
> I looked at the examples for compound widgets (e.g. cw_defroi) and these do
> the trick but they do not use xmanager. Is this the only way.
>
> thx for any help,
>
> Gert
>
>
>
>

Dear Gert

this is a small example how to exchange data.
The trick is to give all the widget functions a UNAME then you can get
access to each of them by widget_info at every program state.
You have only to know the top level base id.
If you use uname you can save and restore data from uvalue of each
element. This is pretty fine.

If you like to have more examples you can have a look at wid1 to wid5
in the widget explaination at

http://www.fz-juelich.de/vislab/software/idl_samples/IDL-Bei spielsammlung.html

These are the examples from our idl lessons at FZ Juelich.

```
PRO small_widget_event,event

   name=WIDGET_INFO(event.id,/UNAME)
   PRINT,name

   WIDGET_CONTROL,event.top,get_uvalue=data


   CASE name OF
     'QUIT': WIDGET_CONTROL,event.top,/destroy
     'START': BEGIN
       WIDGET_CONTROL,get_value=wid,$
        WIDGET_INFO(event.top,find_by_uname='DRAW')
       WSET,wid
       TVSCL,*data
     END
     'SLIDER' :BEGIN
       WIDGET_CONTROL,event.id,get_value=value
       TVSCL,*data<value
     END
     ELSE:
   ENDCASE

END

PRO small_widget
   data=PTR_NEW(DIST(256))
   base=WIDGET_BASE(col=2,title='variables between widgets',$
      uvalue=data)
   id=WIDGET_BUTTON(base,value='START',uname='START')
   id=WIDGET_BUTTON(base,value='QUIT',uname='QUIT')

   base2=WIDGET_BASE(base,row=1, $
       title='variables between widgets')
   id=WIDGET_DRAW(base2,retain=2,xsize=256,ysize=256,$
       uname='DRAW')
   id=WIDGET_SLIDER(base2,/vertical,min=1,$
       max=255,value=255,uname='SLIDER')

   WIDGET_CONTROL,/realize,base
```

```
    XMANAGER,'small_widget',base
    PTR_FREE,data

END
```

--
Reimar Bauer

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

Subject: Re: passing parameters from base to base
Posted by Pavel A. Romashkin on Fri, 22 Nov 2002 17:33:08 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
>
> If these are widget programs we are talking about, the simplest
> way of communicating between them is to send events back and forth.

You are not getting carried away into the world of object widgets, are
you? Seems to me that this would be a sure way to scare off for good
anyone who doesn't write IDL code for living :-)
Cheers,
Pavel

Subject: Re: passing parameters from base to base
Posted by David Fanning on Fri, 22 Nov 2002 18:02:40 GMT
View Forum Message <> Reply to Message

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

> You are not getting carried away into the world of object widgets, are
> you? Seems to me that this would be a sure way to scare off for good
> anyone who doesn't write IDL code for living :-)

I'm just talking about simple ol'

```
   Widget_Control, someID, Send_Event={yourevent}
```

Of course, if you are feeling brave, or you want
to gain favor with the IEPA committee, an object
widget gives you all the bells and whistles. But
it is not necessary for simple widget-to-widget
communication.

And, anyway, I'm working on something that is going
to allow anyone (even you, Pavel) to write object
widget programs in the same way they write widget
programs today. Stay tuned. :-)

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: passing parameters from base to base
Posted by Pavel A. Romashkin on Fri, 22 Nov 2002 18:19:50 GMT

View Forum Message <> Reply to Message

David Fanning wrote:
>
> And, anyway, I'm working on something that is going
> to allow anyone (even you, Pavel) to write object
> widget programs in the same way they write widget
> programs today. Stay tuned. :-)

Won't apply to me. The 5.6 specific features will crash my obsolete
"box" :-(
Good luck though! Hopefully your version of GUI builder will be included
into the 5.7. Make sure it produces code at least as cryptic as the
current one :-) But then it won't matter because you self-altering,
transmogrifying, mind-reading objects will not need to expose the casual
user to any abstracted layers, will they ? :-)
Cheers,
Pavel

Subject: Re: passing parameters from base to base
Posted by David Fanning on Fri, 22 Nov 2002 18:28:02 GMT

View Forum Message <> Reply to Message

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

> But then it won't matter because you self-altering,
> transmogrifying, mind-reading objects will not need to expose the casual
> user to any abstracted layers, will they ? :-)

Well, see, that's the problem. :-(

Cheers,

David

P.S. Let's just say that having at least limited
psychic abilities is not a bad thing when you are
working with object programs. :-)

--
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: passing parameters from base to base
Posted by JD Smith on Mon, 25 Nov 2002 22:28:32 GMT
View Forum Message <> Reply to Message

On Thu, 21 Nov 2002 16:46:23 -0700, David Fanning wrote:

> Gert (Gert.Van.de.Wouwer@NOS_PAMpandora.be) writes:
>
>> I've been trying to figure this one out for a while. I have 2 bases. If
>> in Mainbase the button Set is pushed, a second base SetParams is
>> called. Stuff happens there and the idea is that if SetParams is
>> killed, a series of numbers go back to Mainbase. Now how can you write
>> this neatly, so that the code for the SetParams can easily be used in
>> other progs? These are my thoughts:
>> I could pass a pointer to SetParams that keeps the desired data, but
>> how does Mainbase knows that
>> SetParams is killed and that it needs to update its fields?
>
> If these are widget programs we are talking about, the simplest way of
> communicating between them is to send events back and forth. Typically,
> when program 1 calls program 2 it "registers" that it would like to
> receive updates if "something important" goes on. In my XColors program,

> for example, any program that wants to know about XColors loading a
> color table will register with either NOTIFYPRO, NOTIFYOBJ, or NOTIFYID
> keywords. Then, depending upon whether it is a procedure, object, or
> widget that you want to receive the notification, XCOLORS will use
> Call_Procedure, Call_Method, or SEND_EVENT (via Widget_Control) to
> notify program 1.
>

This is similar to a system I've developed over the years (and which hopefully will be available in the form of a suite of viewer tools "real soon now"). Instead of sending events, I abstract widget events and other forms of intercommunications among objects as "messages", and provide a framework for sending messages, signing up to receive those messages, etc. I call it all "Object Messaging", and "ObjMsg" is the parent class. Any ObjMsg object can send and receive messages from any other ObjMsg object. Widget programs can talk to non-widget programs, and all communication is treated the same way.

Want to get messages from somebody about new pudding flavors?

```
  somebody->MsgSignup,self,/PUDDING_FLAVORS
```

Tired of hearing about pudding flavors?

```
  somebody->MsgSignup,self,/NONE
```

etc. The flow of messages is not static; indeed it sometimes changes quite often during run time. At its essence, there is, of course, no fundamental difference between this mechanism and just carefully keeping track of which methods to call on which objects when, but the beauty is, it relieves the program from having to remember all this, and tends to promote smaller, more modular code.

For instance, my "tvDraw" object contains a draw widget, and sends all kinds of message, including simple motion events. At any given time, anywhere from 0 to ~10 different objects are interested in motion events. But tvDraw simply doesn't care about all that:

```
  self->MsgSend,/TVDRAW_MOTION
```

is sufficient to get all messages sent to all the relevant parties with no further effort. What this means is that, if later on you write another ObjMsg object which would like to hear about motion events, no new code is required.

The up-the-widget-heirarchy event passing paradigm is simple and useful, but for complex programs in encourages you to put everything in one place. Object Messages essentially breaks free from this paradigm: events and messages can be sent anywhere, at anytime.

Once freed from the shackles of object (and especially widget) intercommunication, you can better resist the urge to include everything but the kitchen sink in single burgeoning piles of code, and write small, focused objects, designed to solve one task well.

Of course, it's all smoke and mirrors until I show some code...

JD