Subject: Re: widget_control and group_leader
Posted by Pavel A. Romashkin on Sat, 23 Dec 2000 17:32:34 GMT
View Forum Message <> Reply to Message

Hi Nidhi,
I guess, the use of a common block is justified in this case.
What it boils down to, A does not need to be aware of B. It is the *
foreign_event_handler* that does. So, create a new common block in
B::init with one variable, SELF (or BtopID so that ou can use widget
control on it easily). Then let the foreign_event_handler access that
new common block and gain access to B from on an event received from A.
Trying to be elegant by avoiding common blocks is ok I guess, but then
again, if you think about it, Xmanager itself uses them, so why should
we be ashamed of it?
If you insist on not following "the evil path of Commons", and can
inspect the code for A, try to see if AtopID.uvalue or
AtopID.child.uvalue are not being used. Then, let B "worm" into A
without modifying the code of A at all, at the same time you do
Widget_control, AtopID, group_leader=BtopID. Just set the Uvalue of
AtopID or AtopID.child to BtopID and you will be able to gt it from
A-generated event.
Now, the last and the cleanest, I'd think, would be, at the time you do
widget_control, BtopID, group_leader=AtopID, is to create your own,
useless for A, *named* widget like this:

```
link_key = widget_base(AtopID, uname='Link_base_for_B', $
 map=0, xsize=1, ysize=1, uvalue=BtopID)
```

Then, from any event generated by A, you can call (although searching
for a widget is not my favorite way of programming, it can be justified
by those who hate common blocks)

```
link_key = widget_info(event.top, $
 find_by_name='Link_base_for_B')
```

and use link_key to get to B.self.

Hope this helps.
Cheers,
Pavel

nrk5@cornell.edu wrote:
>
> Lets say I have two widgets, A and B. There are two links between the
> two:
> 1) A.top and B.top are eachother's groupleaders, and
> 2) A uses common blocks and has a variable 'foreign_event_handler' that
> is set by B.

>
> So, when an event is generated by A and the 'Use Foreign Event Handler'
> option is set in the widget, events generated by A go to whatever B set
> 'foreign_event_handler' using:
>
>   widget_control, id, event_pro=foreign_event_handler
>
> Things to note:
> 1) A can't be modified at all. Nothing added or changed. (ie. no more
>   variables)
> 2) B is an object widget and needs to set its structure variables to
>   variables in the events generated by A.
> 3) In B::init, B.top has a uvalue of self.
>
> The question is, how can I use foreign_event_handler to get to 'B self'
> from an event generated by A? My thought was:
>
> PRO foreign_event_handler, eventFromA
>   widget_control, eventFromA.top, get_Group_Leader = BtopID
>   widget_control, BtopID, get_Uvalue = objectReferenceToB
>   ...
> END
>
> And now I would be in business. But, is there such as thing as
> get_group_leader? Is there another way to do this?
>
> I know that not being able to change A doesn't help, else there would be
> a million solutions, but its not my program. The only minor change I
> might be able to make is to create a generic variable in A's common
> block that could be set to whatever, but then I would have to define it
> as a string or a long, and that would restrict its use.
>
> Thanks much,
>
> -- Nidhi
> ----------------
> Nidhi Kalra
> nrk5@cornell.edu
>
> Sent via Deja.com
> http://www.deja.com/
