Subject: Re: Clearing widget events
Posted by Ben Tupper on Wed, 18 Jun 2003 16:07:34 GMT
View Forum Message <> Reply to Message

Ian Dean wrote:
> Hi All,
> Is it possible to clear any pending widget events without causing an
> existing event handler to be called?
>
> I have some software that establishes an event handler using the
> EVENT_PRO flag on a top-level base (I know I should use XMANAGER - but this
> is not suitable in this case).
> This works well, except that one function is time-consuming and users
> tend to hit the button (or other buttons) a number of times while waiting.
> What I would like to do is to clear the backlog of these events when the
> desired button action is complete.
> I have tried using WIDGET_EVENT with /NO_WAIT until no event is
> returned, but using this calls the current event handler, which I don't want
> at this point. I have also tried setting the EVENT_PRO to a null string
> before handling the desired function and then resetting it back afterwards.
> Neither of these methods are satisfactory and I was wondering if there
> is a better way.
>
> In anticipation,
> Ian
>
>

Hello,

I'm not sure how you are using the EVENT_PRO of your top level base without using Xmanager - so I'm not 100% sure I even understand the problem. But that won't restrain me from making suggestions; all my best stuff comes out of bliss-rich ignorance!

(1) There is the CLEAR_EVENTS keyword to WIDGET_CONTROL

From online help IDL 5.6

CLEAR_EVENTS
This keyword applies to all widgets.
If set, any events generated by the widget hierarchy rooted at Widget_ID which have arrived but have not been processed (via the WIDGET_EVENT procedure)are discarded.

(2) I have never tried setting the EVENT_PRO string to null after it has been set to the name of a real event handler, but I have often redirected events to a real dummy event handler - one that does nothing

except return immediately. When appropriate, you can then do as you do now - redirect events to the original event handler.

(3) You could make the the offending button insensitive (via SENSITIVE keyword to WIDGET_CONTROL) - but then you will lose your chance to abort early.

(4) You could chnage the name of the button, while processing, to "For pity's sake, don't press this button!"

How's that for wild hand-waving?

Ben

Subject: Re: Clearing widget events
Posted by MKatz843 on Wed, 18 Jun 2003 20:22:50 GMT
View Forum Message <> Reply to Message

> Is it possible to clear any pending widget events without causing an
> existing event handler to be called?

You can use
widget_control, widget_ID, /CLEAR_EVENTS
but you should consider a few things in advance.
When the computer is busy with a long calculation and the user hits the button repeatedly, I believe that the *operating system* may queue the events. This may affect your ability to clear the events since IDL may not have seen the events yet. Certainly an experiment will settle the issue quickly.

I've found that this can be a thorny issue when you give the user a "stop" button for something. If some other button is clicked several times and is taking a while to process things, then the stop button's event isn't able to magically jump ahead of the other queued events. So stop buttons can't solve itchy-button-finger syndrome.

So, that said, I think a solution you may want to consider is this.
When the user presses the button, set the cursor to hourglass so the user will see that it's calculating, and make the offending button insensitive until the calculation is complete.
widget_control, button_ID, SENSITIVE=0 or 1.
widget_control, HOURGLASS=1 or 0 (I think that's correct).

Then the button can't generate new events until you're ready.

MKatz

Subject: Re: Clearing widget events
Posted by Craig Markwardt on Wed, 18 Jun 2003 20:45:43 GMT
View Forum Message <> Reply to Message

MKatz843@onebox.com (M. Katz) writes:
>
> I've found that this can be a thorny issue when you give the user a
> "stop" button for something. If some other button is clicked several
> times and is taking a while to process things, then the stop button's
> event isn't able to magically jump ahead of the other queued events.
> So stop buttons can't solve itchy-button-finger syndrome.

Hmmm, I've found that when doing something like a "stop" button, I basically bypass XMANAGER, and simply poll events for that button using WIDGET_EVENT() at a convenient point in the computation loop. I am able to receive events for the "stop" button even if other buttons have been clicked first. Those other clicks will be ignored until the computation is finished and XMANAGER is able do deliver them again.

But your points about hourglass cursors and disabling buttons is certainly appropriate.

Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: Clearing widget events
Posted by JD Smith on Wed, 18 Jun 2003 21:08:26 GMT
View Forum Message <> Reply to Message

On Wed, 18 Jun 2003 09:07:34 -0700, Ben Tupper wrote:

> Ian Dean wrote:
>> Hi All,
>> Is it possible to clear any pending widget events without causing
>> an
>> existing event handler to be called?
>>
>> I have some software that establishes an event handler using the
>> EVENT_PRO flag on a top-level base (I know I should use XMANAGER - but
>> this is not suitable in this case).
>> This works well, except that one function is time-consuming and
>> users

>> tend to hit the button (or other buttons) a number of times while
>> waiting.
>> What I would like to do is to clear the backlog of these events
>> when the
>> desired button action is complete.
>> I have tried using WIDGET_EVENT with /NO_WAIT until no event is
>> returned, but using this calls the current event handler, which I don't
>> want at this point. I have also tried setting the EVENT_PRO to a null
>> string before handling the desired function and then resetting it back
>> afterwards.
>> Neither of these methods are satisfactory and I was wondering if
>> there
>> is a better way.
>>
>> In anticipation,
>> Ian
>>
>>
>>
> Hello,
>
> I'm not sure how you are using the EVENT_PRO of your top level base
> without using Xmanager - so I'm not 100% sure I even understand the
> problem. But that won't restrain me from making suggestions; all my
> best stuff comes out of bliss-rich ignorance!
>
> (1) There is the CLEAR_EVENTS keyword to WIDGET_CONTROL
>
> From online help IDL 5.6
>
> CLEAR_EVENTS
> This keyword applies to all widgets.
> If set, any events generated by the widget hierarchy rooted at Widget_ID
> which have arrived but have not been processed (via the WIDGET_EVENT
> procedure)are discarded.
>

Here's what I use, which isn't technically legal, using a forbidden widget_info keyword as it does:

__wa=widget_info(/managed) & for i=0,n_elements(__wa)-1 do widget_control,__wa[i],/clear_events

I have this bound to a keystroke, which also does a retall, getting me back in shape when a widget crashes with 100 motion events in the queue.

JD

Subject: Re: Clearing widget events
Posted by Mark Hadfield on Wed, 18 Jun 2003 21:38:56 GMT
View Forum Message <> Reply to Message

"Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message
news:onel1rglmw.fsf@cow.physics.wisc.edu...
>
> Hmmm, I've found that when doing something like a "stop" button, I
> basically bypass XMANAGER, and simply poll events for that button
> using WIDGET_EVENT() at a convenient point in the computation loop. I
> am able to receive events for the "stop" button even if other buttons
> have been clicked first. Those other clicks will be ignored until the
> computation is finished and XMANAGER is able do deliver them again.

Hmmm. when I want a widget application to be interruptible I add the
following code (buried inside a function) to the computation loop:

```
repeat begin
  event = widget_event(base, BAD_ID=bad_id, /NOWAIT)
endrep until event.id eq 0
```

Here base is the ID of the top-level base. This causes all events queued for
the base to be processed. The event handlers for the buttons themselves
don't have to do anything special.

Just my $0.02 worth. I don't know if this method is better or worse on
balance than Craig's. It makes programming a bit simpler, I think, but it
may be more dangerous because it doesn't stop the user from doing something
inappropriate, like triggering the window manager's "close" command.

BTW, I don't recall why I have BAD_ID in there. Probably for debugging.

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)