Subject: processing keyboard events in X
Posted by Aviv Gladman on Mon, 11 Aug 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Writing a GUI in IDL 5 (SunOS 5) and would like to be able to process keyboard interrupts from any window (as opposed to from stdin). Such an operation would be trivial in C/C++, but I can't seem to find any way of getting IDL to recognize that a key has been pressed while the mouse focus is on a draw widget, for example.Being unable to create widget hotkeys is, IMO, a ridiculous limitation, so I'm sure there must be a way to do it. Any ideas? Please respond via e-mail, if possible (mail reader is *very* slow).

Thanks,

Aviv S. Gladman, MASc

Subject: Re: processing keyboard events in X
Posted by davidf on Mon, 11 Aug 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Aviv Gladman writes:

> Writing a GUI in IDL 5 (SunOS 5) and would like to be able to process
> keyboard interrupts from any window (as opposed to from stdin). Such an
> operation would be trivial in C/C++, but I can't seem to find any way of
> getting IDL to recognize that a key has been pressed while the mouse
> focus is on a draw widget, for example.Being unable to create widget
> hotkeys is, IMO, a ridiculous limitation, so I'm sure there must be a way
> to do it.

One person's "ridiculous limitation" is often another person's "next to impossible task". This often asked for, but never seen (yet), capability is apparently extremely difficult to make work across the multiple platforms RSI supports. RSI tries (rightly, I think) not to be too aggressive at introducing features into IDL that work on one platform, but not on others.

I hear that this is "on the list" of things that they hope to do in the future.

Cheers,

David

David Fanning, Ph.D.

Subject: Re: processing keyboard events in X
Posted by J.D. Smith on Mon, 11 Aug 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Aviv Gladman wrote:
>
> Writing a GUI in IDL 5 (SunOS 5) and would like to be able to process
> keyboard interrupts from any window (as opposed to from stdin). Such an
> operation would be trivial in C/C++, but I can't seem to find any way of
> getting IDL to recognize that a key has been pressed while the mouse
> focus is on a draw widget, for example.Being unable to create widget
> hotkeys is, IMO, a ridiculous limitation, so I'm sure there must be a way
> to do it. Any ideas? Please respond via e-mail, if possible (mail reader
> is *very* slow).
>
> Thanks,
>
> Aviv S. Gladman, MASc

As David points out, this functionality is not built in. But if you're willing to settle for a hack, I have come up with one. The mouse focus and keyboard focus are independent, which you can use to your advantage. It's very simple, really. All you need to do is hide a widget_text widget with all_events set *underneath* your draw widget, and ensure its input focus is set when appropriate in the event handler. I have simply set focus every time through, but a better technique would do so only when entering the window, or when a button on the draw window is clicked (motion events do not remove the input focus), etc. I've tested this on Linux IDL v5.0, but no guarantees are made for other platforms (it *should* work on any of them though).

A few suggestions:

If you want the hotkeys active only while the cursor is inside the draw widget, you must enable widget tracking for the draw widget. On entering the window, set the input focus to the hidden text widget. On leaving the window, set it to something else (maybe just a "dummy" text widget which also lives under the draw widget but has only EDITABLE set). Conversely, you could just turn off the text events for the hidden widget with widget_control, hidden, ALL_TEXT_EVENTS=0, turning events back on for reentry. This method will cause a beep if the user presses a key while outside the draw widget (maybe you'd want that).

The method of setting focus to the "dummy" widget (with EDITABLE enabled) avoids this beep. The ALL_TEXT_EVENTS method might look like: widget_control, hidden, ALL_TEXT_EVENTS=ev.enter.

This technique is in no way tied to a draw widget and can be used in any widget application (as long as you can hide the text widget somewhere), but it will be more difficult when there are other text entry widgets... you must ensure your input focus returns to the hidden text widget when appropriate, which may become annoying for the user -- he might have to click to enter text each time. Encapsulating the functionality (by turning on and off as suggested for draw widgets) to those parts of the application for which it is useful might help.

The ability to turn the hotkeys on and off simply by changing the input focus can be put to good use... for instance, you could disable hotkeys when a click-and-drag operation is being done (that's actually done for you, since button presses change the input focus). But keep in mind that on most systems, the user can change the input_focus with the TAB key, so be careful, and use ALL_TEXT_EVENTS=0 when a danger exists. You could even employ KBRD_FOCUS events for this purpose.

Hope this proves useful.

JD

Here's the code:

```
pro testhotkey_event,ev
  stash=widget_info(ev.handler,/CHILD)
  widget_control, stash, get_uvalue=state,/NO_COPY
  name=tag_names(ev,/structure_name)

  case name of
    'WIDGET_TEXT_CH': $
     begin
       erase
       xyouts,.5,.5,/NORMAL,'Character: '+string(ev.ch)+'('+ $

strtrim(fix(ev.ch),2)+')',ALIGN=.5,CHARSIZE=3.0,COLOR=!D.N_C OLORS/2
       ls=['qQxX']
       wh=where(byte(ls) eq ev.ch,cnt)
       if cnt ne 0 then begin ;quit
          widget_control, ev.top,/DESTROY
          return
       endif
    end

    else: $                ;draw event
```

```
      begin
        types=['button event','motion event']
        erase
        seed=state.sd
        xyouts, .5,.5,/NORMAL,types[(ev.type-1)>0], ALIGNMENT=.5, $
         COLOR=fix(!d.n_colors*randomu(seed)),CHARSIZE=3.0
        state.sd=seed
      end
  endcase
  widget_control, state.hidden, /INPUT_FOCUS
  widget_control, stash,set_uvalue=state,/NO_COPY
  return
end

pro testhotkey
  loadct, 6
  base=widget_base(/COLUMN)
  label=widget_label(base,value='Hot Key Test')
  b=widget_base(base)
  sz=500
  draw=widget_draw(b,xsize=sz,ysize=sz,/motion,/button)
  hidden=widget_text(b,scr_xsize=1,scr_ysize=1,/ALL,FRAME=0)
  info={hidden:hidden,sd:0}
  widget_control, label,set_uvalue=info
  widget_control, b,/REALIZE
  widget_control, draw,get_value=win & wset,win
  XManager, 'testhotkey',base
  return
end
```

Subject: Re: processing keyboard events in X
Posted by Aviv Gladman on Tue, 12 Aug 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Ah yes, the "if we can't have it, no-one can" lowest common denominator theory of software design that makes Microsoft what it is today. I hope the "list" also includes elimination of the annoying (IMO, of course) comma that is used to separate elements in a matrix.

Aviv S. Gladman

On Mon, 11 Aug 1997, David Fanning wrote:

> [snip]
> One person's "ridiculous limitation" is often another person's
> "next to impossible task". This often asked for, but never seen
> (yet), capability is apparently extremely difficult to make work

> across the multiple platforms RSI supports. RSI tries (rightly,
> I think) not to be too aggressive at introducing features into
> IDL that work on one platform, but not on others.
>
> I hear that this is "on the list" of things that they hope to
> do in the future.
} [snip]

Subject: Re: processing keyboard events in X
Posted by davidf on Wed, 13 Aug 1997 07:00:00 GMT
View Forum Message <> Reply to Message

J.D. Smith writes:

> As David points out, this functionality is not built in. But if you're
> willing to settle for a hack, I have come up with one. The mouse focus
> and keyboard focus are independent, which you can use to your
> advantage. It's very simple, really. All you need to do is hide a
> widget_text widget with all_events set *underneath* your draw widget,
> and ensure its input focus is set when appropriate in the event
> handler. I have simply set focus every time through, but a better
> technique would do so only when entering the window, or when a button on
> the draw window is clicked (motion events do not remove the input
> focus), etc. I've tested this on Linux IDL v5.0, but no guarantees are
> made for other platforms (it *should* work on any of them though).

I made the mistake of reading J.D.'s code before I ran his example
program and I thought, "Nah, it ain't gonna work!". Shows you what
I know about IDL. :-)

Nice work, J.D.

There are some days when I just *LOVE* IDL. I think this is going
to be one of them. :-)

Cheers,

David

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

Subject: Re: processing keyboard events in X
Posted by Aviv Gladman on Wed, 13 Aug 1997 07:00:00 GMT
View Forum Message <> Reply to Message

A surprisingly effective and elegant "hack". The hidden text object works perfectly in SunOS 5. I assume the basis for this is the fact that motion events are passed through the text widget to the draw widget event handler, so that motion events can still be processed even if the focus is on the text widget. Haven't tried this on Win95, and I will admit to some curiosity, since Windows and X handle events in fundamentally different ways. Are we exploiting the properties of the X window manager, or of IDL?

Aviv S. Gladman

Imaging Research, Sunnybrook Health Science Centre, Toronto, Canada

On Mon, 11 Aug 1997, J.D. Smith wrote:

> Aviv Gladman wrote:
>>
>> Writing a GUI in IDL 5 (SunOS 5) and would like to be able to process
>> keyboard interrupts from any window (as opposed to from stdin). Such an
>> operation would be trivial in C/C++, but I can't seem to find any way of
>> getting IDL to recognize that a key has been pressed while the mouse
>> focus is on a draw widget, for example.Being unable to create widget
>> hotkeys is, IMO, a ridiculous limitation, so I'm sure there must be a way
>> to do it. Any ideas? Please respond via e-mail, if possible (mail reader
>> is *very* slow).
>>
>> Thanks,
>>
>> Aviv S. Gladman, MASc
>
> As David points out, this functionality is not built in. But if you're
> willing to settle for a hack, I have come up with one. The mouse focus
> and keyboard focus are independent, which you can use to your
> advantage. It's very simple, really. All you need to do is hide a
> widget_text widget with all_events set *underneath* your draw widget,
> and ensure its input focus is set when appropriate in the event
> handler. I have simply set focus every time through, but a better
> technique would do so only when entering the window, or when a button on
> the draw window is clicked (motion events do not remove the input
> focus), etc. I've tested this on Linux IDL v5.0, but no guarantees are
> made for other platforms (it *should* work on any of them though).
> [snip]