Subject: Re: X-windows text events
Posted by Niel Malan on Thu, 04 Apr 2002 16:22:12 GMT
View Forum Message <> Reply to Message

On Thu, 4 Apr 2002 06:39:43 -0700, David Fanning <david@dfanning.com> wrote:
> Niel Malan (Niel.Malan@aber.ac.uk) writes:
>
>> I'm writing a compound widget, basically a text widget with modified behaviour (returning the selected line as value)
>>
>> I'm using IDL 5.4, running an X-windows on Windows NT, the server being some Compaq workstation.
>>
>> My problem is that when my widget application is running, mouse actions in other X-windows, including the IDLDE, generate
>> events. At the moment the problem is small, only bothering me when editing in the IDLDE, but I see trouble ahead.
>>
>> Am I doing something wrong, or is this just the way things are?
>
> I guess I'd put my money on the "something wrong" theory.
> But it is impossible to say what with this short description.
>

The idea of my line-based text widget is that it should only select lines, return the selected line as GET_VALUE, and replace
the selected line with SET_VALUE. I do that by capturing events from a text widget, examining the position of the
cursor/selection, and then setting the selection to the beginning and end of that line. WIDGET_CONTROL's GET_VALUE and
SET_VALUE then uses this selection.

The compound widget works well, but there is some side-effect behaviour. When the widget is first started, behaviour is
normal. However, once the compound-widget event handler has handled some events, events are generated for text selection
attempts on all open X-windows. Sometimes this behavior disappears after working in MS windows, but returns once the event
handler has handled events: I can't yet replicate this at will.

The nett effect is that it is impossible to select text from other X-windows while my widget is running. This includes the
IDLE and IDL online help.

The code is below.

Niel

This is my compound widget, created from the IDL template:

```
;-Get & Set Value routines ------------------------------------------------
PRO textline_set_value, id, value
  stash = WIDGET_INFO(id, /CHILD)
  WIDGET_CONTROL, stash, GET_UVALUE=state, /NO_COPY

 ; Set the value here.
  WIDGET_CONTROL, state.id, SET_VALUE = value, /USE_TEXT_SELECT, /NO_NEWLINE

 ; Send an event so that the newly inserted text is selected
 ; If you don't do this the next SET_VALUE inserts text
 id = long(state.id)
 top = WIDGET_INFO(LONG(state.id), /PARENT)
 handler = 0L
 structure = {id: id, top: top, handler: handler}
  WIDGET_CONTROL, state.id, SEND_EVENT = structure

 ; Restore the state.
  WIDGET_CONTROL, stash, SET_UVALUE=state, /NO_COPY
END

FUNCTION textline_get_value, id
  COMPILE_OPT hidden
  ON_ERROR, 2
  stash = WIDGET_INFO(id, /CHILD)
  WIDGET_CONTROL, stash, GET_UVALUE=state, /NO_COPY

 ; Get the value here
  WIDGET_CONTROL, state.id, GET_VALUE = value, /USE_TEXT_SELECT
 ; Restore the state.
  WIDGET_CONTROL, stash, SET_UVALUE=state, /NO_COPY

        ;Return the value here.
 RETURN, value
END

;-----Compound Widget event  handler---------------------------------------------------- ------------------

FUNCTION textline_event, ev
  COMPILE_OPT hidden
  parent=ev.handler

 ;get the selection/cursor location
 selection = WIDGET_INFO(ev.id, /TEXT_SELECT)
 xy = WIDGET_INFO(ev.id, TEXT_OFFSET_TO_XY = selection[0])

 ; get the beginning of the first line of the text
```

```
 start = WIDGET_INFO(ev.id, TEXT_XY_TO_OFFSET = [0,xy[1]])

 ; get the beginning of the second line
 endt =   WIDGET_INFO(ev.id, TEXT_XY_TO_OFFSET = [0,xy[1]+1]) - 1
 ; handle out-of-bounds end: set end to end of text
 IF endt LT 0 THEN BEGIN
  endt = WIDGET_INFO(ev.id, /TEXT_NUMBER)
 ENDIF

 ; endt now must have valid value; handle invalid start by
 ; pointing start to the first position of the same line
 IF start LT 0 THEN BEGIN
   xy = WIDGET_INFO(ev.id, TEXT_OFFSET_TO_XY = endt - 1 )
   start = WIDGET_INFO(ev.id, TEXT_XY_TO_OFFSET = [0,xy[1]])
 ENDIF

 ; Set the selected text
 WIDGET_CONTROL, ev.id, SET_TEXT_SELECT = [start, endt-start]
;This print statement show when the event occurred
 print, start, endt, SYSTIME(0), FORMAT = '("Start: ",I4," End: ", I4, " Time: ", A0)'

  RETURN, { ID:parent, TOP:ev.top, HANDLER:0L }
END

;----Widget creation routine  ------------------------------------------------------------ -------------

FUNCTION cw_textline, parent, UVALUE = uval, UNAME = uname, VALUE = value, _EXTRA =
extra

  IF (N_PARAMS() EQ 0) THEN MESSAGE, 'Must specify a parent for cw_textline'

 ; Defaults for keywords
  IF NOT (KEYWORD_SET(uval))  THEN uval = 0
  IF NOT (KEYWORD_SET(uname))  THEN uname = 'cw_textline_UNAME'

  state = { id:0 }
  mainbase = WIDGET_BASE(parent, UVALUE = uval, UNAME = uname, $
  EVENT_FUNC = "textline_event", $
  FUNC_GET_VALUE = "textline_get_value", $
  PRO_SET_VALUE = "textline_set_value")
  state.id = WIDGET_TEXT(mainbase, VALUE = value, /ALL_EVENTS, YSIZE = 10)
  WIDGET_CONTROL, WIDGET_INFO(mainbase, /CHILD), SET_UVALUE=state, /NO_COPY
  RETURN, mainbase

END
;-----End of line-selecting compound widget  ------------------------------------------------------------ ---------
```

This is an example widget program that uses the compound widget:

```
; ------- Compound widget testing program ----------
PRO textline_test_event, event
 WIDGET_CONTROL, event.top, get_uvalue = tuv

 IF event.id EQ tuv.button1_id THEN BEGIN
  WIDGET_CONTROL, tuv.text_id, SET_VALUE = 'Replacement Text'
 ENDIF

 IF event.id EQ tuv.button2_id THEN BEGIN
  WIDGET_CONTROL, tuv.text_id, GET_VALUE = value
  PRINT, FORMAT = ('"Got value: ->",A0,"<-")', value
 ENDIF

 RETURN
END

PRO textline_test
 base = WIDGET_BASE(/ROW)
 text_id = cw_textline(base, VALUE = [ 'Line of text',$
      'Next Line', $
      'Another line',$
      'One more', $
      'But this is the last one'], $
    YSIZE = 10 )
 button1_id = widget_button(base, VALUE = '[Set Value]')
 button2_id = widget_button(base, VALUE = '[Get Value]')
 ids = {text_id :text_id, button1_id: button1_id, button2_id: button2_id}
 widget_control, base, SET_UVALUE = ids
 widget_control, base, /realize
 xmanager, 'textline_test', base, /NO_BLOCK
END

;------ End of testing program ----------------------------------
```