Subject: Re: Help with WIDGET_TEXT (non-) events
Posted by zawodny on Fri, 16 Feb 1996 08:00:00 GMT
View Forum Message <> Reply to Message

In article <4g01ge$irq@rosebud.sdsc.edu> Ken Knighton writes:
> Joseph M Zawodny wrote:
>> Hello,
>>
>> I have an application where I am currently using a non-editable
>> text widget to display a variable length list. The text widget has a
>> YSIZE of 1 and SCROLL set. This way only one item (the "current" item)
>> is displayed and the user can change the "current" item using the
>> vertical scroll bar.
>
> What you need is a compound widget with a label or text widget and a
> couple of arrow buttons to change the current value being displayed.
> A quick and dirty implementation of a "spinner" widget that handles
> sequential numeric values rather than a list of strings can be found
> at:
>
> ftp.rsinc.com

snip ...

Thanks Ken,

I have hacked, without remorse, a few chunks of code found at
ftp.rsinc.com and will post my resultant code at the end of this note.
What I was looking for was a way to present a text list in a scrollable
1-line viewport so the user could select an item from the complete list.
I wished to generate an event whenever the user "selected" from the list
by scrolling the list and have the event contain the current "selected"
value and an index to the position of this item in the list. In the long
run I would store in the UVALUE of this compound widget another list (or
vector of structures or widget_id's {which were unrealized, but had useful
UVALUES so I could, in theory, mix variable types}) of the same length.
I'd then use the index to tell me what data to use from this other list.

Anyway, I enclose below the code (CW_SPIN.PRO) for the compound widget and
a driver to demonstrate it's use (SPINTEST.PRO). While I detest the
appearence of bitmap widget_buttons and would rather have had access to a
real scrollbar, this does what I want in a minimal amount of space.

```
; ********* Begin CW_SPIN.PRO *********
function ARROWS, down=down
; This function draws tiny 5x5 bit-mapped arrows for display on widget buttons.
; Keyword DOWN:  Reverses the direction of the arrow so it points down.
```

```
; User needs to include X_BITMAP_EXTRA=3 in calls to WIDGET_BUTTON
 arrow = [[004B],[014B],[031B],[004B],[004B]]
 if keyword_set(down) then arrow = reverse(arrow, 2)
return, arrow
end

pro CW_SPIN_SET, id, v
; This is the procedure that will set a value for the spinner.
   ; Get it, set it, put it back
 child = widget_info(id, /child)
 widget_control, child, get_uvalue=s, /no_copy
 widget_control, s.tid, set_value=strtrim(v.list(v.ptr), 2)
 s.val = v
 widget_control, child, set_uvalue=s, /no_copy
return
end

function CW_SPIN_GET, id
   ; Get it and return it
 child = widget_info(id, /child)
 widget_control, child, get_uvalue=s, /no_copy
 val = s.val
 widget_control, child, set_uvalue=s, /no_copy
return, val
end

function CW_SPIN_EVENT, event
; Spinner event handler

   ; Get the state information out of the TLB of widget.
 child = widget_info(event.handler, /child)
 widget_control,         child, get_uvalue=s, /no_copy
; widget_control, event.handler, get_uvalue=list, /no_copy
 n = n_elements(s.val.list)-1

   ; Which widget was this?
 case event.id of
s.up: s.val.ptr = (s.val.ptr+1) < n
s.down: s.val.ptr = (s.val.ptr-1) > 0
else:
 endcase

; widget_control, event.handler, set_uvalue=list, /no_copy
 widget_control, s.tid, set_value=strtrim(s.val.list(s.val.ptr), 2)

   ; Desense buttons at the endpoints
 if (s.val.ptr eq 0) then sens = 0 else sens=1
 widget_control, s.down, sens=sens
```

```
 if (s.val.ptr eq n) then sens = 0 else sens=1
 widget_control, s.up, sens=sens

   ; Create an event structure.
 ev = {CW_SPIN, id:event.handler, top:event.top $
  , handler:0L, val:s.val.list(s.val.ptr), ptr:s.val.ptr}
 widget_control, child, set_uvalue=s, /no_copy
return, ev
end


function CW_SPIN, root, label=label, value=value, uvalue=uvalue
; Compound widget to display "current item" from a list.
; User selects item using "scroll buttons".
   ; Set defaults
 if n_elements( label) eq 0 then label  = 'Value'
 if n_elements( value) eq 0 then value  = $
  {ptr:0, list:['First','Second','Third']}
 if n_elements(uvalue) eq 0 then uvalue = ['A','B','C']

 tlb = widget_base(root,row=1,frame=1,uvalue=uvalue $
  , event_func     = 'cw_spin_event' $
  , pro_set_value  = 'cw_spin_set' $
  , func_get_value = 'cw_spin_get' )

 l0 = widget_label(tlb, value=label)
 t0 = widget_text( tlb, value=strtrim(value.list(value.ptr),2), xsize=6)
 b0 = widget_base( tlb,/column,ypad=1,space=1)
 up = widget_button(b0,x_bitmap_extra=3, value=arrows())
 dn = widget_button(b0,x_bitmap_extra=3, value=arrows(/down))

   ; Define a state variable and store it in first child.
 s = {up:up, down:dn, tid:t0, val:value}
 widget_control, l0, set_uvalue=s, /no_copy
 widget_control, dn, sens=0

return, tlb
end
; ********* END CW_SPIN.PRO *********
; ********* Begin SPINTEST.PRO *********
pro SPINTEST_EVENT, event
   ; Which widget caused the event?

 type = tag_names(event,/struct)

; Was it a CW_SPIN event
 if(type eq 'CW_SPIN') then begin
  widget_control,event.id,get_uval=uv, /no_copy
```

```
  print,'The '+event.val+' letter is '+uv(event.ptr)
  widget_control,event.id,set_uval=uv, /no_copy
  return
 endif

; Otherwise was WIDGET_BUTTON event
 widget_control, event.id, get_value=val

 case strupcase(val) of
'QUIT': Widget_Control, event.top, /Destroy
ELSE:
 endcase
return
end

pro SPINTEST
   ; This is a program to test whether the spinner program CW_Spin
   ; can get and set values appropriately.

 list=['First','Second','Third','Fourth','Fifth']

 r = widget_base(column=1, title='Spinner Test', align_left=1)

 b = widget_base( r, column=1, frame=1, scr_xsize=250)
 l = widget_label(b, value='This is a Spinner!')
 s =  cw_spin(b,value={ptr:0,list:list},uvalue=['A','B','C','D','E ])

 q = widget_button(r, value='Quit')

 widget_control, r, /realize, set_uvalue=s, /no_copy
 xmanager, 'spintest', r
return
end
; ********* End SPINTEST.PRO *********
```

Thanks again and have fun,

--

Dr. Joseph M. Zawodny KO4LW NASA Langley Research Center
E-mail: J.M.Zawodny@LaRC.NASA.gov MS-475, Hampton VA, 23681-0001