Subject: Widget program disappearing behind IDLDE
Posted by Benjamin Hornberger on Tue, 02 Nov 2004 22:17:15 GMT
View Forum Message <> Reply to Message

Hi all,

maybe somebody can explain this mysterious behaviour (Windows OS):

I have a widget program A from which I can start another widget program B. When B pops up, it makes A insensitive (I have reasons not to use the /modal keyword). When I close B, it will make A sensitive again (in its cleanup procedure). B has two buttons to close it (ok, cancel), plus the Windows-standard "X" on the upper right.

Now I have several programs open. If A is on top, and below is any program other than IDLDE, everything works fine. I pop up B, do my stuff there, and close it again (ok, cancel, or the "X"). A comes back and has the focus.

If A is the active program, and below it is the IDLDE, the following happens: I pop up B, and when I close it again via the ok or cancel buttons, A disappears behind the IDLDE (which gets the focus). The user who doesn't know might think A was closed. If I close B via the "X" on the upper right, everything is fine, A will be on top and have the focus.

If I add "widget_control, A, /show" after the "widget_control, A, /sensitive" in B's cleanup procedure, A will be on top after closing B via ok or cancel, but still IDLDE will have the focus (IDLDE's title bar is blue, A's title bar is grey).

Does anybody know what's going on? I am adding some code below.

Thanks for your help,

Benjamin

Some code from B (B is actually called histogram_gui). The technique with the 2 x n array of notify_ids is from David Fanning's book. histogram_gui_notify will send some info to A via widget_control, send_event=...

```
;;-------------------------------------------------------
;; The OK event handler
PRO histogram_gui_ok, event
  widget_control, event.top, /destroy
END
```

```
;;-------------------------------------------------------
;; The Cancel event handler
PRO histogram_gui_cancel, event
  widget_control, event.top, get_uvalue=info, /no_copy
  info.min = info.orig_min
  info.max = info.orig_max
  histogram_gui_notify, info
  widget_control, event.top, set_uvalue=info, /no_copy
  widget_control, event.top, /destroy
END


;;-------------------------------------------------------
;; the cleanup procedure
PRO histogram_gui_cleanup, tlb
  widget_control, tlb, get_uvalue=info, /no_copy
  IF n_elements(info) EQ 0 THEN return
  ptr_free, info.array
  ;; to make the calling widget sensitive again
  IF info.block_caller THEN BEGIN
     svec = size(info.notify_ids)
     IF svec[0] EQ 1 THEN count = 0 ELSE count = svec[2]-1
     FOR j=0, count DO BEGIN
        IF widget_info(info.notify_ids[1, j], /valid_id) THEN BEGIN
           widget_control, info.notify_ids[1, j], $
             sensitive=1
           widget_control, info.notify_ids[1, j], /show
        ENDIF
     ENDFOR
  ENDIF
  device, decomposed=info.decomposition_state
END
```