Subject: Help with pick menu widget code
Posted by Doug Larson on Tue, 15 Sep 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Hi All,
I am attempting to make a small program to pop up some menu buttons based
on a user set resource. I am attaching the code to this post in the hopes
that someone can tell me how to make it exit properly!

My intent was to understand how to handle different combinations of widgets
in one window. Even after repeated readings from "The Book of David" I still
can't get this to go.

Suggestions? Code fixes?

Thanks,
Doug

--
Douglas J. Larson e-mail: djl@loki.srl.caltech.edu
Space Radiation Lab
California Institute of Technology
Mail Code: 220-47
Pasadena, CA 91125

```
;  ********************************************************** **********
;+
; NAME: SETUP_PICK
;
; PURPOSE:
;       Define the structure for the data file type menu.
;
; KEYWORD PARAMETER:
;
;     Structure elements:
;
; COMMON BLOCKS: None
;
; MODIFICATION HISTORY:
;       Created: Douglas J. Larson, 04 August, 1998.
; 11 August, 1998 - Added arguments "bugmsg" and "nbuttons".
;
```

```
;-
;  ************************************************************ ******************
FUNCTION Setup_pick, bugmsg, nbuttons

 if ( bugmsg eq 1 ) then message,' Entered', /INFORMATIONAL

 pstruct = {  $
                parent:0L, $     ; Widget parent
                base:0L, $            ; Widget base
                wTitle:'', $          ; Widget title (if any)
     xsize:0L, $     ; Widget's x dimension
     ysize:0L, $           ; Widget's y dimension
     wxpos:0L, $     ; Widget's x screen position
     wypos:0L, $           ; Widget's y screen position
                rsrcID:0L, $          ; Resource ID number (unique)
                rsrcType:'PICK', $     ; Resource Type: Pick menu
                exflag:0L, $          ; 0-Exclusive, 1-NonExclusive
                savestate:0L, $        ; 0-Start state unsaved, 1-Saved
                buttonID:intarr(1), $  ; Current button event
                donecancel:0L, $       ; 0-DISMISS, 1-APPLY, 2-CLEAR, 3-RESET
                maxcount:0L, $
                picknames:strarr(nbuttons), $
                oldselcount:0L, $
                oldstate:intarr(nbuttons), $
                selcount:0L, $
                state:intarr(nbuttons), $
                menu_ids:lonarr(nbuttons) }

 if ( bugmsg eq 1 ) then message,' Exited', /INFORMATIONAL

 RETURN, pstruct
END
;  ************************************************************ ******************
;+
; NAME: pickmenu.pro
;
; PURPOSE:
;      Pop up a push button menu, either Exclusive or NonExclusive.
;
; COMMON BLOCKS:
; None.
;
; REQUIRED ROUTINES:
; Setup_pick  - Defines what a pick menu structure looks like.
; FUNCTION pickbutton_Event
; FUNCTION pickfinish_Event
;
; SIDE EFFECTS:
```

```
; None.
;
; VARIABLES:
; bugmsg  - 0-Don't be verbose about activity, 1-Give some progress msgs
; pickPtr - Pointer to the pickable menu's data structure. The structure
;    defines all the attributes of the menu. Some of which are:
;               parent       Widget parent
;               base         Widget base
;               wTitle       Widget title (if any)
;               rsrcID       Resource ID number (unique)
;               rsrcType     Resource Type: Pick menu
;               exflag       0-Exclusive, 1-NonExclusive
;               savestate    0-Start state unsaved, 1-Saved
;               donecancel   0-DISMISS, 1-APPLY, 2-CLEAR, 3-RESET
;               maxcount     Number of buttons
;               picknames    Button labels
;               oldselcount  Prior number of buttons selected
;               oldstate     Prior buttons selected
;               selcount     Number of buttons currently selected
;               state        Current buttons selected
;
; rsrcNum - Unique identifier of an application pick menu.
;
; OUTPUTS:
; Pops a button menu that looks like:
;
;          ------------------------------------------------------
;          | - | @ |        Main Window that has stuff            |
;          ======================================================
;          |                  Pick Menu Buttons                   |
;          |  o Item One   o Item Two   o Item Three   o Item Four  |
;          |                                                      |
;          |  o Item Five  o Item Six   o Item Seven              |
;          |                                                      |
;          | ---------------------------------------------------- |
;          |             DISMISS  APPLY  CLEAR  RESET             |
;          |======================================================|
;
; The button names for the menu, designated here with generic labels as
;       "Items," are contained in the structure pointed to by pickPtr with the
; tag "picknames."
; NonExclusive buttons: DISMISS/APPLY/CLEAR/RESET
; Exclusive buttons: DONE/CANCEL
;
; PROCEDURE:
;       pickmenu, DEBUG=bugmsg, PARENT=parent, PFIRST=pickPtr, RNUM=rsrcNum
; or without keywords:
; pickmenu
```

```
;
; IDL VERSION/COMPATIBILITY:
; Developed under IDL version 5.0.2 using SunOS 5.6 (UNIX).
;
; MODIFICATION HISTORY:
; Created by: Douglas J. Larson, 12 September, 1998.
;
;-
;  ********************************************************** ******************
PRO pickmenu, DEBUG=bugmsg, PARENT=parent, PFIRST=pickPtr, RNUM=rsrcNum

   if(N_PARAMS() EQ 0)then begin
     bugmsg = 1
     message, 'NO pickmenu parameters set, using defaults', /INFORMATIONAL
   endif

   if(bugmsg eq 1) then message,'Entered', /INFORMATIONAL

   if NOT(ptr_valid(pickPtr)) then begin
     message, 'pickPtr NOT set! Using default values.', /INFORMATIONAL
     pickPtr = ptr_new(Setup_pick(bugmsg,10))
   endif

   ; Set default values
   ; ------------------
   if((*pickPtr).wTitle EQ '') then (*pickPtr).wTitle = 'Exclusive Picker'
   if((*pickPtr).rsrcID EQ 0L) then (*pickPtr).rsrcID = 100L
   if((*pickPtr).rsrcTYPE EQ '') then (*pickPtr).rsrcTYPE = 'PICK'
   if((*pickPtr).exflag EQ 0L) then (*pickPtr).exflag = 0L
   if((*pickPtr).donecancel EQ 0L) then (*pickPtr).donecancel = -1L
   if((*pickPtr).maxcount EQ 0L) then begin
     (*pickPtr).maxcount = 10
     (*pickPtr).picknames = ['Newton', 'Copernicus', 'Kepler', 'Maxwell', $
     'Gauss', 'Lorentz', 'Curie', 'Van Allan', 'Feynmann', 'Einstein']
   endif
   nbcols = 5
   nbrows = (*pickPtr).maxcount/nbcols
   if((*pickPtr).xSize eq 0L) then (*pickPtr).xSize = 100*nbcols
   if((*pickPtr).ySize eq 0L) then (*pickPtr).ySize = 60*nbrows
   DEVICE, GET_SCREEN_SIZE=screenSize
   if((*pickPtr).wxpos eq 0L) then $
     (*pickPtr).wxpos = (screenSize(0) - (*pickPtr).xsize) / 2.
   if((*pickPtr).wypos eq 0L) then $
     (*pickPtr).wypos = (screenSize(1) - (*pickPtr).ysize) / 2.

   ; Create the Top Level Base (TLB) for the pick menu
   ; -------------------------------------------------
   ; First check for a valid widget id
```

```
have_parent = WIDGET_INFO (LONG ((*pickPtr).parent), /VALID_ID)
; Now make a base for the menu
if(have_parent)then begin
  pickTLB = WIDGET_BASE(Column = 1, $
        XSIZE = xsize, $
        XOFFSET=(*pickPtr).wxpos,YOFFSET=(*pickPtr).wypos, $
        /Base_Align_Center, /MODAL, /FLOATING, $
        GROUP_LEADER = (*pickPtr).parent)
endif else begin
  pickTLB = WIDGET_BASE(Column = 1, $
        XSIZE = xsize, $
        XOFFSET=(*pickPtr).wxpos,YOFFSET=(*pickPtr).wypos, $
        /Base_Align_Center)
  (*pickPtr).parent = pickTLB
endelse
title = WIDGET_BASE(pickTLB, /COLUMN, GROUP_LEADER = pickTLB)
label = WIDGET_LABEL(title, VALUE=(*pickPtr).wTitle )

btnbase = WIDGET_BASE(pickTLB, Column=1, Frame=1, $
              GROUP_LEADER = pickTLB)

ids=(*pickPtr).menu_ids[0]
if((*pickPtr).exflag eq 0) then begin
  buttons = CW_BGROUP(btnbase, $
        (*pickPtr).picknames[0:(*pickPtr).maxcount-1], $
        IDS = ids, $
        EVENT_FUNC='pickbutton_Event', $
        /EXCLUSIVE, $
        COLUMN=nbcols, /RETURN_ID, UVALUE=pickPtr)
endif else begin
  buttons = CW_BGROUP(btnbase, $
        (*pickPtr).picknames[0:(*pickPtr).maxcount-1], $
        IDS = ids, $
        EVENT_FUNC='pickbutton_Event', $
        /NONEXCLUSIVE, $
        COLUMN=nbcols, /RETURN_ID, UVALUE=pickPtr)
endelse
(*pickPtr).menu_ids=ids
pickmenu_set_state, pickPtr

; Make the bottom row buttons for the final disposition of the menu
;  ------------------------------------------------------------ -----
finishbase = WIDGET_BASE(pickTLB, $
                  EVENT_FUNC='pickfinish_Event', $
                  /ROW, GROUP_LEADER = pickTLB)
if((*pickPtr).exflag eq 0) then begin
  ; Make 'Done' and 'Cancel' buttons:
  done   = WIDGET_BUTTON(finishbase, VALUE = 'DONE')
```

```
      cancel = WIDGET_BUTTON(finishbase, VALUE = 'CANCEL')
    endif else begin
      ; Make 'Dismiss', 'Apply', 'Clear', and 'Reset' buttons:
      dismiss = WIDGET_BUTTON(finishbase, VALUE = 'DISMISS')
      apply   = WIDGET_BUTTON(finishbase, VALUE = 'APPLY')
      clear   = WIDGET_BUTTON(finishbase, VALUE = 'CLEAR')
      reset   = WIDGET_BUTTON(finishbase, VALUE = 'RESET')
    endelse

    ; Realize the widgets:
    WIDGET_CONTROL, pickTLB, /REALIZE
    WIDGET_CONTROL, pickTLB, Set_UValue=pickPtr, /NO_COPY

    XMANAGER, 'picked', pickTLB, GROUP_LEADER = pickTLB

END
;  ************************************************************ *********
PRO pickmenu_set_state, s
    if((*s).savestate eq 0)then begin
      (*s).savestate = 1
      (*s).oldselcount = 0
      for i = 0, (*s).maxcount-1 do begin
        (*s).oldstate[i]=0
      endfor
    endif else begin
      for i = 0, (*s).maxcount-1 do begin
        (*s).state[i]=0
      endfor
    endelse
END
;  ************************************************************ *********
FUNCTION pickbutton_Event, event

; Retrieve the data we stored in the top level base
WIDGET_CONTROL, event.top, GET_UVALUE = pickPtr
WIDGET_CONTROL, event.id, GET_VALUE = eventvalue
print,'pickbutton_Event: eventvalue=',eventvalue
eventname = Tag_Names(event, /Structure_Name)
print,'pickbutton_Event: eventname=',eventname

IF(where(tag_names(event) EQ 'VALUE'))[0] NE -1 THEN BEGIN
  button_ID = WHERE((*pickPtr).menu_ids EQ event.value, count)
  print,'pickbutton_Event: button_ID=',button_ID
  print,'pickbutton_Event: button name=',(*pickPtr).picknames[button_ID]
  (*pickPtr).buttonID = button_ID
  IF((count gt 0) and (count le (*pickPtr).maxcount-1))THEN BEGIN
    (*pickPtr).state[button_id] = 1
  ENDIF
```

```
ENDIF

WIDGET_CONTROL, event.top, SET_UVALUE= pickPtr, /NO_COPY
RETURN, 1
END
;  ************************************************************ *********
FUNCTION pickfinish_Event, event

eventname = Tag_Names(event, /Structure_Name)
print,'pickfinish_event: eventname=',eventname
IF eventname NE 'WIDGET_BUTTON' THEN RETURN, 1

; Retrieve the data we stored in the top level base
WIDGET_CONTROL, event.id, GET_VALUE = buttonvalue
print,'pickfinish_event: buttonvalue=',buttonvalue

CASE buttonvalue OF
    'DONE': BEGIN
      (*pickPtr).donecancel = 0L
      WIDGET_CONTROL, event.top, /DESTROY
      END
    'CANCEL': BEGIN
      (*pickPtr).donecancel = 1L
      (*pickPtr).selcount = (*pickPtr).oldselcount
      for i = 0, (*pickPtr).maxcount-1 do begin
         (*pickPtr).state[i]=(*pickPtr).oldstate[i]
      endfor
      WIDGET_CONTROL, event.top, /DESTROY
      END
    'DISMISS': BEGIN
      (*pickPtr).donecancel = 0L
      WIDGET_CONTROL, event.top, /DESTROY
      END
    'APPLY': BEGIN
      (*pickPtr).donecancel = 1L
      (*pickPtr).oldselcount = (*pickPtr).selcount
      for i = 0, (*pickPtr).maxcount-1 do begin
        (*pickPtr).oldstate[i]=(*pickPtr).state[i]
      endfor
      WIDGET_CONTROL, event.top, /DESTROY
      END
    'CLEAR': BEGIN
      (*pickPtr).donecancel = 2L
      ; DeSelect all pick menu buttons
      for i = 0, (*pickPtr).maxcount-1 do begin
         (*pickPtr).state[i] = 0
         ; Clear ALL the buttons:
         WIDGET_CONTROL, (*pickPtr).menu_ids[i], SET_BUTTON=0
```

```
      endfor
      (*pickPtr).selcount = 0
      END
    'RESET': BEGIN
      (*pickPtr).donecancel = 3L
      ; Restore prior settings of pick menu buttons
      for i = 0, (*pickPtr).maxcount-1 do begin
         (*pickPtr).state[i]=(*pickPtr).oldstate[i]
         WIDGET_CONTROL, (*pickPtr).menu_ids[i], $
                  SET_BUTTON=(*pickPtr).state[i]
      endfor
      (*pickPtr).selcount = (*pickPtr).oldselcount
      END
ENDCASE

WIDGET_CONTROL, event.top, SET_UVALUE= pickPtr, /NO_COPY
RETURN, 1
END
```

File Attachments

1) plea.let, downloaded 108 times