Subject: Multiple Selection List Widget
Posted by davidf on Fri, 23 Jan 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Hi Folks,

Deb Summa was looking for a multiple selection list widget the other day. I mentioned that there was no good way to implement such a thing in IDL, but there were hacks. Deb was kind enough to call that bluff, so here is a hack. :-)

I don't pretend its elegant or the last word on selection widgets, but it took me an hour or so while I was waiting for the kids to get the movie started.

I wrote the list selection part as a compound widget that has an Accept Selection button. Selecting or clicking items in the list toggles the item on or off. These events are "swallowed" by the compound widget. When you hit the Accept Selection button, the compound widget generates an anonymous structure event with an INDEX field. This field is a vector with as many elements as there are items in the list. A one indicates the item is selected, a zero indicates is is not selected. The program using the compound widget can do whatever it likes with the event.

You can get and set the value of the list widget in the normal way. I've included a little TEST program at the bottom of the code to exercise the compound widget, named CW_MLIST. Simply compile this code and type "TEST" to run the program.

The program has NO documentation (tired of writing), but it is pretty trivial so I think you will have no trouble understanding how it works. If I get a few minutes later on, I will probably write this up as a tip and put some documentation in it. :-)

Cheers,

David

```
**************************************************
PRO CW_MList_Cleanup, id

   ; Cleans up compound widget pointers.
```

```
Widget_Control, id, Get_UValue=info, /No_Copy
IF N_Elements(info) NE 0 THEN BEGIN
   Ptr_Free, info.value
   Ptr_Free, info.on
ENDIF
END;-----------------------------------------------



PRO CW_MList_Set, id, value

   ; Sets the value of the list widget.

child = Widget_Info(id, /Child)
Widget_Control, child, Get_UValue=info, /No_Copy
value = '   ' + value
*info.on = BytArr(N_Elements(value))
*info.value = value
Widget_Control, info.listID, Set_Value=value
Widget_Control, child, Set_UValue=info, /No_Copy
END;-----------------------------------------------



FUNCTION CW_MList_Get, id

   ; Gets the value of the list widget.

child = Widget_Info(id, /Child)
Widget_Control, child, Get_UValue=info, /No_Copy
length = Max(StrLen(*info.value))
value = StrMid(*info.value, 3, length)
Widget_Control, child, Set_UValue=info, /No_Copy
RETURN, value
END;-----------------------------------------------



FUNCTION CW_MList_Events, event

   ; Handles compound widget events. All list widget
   ; events are swallowed.

thisEvent = Tag_Names(event, /Structure_Name)

IF thisEvent EQ 'WIDGET_LIST' THEN BEGIN
   child = Widget_Info(event.handler, /Child)
   Widget_Control, child, Get_UValue=info, /No_Copy
```

```
   IF (*info.on)(event.index) THEN $
      (*info.on)(event.index) = 0 ELSE $
      (*info.on)(event.index) = 1

   thisItem = (*info.value)(event.index)
   length = StrLen(thisItem)
   IF (*info.on)(event.index) THEN BEGIN
      StrPut,thisItem,'*', 1
   ENDIF ELSE BEGIN
      StrPut,thisItem,' ', 1
   ENDELSE
   (*info.value)(event.index) = thisItem
   Widget_Control, event.id, Set_Value=*info.value
   Widget_Control, child, Set_UValue=info, /No_Copy
   RETURN, 0
ENDIF

IF thisEvent EQ 'WIDGET_BUTTON' THEN BEGIN
   child = Widget_Info(event.handler, /Child)
   Widget_Control, child, Get_UValue=info, /No_Copy
   on = *info.on
   cw_tlb = info.cw_tlb
   Widget_Control, child, Set_UValue=info, /No_Copy
   RETURN, {ID:cw_tlb, TOP:event.top, $
            HANDLER:0L, INDEX:on}
ENDIF
END;-----------------------------------------------



FUNCTION CW_MLIST, parent, Value=value, UValue=uvalue, $
   _Extra=extra

IF N_Elements(value) EQ 0 THEN value=''
IF N_Elements(uvalue) EQ 0 THEN uvalue='CW_MLIST'

value = '    ' + value

cw_tlb = Widget_Base(parent, $
   UValue=uvalue, $
   Column=1, $
   Frame=1, $
   Event_Func='CW_MList_Events', $
   Pro_Set_Value='CW_MList_Set', $
   Func_Get_Value='CW_MList_Get')
listID = Widget_List(cw_tlb, Value=value, _Extra=extra, $
   YSize=N_Elements(value))
```

```
buttonID = Widget_Button(cw_tlb, Value='Accept Selections')

info = { listID:listID, $
        buttonID:buttonID, $
        cw_tlb:cw_tlb, $
        on:Ptr_New(BytArr(N_Elements(value))), $
        value:Ptr_New(value) }

Widget_Control, listID, Set_UValue=info, /No_Copy, $
  Kill_Notify='CW_MList_Cleanup'
RETURN, cw_tlb
END;-------------------------------------------------



PRO Test_Event, event
Widget_Control, event.top, Get_UValue=wlist
Widget_Control, event.id, Get_UValue=buttonEvent

CASE buttonEvent OF
  'MLIST': BEGIN
     items = Where(event.index EQ 1, count)
     IF count EQ 0 THEN items = 'None Selected'
     Print, 'Selected Items: ', items
     END
  'GET': BEGIN
     Widget_Control, wlist, Get_Value=thisList
     Print, thisList
     ENDCASE
  'SET': BEGIN
     newList = ['Man', 'Woman', 'Alien', 'Whosit', 'Thing']
     Widget_Control, wList, Set_Value=newList
     ENDCASE
  'QUIT': Widget_Control, event.top, /Destroy
ENDCASE
END;-------------------------------------------------



PRO Test
tlb = Widget_Base(Column=2)
leftbase = Widget_Base(tlb, /Column)
rtbase = Widget_Base(tlb,/Column)
data = ['dog', 'cat','mouse', 'coyote']
wlist = CW_MList(leftbase, value=data, Ysize=4, UValue='MLIST')
get = Widget_Button(rtbase, value='Get List Values', UValue='GET')
set = Widget_Button(rtbase, value='Set List Values', UValue='SET')
quit = Widget_Button(rtbase, value='Quit', UValue='QUIT')
```

```
Widget_Control, tlb, /Realize, Set_UValue=wlist
XManager, 'test', tlb, /No_Block
END;-------------------------------------------------
```

****

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