Subject: Re: Contributed Source: CW_selectaxes (Corrected and working!)
Posted by Doug Larson on Mon, 28 Sep 1998 07:00:00 GMT
View Forum Message <> Reply to Message

For some reason the source did not attach properly.
Here it is (I hope) :)

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

```
; ************************************************************ *****************
;+
; NAME:
;      CW_SelectAxes
;
; PURPOSE:
;      A compound widget to select axes for an arbitrary plotting application.
;      The default is (X,Y) but the DIMENSION keyword can be set to 3 for Z.
;
; AUTHOR:
;      Douglas J. Larson, Ph.D.
;      Space Radiation Lab
;      Mail Code: 220-47
;      California Institute of Technology
;      Pasadena, California  91125
;      Email: djl@srl.caltech.edu
;
; MODULES REQUIRED:
;      CW_SelectAxes_get_value
;      CW_SelectAxes_set_value
;      CW_SelectAxes_event
;      CW_SelectAxes
;      These two modules test the CW_SelectAxes modules:
;      test_CW_SelectAxes_event
;      test_CW_SelectAxes
;
; CATEGORY:
;      WIDGETS
;
; CALLING SEQUENCE:
;      result = SelectAxes(parent)
;
```

```
; INPUTS:
;       Parent:        The ID of the parent widget.
;
; OPTIONAL KEYWORD PARAMETERS:
;       LABELLIST:     Default is ['X Axis', 'Y Axis', ['Z Axis']]
;       LABELFONT:     Font used for the list labels
;       LABELPLACE:    0: TOP of each list
;                      1: LEFT for row layout, RIGHT for column
;       LISTNAMES:     Names of the list elements, copied for all axes
;       LISTFONT:      Font used for the list elements
;       LISTPLACE:     0: Lists laid out as a ROW
;                      1: Lists laid out vertically in a COLUMN
;       TITLEAS:       The CW title title (The default is no title.)
;       TITLEFONT:     Font to use for the select axes title
;       TITLEPLACE:    0: TOP
;                      1: LEFT SIDE
;       FRAME:         Set this keyword to have a frame drawn around the
;                      widget. The default is FRAME=0.
;       UVALUE:        The "user value" to be assigned to the widget.
;       VALUE:         The initial value setting of the widget.
;       XSIZE:         Width of the list widget in characters.
;       YSIZE:         Height of the list widget in characters.
;
; OUTPUTS:
;       The ID of the created widget is returned.
;
; SIDE EFFECTS:
;       This widget generates event structures containing a field
;       named value when its selection thumb is moved.
;
; PROCEDURE:
;       WIDGET_CONTROL, id, SET_VALUE=value can be used to change the
;             current value displayed by the widget.
;
;       WIDGET_CONTROL, id, GET_VALUE=var can be used to obtain the current
;             values displayed by the widget.
;
; MODIFICATION HISTORY:
;       Created by: Douglas J. Larson, 24 September 1998
;       26 September, 1998 - Changed the default font to be OS aware.
;             Implemented the fix to my return structure suggeested by
;             David Fanning, now it works!
;
;-
; ==============================================================
=================
FUNCTION CW_SelectAxes_get_value, id
   ; Recover the state of this compound widget
```

```
  stash = WIDGET_INFO(id, /CHILD)
  WIDGET_CONTROL, stash, GET_UVALUE=state, /NO_COPY

  if(state.dimension eq 2)then $
    ret = [ state.ListIndexX, state.ListIndexY ]

  if(state.dimension eq 3)then $
    ret = [ state.ListIndexX, state.ListIndexY, state.ListIndexZ ]

  WIDGET_CONTROL, stash, SET_UVALUE=state, /NO_COPY
  return, ret
end
;  ---------------------------------------------------------- -----------------
FUNCTION CW_SelectAxes_set_value, id
  ; Recover the state of this compound widget
  stash = WIDGET_INFO(id, /CHILD)
  WIDGET_CONTROL, stash, GET_UVALUE=state, /NO_COPY

  if(state.selectList eq 0)then begin
    state.ListIndexX = value
  endif

  if(state.selectList eq 1)then begin
    state.ListIndexY = value
  endif

  if(state.selectList eq 2)then begin
    state.ListNamesZ = value
  endif

  WIDGET_CONTROL, stash, SET_UVALUE=state, /NO_COPY
end
;  ---------------------------------------------------------- -----------------
FUNCTION CW_SelectAxes_event, event

; Retrieve the structure from the child that contains the sub ids
parent=event.handler
stash = WIDGET_INFO(parent, /CHILD)
WIDGET_CONTROL, stash, GET_UVALUE=state, /NO_COPY
eventValue = event.index

  ; See which list was clicked, the X, Y, or Z axis.
  CASE event.ID OF
    state.xaxisID : BEGIN
       state.selectList = 0
       selectNames = state.listNamesX
       if(state.bugmsg eq 1)then print,'Xaxis List=',eventValue
       END
```

```
      state.yaxisID : BEGIN
         state.selectList = 1
         selectNames = state.listNamesY
         if(state.bugmsg eq 1)then print,'Yaxis List=',eventValue
         END
      state.zaxisID : BEGIN
         state.selectList = 2
         selectNames = state.listNamesZ
         if(state.bugmsg eq 1)then print,'Zaxis List=',eventValue
         END
   ENDCASE

   ret = { SelectAxes_event, ID:parent, TOP:event.top, HANDLER:0L, $
         axis:state.selectList, index:event.index, $
         items:Ptr_New(selectNames) }

WIDGET_CONTROL, stash, SET_UVALUE = state, /NO_COPY

RETURN, ret

END
;  ---------------------------------------------------------- ------------------
FUNCTION CW_SelectAxes, parent, DEBUG=bugmsg, DIMENSION = dimension, $
      LABELLIST=labellist, LABELFONT=labelfont, LABELPLACE=LabelPlace, $
      TITLEAS = titleas, TITLEFONT=titlefont, TITLEPLACE=titleplace, $
      FRAME = frame, UVALUE = uval, VALUE = val, $
      LISTNAMES=listnames, LISTFONT=listfont, LISTPLACE=listplace, $
      XAXISLIST = ListNamesX, $
      YAXISLIST = ListNamesY, $
      ZAXISLIST = ListNamesZ, $
      XSIZE = xsize, YSIZE = ysize, ZSIZE = zsize

   if(N_PARAMS() EQ 0) then MESSAGE, 'Incorrect number of arguments'
   ON_ERROR, 2      ;return to caller

   ; Defaults for keywords
   ; ---------------------
   if NOT(KEYWORD_SET(bugmsg)) then bugmsg = 0
   if N_ELEMENTS(dimension) EQ 0 then dimension = 2
   if N_ELEMENTS(labellist) EQ 0 then begin
      CASE dimension OF
      2: begin
         labellist = STRARR(2)
         labellist[0] = 'X Axis'
         labellist[1] = 'Y Axis'
         end
      3: begin
         labellist = STRARR(3)
      labellist[0] = 'X Axis'
      labellist[1] = 'Y Axis'
      labellist[2] = 'Z Axis'
      end
    ENDCASE
  endif
  if NOT(KEYWORD_SET(labelfont))then begin
    case !version.os_family of
     'Windows': BEGIN
            labelfont = 'times'
            END
     'MacOS': BEGIN
            labelfont = 'times'
            END
     'unix' : BEGIN
            labelfont = $
             '-adobe-helvetica-bold-r-normal--12-120-75-75-p-70-iso8859-1 '
            END
    endcase
  endif
  if N_ELEMENTS(LabelPlace) EQ 0   then LabelPlace = 0
  if N_ELEMENTS(titleas) EQ 0      then titleas = ""
  if N_ELEMENTS(titlefont) EQ 0    then titlefont = labelfont
  if N_ELEMENTS(titleplace) EQ 0   then titleplace = 0
  if N_ELEMENTS(frame) EQ 0     then frame = 0
  if N_ELEMENTS(uval) EQ 0      then uval = 0
  if N_ELEMENTS(val) EQ 0       then val = 0
  if N_ELEMENTS(xsize) EQ 0     then xsize = 6
  if N_ELEMENTS(ysize) EQ 0     then ysize = 6
  if N_ELEMENTS(listfont) EQ 0  then listfont = labelfont
  if N_ELEMENTS(listplace) EQ 0 then listplace = 0

  ; The actual list elements can be specified on a per list
  ; basis, or once with the LISTNAMES keyword.
  nListNames  = N_ELEMENTS(ListNames)
  nListNamesX = N_ELEMENTS(ListNamesX)
  nListNamesY = N_ELEMENTS(ListNamesY)
  nListNamesZ = N_ELEMENTS(ListNamesZ)
  if(nListNames eq 0)then begin
    filldefaults = 0L
    CASE dimension OF
      2: begin
        if((nListNamesX EQ 0) or (nListNamesY EQ 0)) then filldefaults = 1L
        end
      3: begin
        if((nListNamesX EQ 0) or $
          (nListNamesY EQ 0) or $
          (nListNamesZ EQ 0)) then begin
          filldefaults = 1L
        endif
        end
    ENDCASE
    if(filldefaults EQ 1L) then begin
      nListNamesX = 10
      nListNamesY = nListNamesX
      nListNamesZ = nListNamesX
      ListNamesX = STRARR(nListNamesX)
      ListNamesY = STRARR(nListNamesY)
      ListNamesZ = STRARR(nListNamesZ)
      for i = 0, nListNamesX-1 do begin
         name = 'Value '+strcompress(STRING(i))
         ListNamesX[i] = name
         ListNamesY[i] = name
         ListNamesZ[i] = name
      endfor
    endif
  endif else begin
    nListNamesX = nListNames
    nListNamesY = nListNames
    nListNamesZ = nListNames
    ListNamesX = STRARR(nListNames)
    ListNamesY = STRARR(nListNames)
    ListNamesZ = STRARR(nListNames)
    ListNamesX = ListNames[*]
    ListNamesY = ListNames[*]
    ListNamesZ = ListNames[*]
  endelse

  ; Setup placement of a title for this CW
  ; ---------------------------------------
  if(titleplace eq 0)then begin         ; COLUMN
    SelectAxesTLB=WIDGET_BASE(parent, /COLUMN)
    titleBase = WIDGET_BASE(SelectAxesTLB, /ALIGN_CENTER)
  endif else begin                      ; ROW
    SelectAxesTLB=WIDGET_BASE(parent, /ROW)
    titleBase = WIDGET_BASE(SelectAxesTLB, /ALIGN_CENTER)
  endelse
  title = WIDGET_LABEL(titleBase, VALUE=titleas, FONT=titlefont)

  ; Setup layout of lists as a row or a column
  ; ------------------------------------------
  if(listplace eq 0)then begin         ; ROW
    SelectAxesBase=WIDGET_BASE(SelectAxesTLB, /ROW)

    ; Setup placement of each individual list's title
    ; ----------------------------------------------
```

```
CASE LabelPlace OF
   0 : BEGIN
      xaxisBase = WIDGET_BASE(SelectAxesBase, GROUP_LEADER = parent, /COL)
      labelxBase = WIDGET_BASE(xaxisBase, /ALIGN_CENTER, /COL)
      labelx = WIDGET_LABEL(labelxBase, VALUE=labellist[0], $
             FONT=labelfont)
      xaxisID = WIDGET_LIST(xaxisBase, VALUE=ListNamesX, $
              UVALUE=labellist[0], $
              FONT=listfont, $
              XSIZE=xsize, YSIZE=ysize)

      yaxisBase = WIDGET_BASE(SelectAxesBase, GROUP_LEADER = parent, /COL)
      labelyBase = WIDGET_BASE(yaxisBase, /ALIGN_CENTER, /COL)
      labely = WIDGET_LABEL(labelyBase, VALUE=labellist[1], $
             FONT=labelfont)
      yaxisID = WIDGET_LIST(yaxisBase, VALUE=ListNamesY, $
              UVALUE=labellist[1], $
              FONT=listfont, $
              XSIZE=xsize, YSIZE=ysize)

      if(dimension eq 3)then begin
        zaxisBase = WIDGET_BASE(SelectAxesBase, $
                 GROUP_LEADER = parent, /COL)
        labelzbase = WIDGET_BASE(zaxisBase, /ALIGN_CENTER, /COL)
        labelz = WIDGET_LABEL(labelzbase, VALUE=labellist[2], $
               FONT=labelfont)
        zaxisID = WIDGET_LIST(zaxisBase, VALUE=ListNamesZ, $
                UVALUE=labellist[2], $
                FONT=listfont, $
                XSIZE=xsize, YSIZE=ysize)

      endif else begin
        zaxisID = 0L
      endelse
      END
   1 : BEGIN
      xaxisBase = WIDGET_BASE(SelectAxesBase, GROUP_LEADER = parent, /ROW)
      labelxBase = WIDGET_BASE(xaxisBase, /ALIGN_CENTER)
      labelx = WIDGET_LABEL(labelxBase, VALUE=labellist[0], $
             FONT=labelfont)
      xaxisID = WIDGET_LIST(xaxisBase, VALUE=ListNamesX, $
              UVALUE=labellist[0], $
              FONT=listfont, $
              XSIZE=xsize, YSIZE=ysize)

      yaxisBase = WIDGET_BASE(SelectAxesBase, GROUP_LEADER = parent, /ROW)
      labelyBase = WIDGET_BASE(yaxisBase, /ALIGN_CENTER)
      labely = WIDGET_LABEL(labelyBase, VALUE=labellist[1], $
              FONT=labelfont)
          yaxisID = WIDGET_LIST(yaxisBase, VALUE=ListNamesY, $
               UVALUE=labellist[1], $
               FONT=listfont, $
               XSIZE=xsize, YSIZE=ysize)

          if(dimension eq 3)then begin
            zaxisBase = WIDGET_BASE(SelectAxesBase, $
                  GROUP_LEADER = parent, /ROW)
            labelzbase = WIDGET_BASE(zaxisBase, /ALIGN_CENTER)
            labelz = WIDGET_LABEL(labelzbase, VALUE=labellist[2], $
                 FONT=labelfont)
            zaxisID = WIDGET_LIST(zaxisBase, VALUE=ListNamesZ, $
                 UVALUE=labellist[2], $
                 FONT=listfont, $
                 XSIZE=xsize, YSIZE=ysize)
          endif else begin
            zaxisID = 0L
          endelse
          END
      ENDCASE
    endif else begin                  ; COLUMN
      SelectAxesBase=WIDGET_BASE(SelectAxesTLB, /COLUMN)

      ; Setup placement of each individual list's title
      ; ---------------------------------------------
      CASE LabelPlace OF
        0 : BEGIN
          xaxisBase = WIDGET_BASE(SelectAxesBase, GROUP_LEADER = parent, /COL)
          labelxBase = WIDGET_BASE(xaxisBase, /ALIGN_CENTER, /COL)
          labelx = WIDGET_LABEL(labelxBase, VALUE=labellist[0], $
              FONT=labelfont)
          xaxisID = WIDGET_LIST(xaxisBase, VALUE=ListNamesX, $
               UVALUE=labellist[0], $
               FONT=listfont, $
               XSIZE=xsize, YSIZE=ysize)

          yaxisBase = WIDGET_BASE(SelectAxesBase, GROUP_LEADER = parent, /COL)
          labelyBase = WIDGET_BASE(yaxisBase, /ALIGN_CENTER, /COL)
          labely = WIDGET_LABEL(labelyBase, VALUE=labellist[1], $
              FONT=labelfont)
          yaxisID = WIDGET_LIST(yaxisBase, VALUE=ListNamesY, $
               UVALUE=labellist[1], $
               FONT=listfont, $
               XSIZE=xsize, YSIZE=ysize)

          if(dimension eq 3)then begin
            zaxisBase = WIDGET_BASE(SelectAxesBase, $
                    GROUP_LEADER = parent, /COL)
            labelzbase = WIDGET_BASE(zaxisBase, /ALIGN_CENTER, /COL)
            labelz = WIDGET_LABEL(labelzbase, VALUE=labellist[2], $
                   FONT=labelfont)
            zaxisID = WIDGET_LIST(zaxisBase, VALUE=ListNamesZ, $
                    UVALUE=labellist[2], $
                    FONT=listfont, $
                    XSIZE=xsize, YSIZE=ysize)
          endif else begin
            zaxisID = 0L
          endelse
          END
        1 : BEGIN
          xaxisBase = WIDGET_BASE(SelectAxesBase, GROUP_LEADER = parent, /ROW)
          xaxisID = WIDGET_LIST(xaxisBase, VALUE=ListNamesX, $
                  UVALUE=labellist[0], $
                  FONT=listfont, $
                  XSIZE=xsize, YSIZE=ysize)
          labelxBase = WIDGET_BASE(xaxisBase, /ALIGN_CENTER)
          labelx = WIDGET_LABEL(labelxBase, VALUE=labellist[0], $
                 FONT=labelfont)

          yaxisBase = WIDGET_BASE(SelectAxesBase, GROUP_LEADER = parent, /ROW)
          yaxisID = WIDGET_LIST(yaxisBase, VALUE=ListNamesY, $
                  UVALUE=labellist[1], $
                  FONT=listfont, $
                  XSIZE=xsize, YSIZE=ysize)
          labelyBase = WIDGET_BASE(yaxisBase, /ALIGN_CENTER)
          labely = WIDGET_LABEL(labelyBase, VALUE=labellist[1], $
                 FONT=labelfont)

          if(dimension eq 3)then begin
            zaxisBase = WIDGET_BASE(SelectAxesBase, $
                    GROUP_LEADER = parent, /COLUMN)
            zaxisID = WIDGET_LIST(zaxisBase, VALUE=ListNamesZ, $
                    UVALUE=labellist[2], $
                    FONT=listfont, $
                    XSIZE=xsize, YSIZE=ysize)
            labelzbase = WIDGET_BASE(zaxisBase, /ALIGN_CENTER)
            labelz = WIDGET_LABEL(labelzbase, VALUE=labellist[2], $
                   FONT=labelfont)
          endif else begin
            zaxisID = 0L
          endelse
          END
      ENDCASE
    endelse
```

```
  ; The state is dependent on the number of axes
  ; -------------------------------------------
  selectList=0
  CASE dimension OF
    2: begin
      state = { SelectAxesTLB: SelectAxesTLB, $
              selectList: selectList, $
              bugmsg: bugmsg, $
              dimension: dimension, $
              xaxisID: xaxisID, $
              yaxisID: yaxisID, $
              ListIndexX: LONARR(nListNamesX), $
              ListIndexY: LONARR(nListNamesY), $
              ListNamesX: ListNamesX, $
              ListNamesY: ListNamesY }
      end
    3: begin
      state = { SelectAxesTLB: SelectAxesTLB, $
              selectList: selectList, $
              bugmsg: bugmsg, $
              dimension: dimension, $
              xaxisID: xaxisID, $
              yaxisID: yaxisID, $
              zaxisID: zaxisID, $
              ListIndexX: LONARR(nListNamesX), $
              ListIndexY: LONARR(nListNamesY), $
              ListIndexZ: LONARR(nListNamesZ), $
              ListNamesX: ListNamesX, $
              ListNamesY: ListNamesY, $
              ListNamesZ: ListNamesZ }
      end
  ENDCASE

WIDGET_CONTROL, SelectAxesTLB, SET_UVALUE = uval, $
            EVENT_FUNC = 'CW_SelectAxes_event', $
            PRO_SET_VALUE = 'CW_SelectAxes_set_value', $
            FUNC_GET_VALUE = 'CW_SelectAxes_get_value'

WIDGET_CONTROL, WIDGET_INFO(SelectAxesTLB, /CHILD), SET_UVALUE=state,
/NO_COPY
RETURN, SelectAxesTLB

END
; ************************************************************ ******************
; ************************************************************ ******************
; **********                                        **********
; **********  U N I T   T E S T E R   F O R :: CW_SelectAxes      **********
; **********                                        **********
; ********************************************************* *****************
; ********************************************************* *****************
;+
; NAME: test_CW_SelectAxes_event
;
; PURPOSE:
;     This is just a simple event loop for the unit tester. It demonstrates
;     some of the features of this compund widget.
; -
; ==========================================================
==================
PRO test_CW_SelectAxes_event, event
   WIDGET_CONTROL, event.id, GET_UVALUE = uvalue

   AxesGroup = uvalue
   Print, '--------------------------------------'
   Print, 'Axis Group: ', AxesGroup
   Print, 'Axis: ', event.axis
   Print, 'Index: ', event.index
   items = *event.items
   Print, 'Selected Item: ', items[event.index]

END
; --------------------------------------------------------- -----------------
;+
; NAME: test_CW_SelectAxes
;
; PURPOSE:
;     This is the unit tester for the compund widget CW_SelectAxes. What this
;     does is show a simple example of how to use the widget and make sure that
;     it is working properly.
; USE:
;     1) Start IDL
;     2) Open this file, with all it's pieces!
;     3) Compile the file
;     4) Type test_CW_SelectAxes at the IDL prompt.
;
; MODULES REQUIRED:
;     CW_SelectAxes_get_value
;     CW_SelectAxes_set_value
;     CW_SelectAxes_event
;     CW_SelectAxes
;     These two modules test the CW_SelectAxes modules:
;     test_CW_SelectAxes_event
;     test_CW_SelectAxes
;
; AUTHOR:
;     Douglas J. Larson, Ph.D.
;      Space Radiation Lab
;      Mail Code: 220-47
;      California Institute of Technology
;      Pasadena, California  91125
;      Email: djl@srl.caltech.edu
;
; CATEGORY:
;      COMPOUND WIDGET UNIT TESTER
;
; CALLING SEQUENCE:
;      test_SelectAxes
; -
;  ============================================================
==================
PRO test_SelectAxes
sfont1 = '-adobe-helvetica-*-o-*-*-*-120-*-*-*-*-*-*'
sfont2 =  '-adobe-helvetica-bold-r-normal--24-240-75-75-p-138-iso8859- 1'
sfont3 =  '-adobe-helvetica-bold-r-normal--12-120-75-75-p-70-iso8859-1 '

slideTLB=WIDGET_BASE(/col)
ListSetID0=CW_SelectAxes(slideTLB)
ListSetID1=CW_SelectAxes(slideTLB, DIMENSION = 3, UVALUE=1)

timelist=['Time', 'livetime']
detectorlist=['m1', 'm2', 'm3', 'adcor', 'D1', 'D2', 'D3', 'D4']
ListSetID2=CW_SelectAxes(slideTLB, DEBUG=1, $
          TITLEAS = 'Real Data', TITLEPLACE=0, $
          LABELFONT=sfont1, LABELPLACE = 0, $
          XAXISLIST = timelist, $
          YAXISLIST = detectorlist, $
          LISTPLACE=0, $
          UVALUE=2)

spkids=['Kenny', 'Kyle', 'Cartman', 'Stan', 'Damien', 'Ike', 'Shelley']
spadults=['Chef', 'Matt Stone', 'Trey Parker', 'Mr. Garrison', 'Ned', $
        'Officer Barbrady', 'Mr. Mackey', 'Big Gay Al', 'Miss Crabtree']
spmisc=['Cows', 'Mr. Hankey', 'Mr. Hat', 'Terrance & Phillip']
ListSetID3=CW_SelectAxes(slideTLB, DEBUG=1, $
          TITLEAS = 'South Park Characters', TITLEPLACE=0, $
          LABELFONT=sfont1, LABELPLACE = 0, $
          LABELLIST = ['Kids', 'Adults', 'Oddities'], LISTPLACE=0, $
          DIMENSION = 3, $
          XAXISLIST = spkids, $
          YAXISLIST = spadults, $
          ZAXISLIST = spmisc, $
          UVALUE=3)

; Realize the widget:
```

```
WIDGET_CONTROL, slideTLB, /REALIZE

; Hand off control of the widget to the XMANAGER:
XMANAGER, "test_CW_SelectAxes", slideTLB, GROUP_LEADER=slideTLB, /NO_BLOCK

END
```

File Attachments

1) cw_selectaxes.pro, downloaded 105 times