

---

Subject: Source Code: A min/max compound widget  
Posted by [Doug Larson](#) on Wed, 23 Sep 1998 07:00:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hi All,

I just finished a compound widget using sliders for getting the minimum and maximum of some range. I have included a simple test code at the bottom of the file for people to see what the widget does. It works on my system (Solaris), but your mileage may vary. Multi-platform fixes are welcome! Also, there are bound to be bugs or bad style elements so feel free to exterminate the bugs and make suggestions!

Doug

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

```
. *****  
;  
;+  
; NAME:  
; CW_DoubleSlide  
;  
; PURPOSE:  
; A compound widget to select a range of values with two sliders. One slider  
; contols the minimum value of a range and the other the maximum. When the  
; Minimum Slide is moved, the new minimum value updates the allowed minimum  
; of the Maximum Slider. Conversely, when the Maximum Slider is moved, the  
; new maximum value becomes the top end of the Minimum Slide.  
; COMMENT: If RSI supplied a double slider this would not be needed! :)  
;  
; 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_DoubleSlide_get_value
; CW_DoubleSlide_set_value
; CW_DoubleSlide_event
; CW_DoubleSlide
; These two modules test the CW_DoubleSlide modules:
; test_CW_DoubleSlide_event
; test_CW_DoubleSlide
;
; CATEGORY:
; WIDGETS
;
; CALLING SEQUENCE:
; result = DoubleSlide(parent)
;
; INPUTS:
;   Parent:      The ID of the parent widget.
;
; OPTIONAL KEYWORD PARAMETERS:
; SLIDESRC: 0: Horizontal (Minimum Slide on left, Maximum on right)
;   1: Stacked (Maximum Slide on top, Minimum on bottom)
; LABELS: Default is ['MIN', 'MAX']
; LABELFONT: Font to use for the double slider labels
; LABELPLACE: 0 - Horizontal: MIN | slider | slider | MAX
;   1 - Horizontal: MIN | slider | MAX | slider
;   0 - Stacked: MIN slider | MAX slider
;   1 - Stacked: MIN | slider | MAX | slider
;   TITLED:      The title of slider. (The default is no title.)
; TITLEFONT: Font to use for the double slider title
; TITLERC: 0: ROW, title located centered to the left
;   1: COLUMN, title centered at the top
;   EDIT:        Set this keyword to make the slider label be
;                 editable. The default is EDIT=0.
;   FORMAT:       Provides the format in which the slider value is
;                 displayed. This should be a format as accepted by
;                 the STRING procedure. The default is FORMAT='(G13.6)'
;   FRAME:        Set this keyword to have a frame drawn around the
;                 widget. The default is FRAME=0.
;   MAXIMUM:      The maximum value of the slider. The default is
;                 MAXIMUM=100.
;   MINIMUM:      The minimum value of the slider. The default is
;                 MINIMUM=0.
;   SCROLL        Sets the SCROLL keyword to the WIDGET_SLIDER underlying
;                 this compound widget. Unlike WIDGET_SLIDER, the
;                 value given to SCROLL is taken in the floating units
;                 established by MAXIMUM and MINIMUM, and not in pixels.
;   SUPPRESS_VALUE: If true, the current slider value is not displayed.
;                 The default is SUPPRESS_VALUE=0.
;   UVALUE:       The user value for the widget.

```

```

; VALUE:      The initial value of the slider
; XSIZE:     For horizontal sliders, sets the length.
; YSIZE:     For vertical sliders, sets the height.
; DRAG:      Set this keyword to zero if events should only
;             be generated when the mouse is released. If it is
;             non-zero, events will be generated continuously
;             when the slider is adjusted. Note: On slow systems,
;             /DRAG performance can be inadequate. The default
;             is DRAG=0.
;
;
; 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.
; This is a floating point value.
;
;
; 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, 21 September 1998
; Pieces of CW_FSLIDER were used to help make parts of this code.
;
;
;-
; =====
=====
FUNCTION CW_DoubleSlide_get_value, id
; Recover the state of this compound widget
stash = WIDGET_INFO(id, /CHILD)
WIDGET_CONTROL, stash, GET_UVALUE=state, /NO_COPY

ret = [ state.minSlideValue, state.maxSlideValue ]

WIDGET_CONTROL, stash, SET_UVALUE=state, /NO_COPY
return, ret
end
; -----
FUNCTION CW_DoubleSlide_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.selectslider eq 0)then begin
  state.minSlideValue = value
endif

if(state.selectslider eq 0)then begin
  state.maxSlideValue = value
endif

  WIDGET_CONTROL, stash, SET_UVALUE=state, /NO_COPY
end
; -----
FUNCTION CW_DoubleSlide_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
WIDGET_CONTROL, event.id, GET_VALUE = slidervalue

if(state.bugmsg eq 1)then begin
  print,'----- begin DoubleSlide_event -----'
  help,state,/structure
  help,event,/structure
  print,'event.id=',event.id
  print,'-----'
endif

; See which widget was adjusted, the minimum or the maximum.
CASE event.ID OF
  state.minSlideID : BEGIN
    state.selectslider = 0
    if(state.bugmsg eq 1)then print,'Minimum=',slidervalue
    if(slidervalue le state.maxSlideValue)then begin
      state.minSlideValue = slidervalue
    endif else begin
      WIDGET_CONTROL, state.minSlideID, SET_VALUE=state.minSlideValue
    endelse
  END
  state.maxSlideID : BEGIN
    state.selectslider = 1
    if(state.bugmsg eq 1)then print,'Maximum=',slidervalue
    if(slidervalue ge state.minSlideValue)then begin
      state.maxSlideValue = slidervalue
    endif else begin
      WIDGET_CONTROL, state.maxSlideID, SET_VALUE=state.maxSlideValue
    endelse
  END
ENDCASE

```

```
ret = { DoubleSlide_event, ID:parent, TOP:event.top, HANDLER:0L, $
      MIN: state.minSlideValue, MAX: state.maxSlideValue}
WIDGET_CONTROL, stash, SET_UVALUE = state, /NO_COPY
```

```
RETURN, ret
```

```
END
```

```
; -----
FUNCTION CW_DoubleSlide, parent, DEBUG=bugmsg, SLIDESRC = slidesrc, $
      LABELS=labels, LABELFONT=labelfont, LABELPLACE=labelplace, $
      TITLEDs = titleds, TITLEFONT=titlefont, TITLERC=titlerc, $
      MAXIMUM = max, MINIMUM = min, SCROLL = scroll, SUPPRESS_VALUE = sup, $
      EDIT = edit, FRAME = frame, UVALUE = uval, VALUE = val, $
      XSIZE = xsize, YSIZE = ysize, FORMAT=format, DRAG = drag
```

```
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 NOT(KEYWORD_SET(slidesrc)) then slidesrc = 0
```

```
if NOT(KEYWORD_SET(labels)) then begin
```

```
  labels = STRARR(2)
```

```
  labels[0] = 'MIN'
```

```
  labels[1] = 'MAX'
```

```
endif
```

```
if NOT(KEYWORD_SET(labelfont)) then labelfont = $
```

```
  '-adobe-helvetica-bold-r-normal--12-120-75-75-p-70-iso8859-1 '
```

```
if NOT(KEYWORD_SET(labelplace)) then labelplace = 0
```

```
if N_ELEMENTS(titleds) EQ 0 then titleds = ""
```

```
if NOT(KEYWORD_SET(titlefont)) then titlefont = labelfont
```

```
if NOT(KEYWORD_SET(titlerc)) then titlerc = 0
```

```
if N_ELEMENTS(max) EQ 0 then max = 100.0
```

```
if N_ELEMENTS(min) EQ 0 then min = 0.0
```

```
if NOT(KEYWORD_SET(scroll)) then scroll = 10000 ELSE $
```

```
scroll = ABS(LONG((float(scroll) / (max - min)) * 1000000))
```

```
if NOT(KEYWORD_SET(sup)) then sup = 0
```

```
if NOT(KEYWORD_SET(edit)) then edit = 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 begin
```

```
  val = 0
```

```
  minSlideValue = min
```

```
  maxSlideValue = max
```

```
endif else begin
```

```
  minSlideValue = val
```

```
  maxSlideValue = val
```

```

endelse
if N_ELEMENTS(xsize) EQ 0 then xsize = 100
if N_ELEMENTS(ysize) EQ 0 then ysize = 100
if NOT KEYWORD_SET(format) then format='(G13.6)'
if N_ELEMENTS(drag) EQ 0 then drag = 0

if(titlerc eq 0)then begin ; ROW
  doubleslideTLB=WIDGET_BASE(parent, /ROW)
  titleBase = WIDGET_BASE(doubleslideTLB, /ALIGN_CENTER)
endif else begin ; COLUMN
  doubleslideTLB=WIDGET_BASE(parent, /COLUMN)
  titleBase = WIDGET_BASE(doubleslideTLB, /ALIGN_CENTER)
endelse
title = WIDGET_LABEL(titleBase, VALUE=titleds, FONT=titlefont)

selectslider=0
if(slidesrc eq 0)then begin
  doubleslideBase=WIDGET_BASE(doubleslideTLB, /ROW)
  minSlideBase = WIDGET_BASE(doubleslideBase, GROUP_LEADER = parent, /ROW)
  labelminbase = WIDGET_BASE(minSlideBase)
  labelmin = WIDGET_LABEL(labelminbase, VALUE=labels[0], FONT=labelfont)
  minSlideID = CW_FSLIDER(minSlideBase, /EDIT, $
    MINIMUM=min, MAXIMUM=max, VALUE=minSlideValue)
CASE labelplace OF
  0 : BEGIN
    maxSlideBase = WIDGET_BASE(doubleslideBase, GROUP_LEADER = parent, /ROW)
    maxSlideID = CW_FSLIDER(maxSlideBase, /EDIT, $
      MINIMUM=min, MAXIMUM=max, VALUE=maxSlideValue)
    labelmaxbase = WIDGET_BASE(maxSlideBase)
    labelmax = WIDGET_LABEL(labelmaxbase, VALUE=labels[1], FONT=labelfont)
  END
  1 : BEGIN
    maxSlideBase = WIDGET_BASE(doubleslideBase, GROUP_LEADER = parent, /ROW)
    labelmaxbase = WIDGET_BASE(maxSlideBase)
    labelmax = WIDGET_LABEL(labelmaxbase, VALUE=labels[1], FONT=labelfont)
    maxSlideID = CW_FSLIDER(maxSlideBase, /EDIT, $
      MINIMUM=min, MAXIMUM=max, VALUE=maxSlideValue)
  END
ENDCASE
endif else begin
  doubleslideBase=WIDGET_BASE(doubleslideTLB, /COLUMN)
  maxSlideBase = WIDGET_BASE(doubleslideBase, GROUP_LEADER = parent, /COLUMN)
  labelmaxbase = WIDGET_BASE(maxSlideBase)
  labelmax = WIDGET_LABEL(labelmaxbase, VALUE=labels[1], FONT=labelfont)
  maxSlideID = CW_FSLIDER(maxSlideBase, /EDIT, $
    MINIMUM=min, MAXIMUM=max, VALUE=maxSlideValue)
CASE labelplace OF
  0 : BEGIN

```

```

        minSlideBase = WIDGET_BASE(doubleslideBase, GROUP_LEADER = parent,
/COLUMN)
        minSlideID = CW_FSLIDER(minSlideBase, /EDIT, $
            MINIMUM=min, MAXIMUM=max, VALUE=minSlideValue)
        labelminbase = WIDGET_BASE(minSlideBase)
        labelmin = WIDGET_LABEL(labelminbase, VALUE=labels[0], FONT=labelfont)
        END
    1 : BEGIN
        minSlideBase = WIDGET_BASE(doubleslideBase, GROUP_LEADER = parent,
/COLUMN)
        labelminbase = WIDGET_BASE(minSlideBase)
        labelmin = WIDGET_LABEL(labelminbase, VALUE=labels[0], FONT=labelfont)
        minSlideID = CW_FSLIDER(minSlideBase, /EDIT, $
            MINIMUM=min, MAXIMUM=max, VALUE=minSlideValue)
        END
    ENDCASE
endelse

if(bugmsg eq 1)then begin
    print,'doubleslideTLB=',doubleslideTLB
    print,'minSlideBase=',minSlideBase,' maxSlideBase=',maxSlideBase
    print,'minSlideID=',minSlideID,' maxSlideID=',maxSlideID
endif

state = { doubleslideTLB: doubleslideTLB, $
    selectslider: selectslider, $
    bugmsg: bugmsg, $
    MINIMUM:min, $
    MAXIMUM:max, $
    minSlideID: minSlideID, $
    maxSlideID: maxSlideID, $
    minSlideValue: minSlideValue, $
    maxSlideValue: maxSlideValue}

WIDGET_CONTROL, doubleslideTLB, SET_UVALUE = uval, $
    EVENT_FUNC = 'CW_DoubleSlide_event', $
    PRO_SET_VALUE = 'CW_DoubleSlide_set_value', $
    FUNC_GET_VALUE = 'CW_DoubleSlide_get_value'

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

END
. *****
.
. *****
.
. *****
. *****
. ***** UNIT TESTER FOR :: CW_DoubleSlide *****
.

```

```

. *****
;
; *****
; *****
;
;+
; NAME: test_CW_DoubleSlide_event
;
;
; PURPOSE:
; This is just a simple event loop for the unit tester. It demonstrates
; some of the features of this compound widget.
; -
; =====
=====
PRO test_CW_DoubleSlide_event, event
  WIDGET_CONTROL, event.id, GET_UVALUE = uvalue
  WIDGET_CONTROL, event.id, GET_VALUE = value

CASE uvalue OF
  0 : BEGIN
    print,'Range 0: minSlideValue=',value[0]
    print,'      maxSlideValue=',value[1]
    END
  1 : BEGIN
    print,'Range 1: minSlideValue=',value[0]
    print,'      maxSlideValue=',value[1]
    END
  2 : BEGIN
    print,'Range 2: minSlideValue=',value[0]
    print,'      maxSlideValue=',value[1]
    END
  3 : BEGIN
    print,'Range 3: minSlideValue=',value[0]
    print,'      maxSlideValue=',value[1]
    END
  4 : BEGIN
    print,'Range 4: minSlideValue=',value[0]
    print,'      maxSlideValue=',value[1]
    END
ENDCASE

; WIDGET_CONTROL, event.top, SET_UVALUE= dBInfo

END
; -----
;+
; NAME: test_CW_DoubleSlide
;
;
; PURPOSE:
; This is the unit tester for the compound widget CW_DoubleSlide. 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_DoubleSlide at the IDL prompt.
;
; MODULES REQUIRED:
; CW_DoubleSlide_get_value
; CW_DoubleSlide_set_value
; CW_DoubleSlide_event
; CW_DoubleSlide
; These two modules test the CW_DoubleSlide modules:
; test_CW_DoubleSlide_event
; test_CW_DoubleSlide
;
; 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_DoubleSlide
; -
; =====
=====
PRO test_DoubleSlide
sfont1 = '-adobe-helvetica-*-*-*-*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)
rangID0=CW_DoubleSlide(slideTLB)
rangID1=CW_DoubleSlide(slideTLB, DEBUG=1, $
    TITLED5 = 'Pigs Currently Flying', TITLERC=0, $
    LABELS = ['Min', 'Max'], LABELFONT=sfont1, LABELPLACE = 0, $
    MINIMUM=0.0, MAXIMUM=1.e6, VALUE=3.0, $
    SLIDESRC = 0, $
    UVALUE=1)
rangID2=CW_DoubleSlide(slideTLB, DEBUG=1, $
    LABELS = ['Low', 'High'], LABELFONT=sfont2, LABELPLACE = 1, $

```

```
TITLED3 = 'Stock Market', TITLERC=1, $
MINIMUM=0.0, MAXIMUM=9000.0, VALUE=7983.00, $
SLIDESRC = 0, $
UVALUE=2)
rangeID3=CW_DoubleSlide(slideTLB, DEBUG=1, $
LABELS = ['Floor', 'Ceiling'], LABELFONT=sfont3, LABELPLACE = 0, $
TITLED3 = 'Allowed Range', TITLERC=1, $
MINIMUM=0.0, MAXIMUM=12.0, VALUE=6.0, $
SLIDESRC = 1, $
UVALUE=3)
rangeID4=CW_DoubleSlide(slideTLB, DEBUG=1, $
LABELS = ['Ground', 'Sky'], LABELFONT=sfont3, LABELPLACE = 1, $
TITLED4 = 'Plane Altitude', TITLERC=0, $
MINIMUM=0.0, MAXIMUM=25000.0, VALUE=10000.0, $
SLIDESRC = 1, $
UVALUE=4)

; Realize the widget:
WIDGET_CONTROL, slideTLB, /REALIZE

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

END
```

## File Attachments

---

1) [cw\\_doubleslide.pro](#), downloaded 98 times

---