Subject: Re: Table Widget
Posted by DMottershead on Fri, 23 Jan 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Following are the files I have been using. The first is a file with a '.cnv' extension full of ASCII column data. The function ra_check_file has been included because it didn't believe the data file was ASCII for some reason returning the value -4 to the main procedure. These values have therefore been adjusted. the same applies for function at_check_file. Next is the readgen procedure, followed by the normalize procedure. Then comes the procedures. Have fun playing with it. Oh, David, it's only 11:03 am at the moment and it is a great day. I might just have a few beers for you this evening while watching the sunset as suggested. If the guys at RSI are reading I would like to know how to change tack and re-design this code using OO methodology. Since I was taught this way I seem to be locked into producing code only this way.

* Sea-Bird SBE 19 Data File:
* FileName = \SEA_19\RAW_DATA\wl181208.HEX
* Software Version 4.206
* Temperature SN = 1670
* Conductivity SN = 1670
* System UpLoad Time = Dec 19 1997 11:31:40
* Ship:
* Cruise:
* Station:
* Latitude:
* Longitude:
* ds
* SEACAT PROFILER V3.0e SN 1670 12/19/97 11:29:41.198
* strain gauge pressure sensor: S/N = 165408, range = 100 psia, tc = 218
* clk = 32768.078 iop = 197 vmain = 8.0 vlith = 5.8
* mode = PROFILE ncasts = 11
* sample rate = 1 scan every 0.5 seconds
* minimum raw conductivity frequency for pump turn on = 2500 hertz
* pump delay = 45 seconds
* samples = 7549 free = 35824 lwait = 0 msec
* SW1 = 80 battery cutoff = 5.8 volts
* number of voltages sampled = 4
* logdata = NO
* S>
* cast 8 12/18 13:09:04 samples 5832 to 6444 sample rate = 1 scan every 0.5 seconds stop = switch off
# nquan = 14
# nvalues = 32
# units = metric
# name 0 = scan: scan number

name 1 = pr: pressure [db]
name 2 = c0mS/cm: conductivity [mS/cm]
name 3 = t090: temperature, ITS-90 [deg C]
name 4 = oxC: oxygen, current [ï¿½A]
name 5 = oxT: oxygen, temperature [deg C]
name 6 = ph: pH
name 7 = obs: backscatterance
name 8 = oxMg/L: oxygen [mg/l]
name 9 = oxPS: oxygen, percent saturation
name 10 = depS: depth, salt water [m]
name 11 = sal00: salinity, PSS-78 [PSU]
name 12 = density00: density [kg/m^3]
name 13 = flag: 0.000e+00
span 0 = 244, 557
span 1 = 0.100, 3.200
span 2 = 11.341755, 19.018812
span 3 = 26.6812, 27.5354
span 4 = 26.88405, 35.08348
span 5 = 26.81935, 27.43715
span 6 = 8.682, 8.891
span 7 = 3.79, 34.75
span 8 = 8.01139, 10.05818
span 9 = 104.75969, 134.45134
span 10 = 0.099, 3.182
span 11 = 6.1043, 10.7384
span 12 = 1000.9350, 1004.4758
span 13 = 0.000e+00, 0.000e+00
interval = decibars: 0.1
start_time = Dec 18 1998 13:09:04
bad_flag = -9.990e-29
serial_numbers = t0:1670, c0:1670, ox:230462, ph:180179, obs:570,
pr:165408
datcnv_date = Jan 19 1998 13:07:57, 4.206
datcnv_in = WL181208.HEX 1670_O62.CON 1670 1670 165408
datcnv_skipover = 244
filter_date = Jan 19 1998 13:08:40, 4.206
filter_in = WL181208.CNV
filter_low_pass_tc_A = 0.500
filter_low_pass_tc_B = 0.000
filter_low_pass_A_vars = c0mS/cm
filter_low_pass_B_vars =
alignctd_date = Jan 19 1998 13:09:11, 4.206
alignctd_in = WL181208.CNV
alignctd_cond_adv_secs = 0.500, 0.000
alignctd_temp_adv_secs = 0.500, 0.000
alignctd_oxygen_adv_secs = 2.000, 0.000
alignctd_uservar_index_adv_secs= -1 0.000
derive_date = Jan 19 1998 13:11:39, 4.206

derive_in = WL181208.CNV 1670_O62.CON
derive_time_window_docdt = seconds: 2
binavg_date = Jan 19 1998 13:14:15, 4.206
binavg_in = WL181208.CNV
binavg_bintype = Pressure Bins
binavg_binsize = 0.10
binavg_excl_bad_scans = yes
binavg_downcast_only = yes
binavg_skipover = 0
binavg_surface_bin = yes, min = 0.100, max = 0.050, value = 0.000
derive_date = Jan 19 1998 13:19:44, 4.206
derive_in = WL181208.CNV 1670_O62.CON
file_type = ascii
END
244 0.100 11.341755 27.5354 28.75453 27.43715
8.891 3.91 8.37270 110.05114 0.099 6.1043 1000.9350
0.000e+00
247 0.200 11.352964 27.4969 28.62960 27.43715
8.891 3.91 8.31133 109.17754 0.199 6.1158 1000.9551
0.000e+00
250 0.300 11.355061 27.4864 28.51698 27.42610
8.891 3.91 8.28345 108.79274 0.298 6.1183 1000.9604
0.000e+00
252 0.400 11.355519 27.4834 28.49509 27.42905
8.891 3.91 8.19743 107.65771 0.398 6.1190 1000.9622
0.000e+00
259 0.500 11.352523 27.4843 28.12955 27.42277
8.891 3.91 8.19624 107.64265 0.497 6.1171 1000.9610
0.000e+00
270 0.600 11.353065 27.4994 28.48696 27.41876
8.891 3.91 8.29769 109.00318 0.597 6.1155 1000.9559
0.000e+00
281 0.700 11.355480 27.4956 28.36474 27.41317
8.891 3.91 8.25382 108.42057 0.696 6.1174 1000.9588
0.000e+00
292 0.800 11.353315 27.4969 28.30524 27.41272
8.891 3.91 8.23966 108.23624 0.796 6.1159 1000.9578
0.000e+00
301 0.900 11.360023 27.4994 28.27618 27.41286
8.891 3.95 8.23303 108.15605 0.895 6.1195 1000.9602
0.000e+00
315 1.000 11.363832 27.4897 28.25508 27.40417
8.889 3.91 8.22953 108.09360 0.995 6.1230 1000.9660
0.000e+00
330 1.100 11.375402 27.4902 28.21122 27.40289
8.890 3.91 8.21651 107.92766 1.094 6.1296 1000.9713
0.000e+00
346 1.200 11.383255 27.4825 28.19589 27.38999

8.886 3.91 8.21816 107.93795 1.193 6.1352 1000.9781
0.000e+00
358 1.300 11.403282 27.4696 28.15482 27.38223
8.884 3.91 8.20678 107.77198 1.293 6.1485 1000.9922
0.000e+00
367 1.400 11.402014 27.4687 28.14755 27.37433
8.882 3.86 8.21137 107.83017 1.392 6.1478 1000.9924
0.000e+00
377 1.500 11.440990 27.4618 28.12294 27.37147
8.881 3.91 8.20192 107.70720 1.492 6.1714 1001.0124
0.000e+00
386 1.600 11.445674 27.4511 28.15318 27.36009
8.876 3.91 8.22350 107.97268 1.591 6.1755 1001.0190
0.000e+00
393 1.700 11.454218 27.4483 28.13526 27.34986
8.874 3.91 8.21002 107.79354 1.691 6.1808 1001.0242
0.000e+00
402 1.800 11.476840 27.4324 28.12158 27.34658
8.872 3.91 8.20934 107.76357 1.790 6.1960 1001.0406
0.000e+00
415 1.900 11.497839 27.4074 28.04540 27.33311
8.866 3.85 8.19730 107.56746 1.890 6.2115 1001.0597
0.000e+00
427 2.000 11.515682 27.3773 28.03333 27.30655
8.862 3.79 8.20778 107.65623 1.989 6.2258 1001.0795
0.000e+00
438 2.100 11.540717 27.2866 27.95642 27.27967
8.864 3.87 8.19891 107.38373 2.088 6.2523 1001.1257
0.000e+00
449 2.200 11.575185 27.1802 27.86453 27.23247
8.860 3.91 8.19273 107.12166 2.188 6.2865 1001.1820
0.000e+00
461 2.300 11.695685 26.8873 27.57518 27.12353
8.849 3.91 8.15635 106.15904 2.287 6.3965 1001.3477
0.000e+00
471 2.400 11.824458 26.6812 27.18611 26.97180
8.833 3.91 8.10355 105.14733 2.387 6.5009 1001.4841
0.000e+00
481 2.500 12.416226 26.7270 27.05950 26.84815
8.804 3.91 8.11289 105.55880 2.486 6.8460 1001.7295
0.000e+00
489 2.600 12.981917 26.9005 26.94231 26.81935
8.733 3.91 8.01268 104.75969 2.586 7.1571 1001.9133
0.000e+00
498 2.700 13.292703 26.9631 26.88405 26.84705
8.682 3.91 8.01139 104.96290 2.685 7.3327 1002.0271
0.000e+00
506 2.800 13.386587 26.9829 27.09122 26.88834

```
8.687      3.91    8.07047  105.80539     2.785     7.3856  1002.0614
0.000e+00
      515     2.900  14.944644    27.0108   27.70010   26.91687
8.689      3.91    8.51524  112.30018     2.884     8.3151  1002.7477
0.000e+00
      523     3.000  18.318452    27.1509   31.90018   26.95435
8.772      3.91   10.05818  134.45134     2.983    10.3396  1004.2185
0.000e+00
      543     3.100  19.018813    27.2907   35.08348   27.09717
8.844      6.68    9.76952  131.13786     3.083    10.7384  1004.4758
0.000e+00

;
 ------------------------------------------------------------ -----------------
; Doesn't Believe That The Previous File Is ASCII Data for some reason
;
; Purpose: Check that the input filename is a string, exists, and
appears
;          to be ASCII... also if the file is just columned ascii data,
;          then guess at the number of default columns of data.

function ra_check_file, fname, default_num_columns=default_num_columns
  catch, error_status
  if (error_status ne 0) then begin
    if (n_elements(unit) gt 0) then free_lun, unit
    return, -3 ; unexpected error reading from file
  endif
  ;
  info = size(fname)
  if (info(info(0)+1) ne 7) then return, -1 ; filename isn't a string
  ;
  openr, unit, fname, error=error, /get_lun
  if (error eq 0) then begin
    finfo = fstat(unit)
    ; set non-ascii values in lookup table
    ;
    lut = bytarr(256) + 1b
    lut[7:13]   = 0b
    lut[32:127] = 0b
    data = bytarr(32767<finfo.size, /nozero)
    readu, unit, data
    carriage_return = (total(data eq 10b) gt 0 or total(data eq 13b) gt
0)
    if (carriage_return eq 0) then begin
    ; looks like a binary file
    ;
      free_lun, unit
      return, 0
```

```
    endif
    non_printable   = (total(lut(data)) gt 0)
    if (non_printable) then begin
    ; looks like a binary file
    ;
      free_lun, unit
      return, 0
    endif
    ; everything looks ok, now guess at the number of columns...
    ;
    point_lun, unit, 0
    line = ''
    readf, unit, line
    free_lun, unit
    bline = byte(strtrim(strcompress(line),2))
    ptr = where(bline eq 32, num_spaces)
    default_num_columns = num_spaces + 1
  endif else $
    return, -2 ; unable to open file
end

;
 ------------------------------------------------------------ -----------------
;
;  Purpose: Check that the input filename is a string, exists, and
appears
;           to be ASCII...
;
function at_check_file, fname
  catch, error_status
  if (error_status ne 0) then begin
    if (n_elements(unit) gt 0) then free_lun, unit
    return, -3 ; unexpected error reading from file
  endif
  ;
  info = size(fname)
  if (info(info(0)+1) ne 7) then return, -1 ; filename isn't a string
  ;
  openr, unit, fname, error=error, /get_lun
  if (error eq 0) then begin
    finfo = fstat(unit)
    ; set non-ascii values in lookup table
    lut = bytarr(256) + 1b
    lut[7:13]   = 0b
    lut[32:127] = 0b
    data = bytarr(32767<finfo.size, /nozero)
    readu, unit, data
    free_lun, unit
```

```
   carriage_return = (total(data eq 10b) gt 0 or total(data eq 13b) gt
0)
   if (carriage_return eq 0) then return, 0 ; looks like a binary file
   non_printable   = (total(lut(data)) gt 0)
   if (non_printable) then return, 0 $ ; looks like a binary file
   else                    return, 0    ; everything is cool
  endif else $
   return, -2 ; unable to open file
end

function readgen

filename = dialog_pickfile(/read,/noconfirm, path='d:\idl\ctd', filter =
'*.cnv')
if filename eq '' then return,0

mytemplate = ASCII_Template(filename, browse_lines = 150)

result = READ_ASCII(filename, template = mytemplate, header = hdr)

return, result
end


FUNCTION Normalize, range, Position=position

   ; This is a utility routine to calculate the scaling vector
   ; required to position a vector of specified range at a
   ; specific position given in normalized coordinates. The
   ; scaling vector is given as a two-element array like this:
   ;
   ;   scalingVector = [translationFactor, scalingFactor]
   ;
   ; The scaling vector should be used with the [XYZ]COORD_CONV
   ; keywords of a graphics object or model. For example, if you
   ; wanted to scale an X axis into the data range of -0.5 to 0.5,
   ; you might type something like this:
   ;
   ;   xAxis->GetProperty, Range=xRange
   ;   xScale = Normalize(xRange, Position=[-0.5, 0.5])
   ;   xAxis, XCoord_Conv=xScale

IF (N_Elements(position) EQ 0) THEN position = [0.0, 1.0] ELSE $
   position=Float(position)
range = Float(range)

scale = [((position[0]*range[1])-(position[1]*range[0])) / $
   (range[1]-range[0]), (position[1]-position[0])/(range[1]-range[0])]
```

```
RETURN, scale
END

pro CTD_Table_Widget_Events, event

    ; This event handler handles draw widget expose events.

WIDGET_CONTROL, event.top, GET_UVALUE = ptr
;WIDGET_CONTROL, event.id, GET_UVALUE = uval

    ; Draw the graphic.


result = widget_info(event.id,/table_select)
result2 = widget_info(event.id,/table_edit_cell)
;help, result
;print, 'normal' ,result
;help, result2
;print, 'normal2' ,result2


;if result2 ne [-1,-1] then begin
;a = result2
;result = widget_info(event.id,/table_select)
;endif

a=result(0)
b=result(1)
c=result(2)
d=result(3)

if abs(d - b) lt 0.0001 then return

data = (*ptr).plotdata(a:c, b:d)

; set the plot data

(*ptr).plot.thisPlot->SetProperty, datay = data

(*ptr).plot.thisPlot->GetProperty, XRange=xrange, YRange=yrange

; Set up the scaling so that the axes for the plot and the
; plot data extends from 0->1 in the X and Y directions.

xs = Normalize(xrange)
ys = Normalize(yrange)
```

```
; Scale the plot data into 0->1.

;print, 'xs[0] = ',xs[0], 'ys[0] = ', ys[0]
;xmajor = (xrange[1]-xrange[0])/5
;ymajor = (yrange[1]-yrange[0])/5
;yrange[0] = 0

(*ptr).plot.thisPlot->SetProperty, XCoord_Conv=xs, YCoord_Conv=ys
(*ptr).plot.xAxis1->SetProperty, Ticklen = 0.01
;(*ptr).plot.xAxis1->SetProperty, major = xmajor
(*ptr).plot.xAxis1->SetProperty, minor = 4
(*ptr).plot.xAxis1->SetProperty, range = xrange
(*ptr).plot.xAxis1->SetProperty, XCoord_Conv = xs
(*ptr).plot.xAxis1->SetProperty, tickdir = 1
(*ptr).plot.xAxis1->SetProperty, Location = [0,0]
;(*ptr).plot.xAxis1->SetProperty, Location = [xs[0],ys[0]]
(*ptr).plot.yAxis1->SetProperty, Ticklen = 0.01
;(*ptr).plot.xAxis1->SetProperty, major = ymajor
(*ptr).plot.yAxis1->SetProperty, minor = 4
(*ptr).plot.yAxis1->SetProperty, range = yrange
(*ptr).plot.yAxis1->SetProperty, YCoord_Conv = ys
(*ptr).plot.yAxis1->SetProperty, tickdir = 1
;(*ptr).plot.yAxis1->SetProperty, Location = [xs[0],ys[0]]
(*ptr).plot.yAxis1->SetProperty, Location = [0,0]

;(*ptr).plot.xAxis1->GetProperty, Ticktext = xAxisText
;(*ptr).plot.yAxis1->GetProperty, Ticktext = yAxisText

xaxistext = Obj_New('IDLgrText', font = (*ptr).plot.helvetica6pt)
yaxistext = Obj_New('IDLgrText', font = (*ptr).plot.helvetica6pt)

xAxisText->SetProperty, Font=(*ptr).plot.helvetica6pt
yAxisText->SetProperty, Font=(*ptr).plot.helvetica6pt

(*ptr).obj.omodel->add, xaxistext
(*ptr).obj.omodel->add, yaxistext

; render the graphics to the window

(*ptr).obj.owindow -> Draw, (*ptr).obj.oview
;endif else begin
;help, result2
;a = result2
;print, a
;return
;endelse

    ;Put the info structure back.
```

```
;Widget_Control, event.top, Set_UValue=ptr, /No_Copy

;case uval of
;'insertrows': begin

;result = widget_info(event.id,/table_select)
;print, 'ir', result
;result = widget_info(event.id,/table_select)
;widget_control, insert_rows

;return
;end
;'insertcols': begin

;result = widget_info(event.id,/table_select)
;print, 'ic', result
;result = widget_info(event.id,/table_select)
;widget_control, insert_rows

return
;end
;endcase

END

pro ctdplt_event, event


WIDGET_CONTROL, event.top, GET_UVALUE = ptr
WIDGET_CONTROL, event.id, GET_UVALUE = uval

case uval of
'exit':begin
  ptr_free, ptr
       WIDGET_CONTROL, event.top, /DESTROY
       return
       end

'draw':begin
       WIDGET_CONTROL, event.top, /realize
       return
       end


;case uval of
'insertrows': begin
```

```
result = CTD_Table_Widget_Events(event)
;print, 'ir', result
;result = widget_info(event.id,/table_select)
;widget_control, insert_rows

return
end

'insertcols': begin

result = widget_info(event.id,/table_select)
;print, 'ic', result
;result = widget_info(event.id,/table_select)
;widget_control, insert_rows

return
end

'IDLhelp': begin
        online_help
        end

'printsetup': begin
     result = Dialog_PrinterSetup((*ptr).thisPrinter)
 end

'print': begin
        dstatus = DIALOG_MESSAGE(/QUESTION, $
          'Printing Can Take a long time on a windows printer.
Continue??')
        if (strupcase(dstatus) eq 'YES') then begin
        result = Dialog_PrintJob((*ptr).thisPrinter)
        IF result EQ 1 THEN BEGIN
           (*ptr).thisPrinter->Draw, (*ptr).obj.oview
           (*ptr).thisPrinter->NewDocument
        ENDIF
       endif
        WIDGET_CONTROL, event.top, /realize
end

endcase
return
end


pro ctdplt

data = readgen()
```

```
help, data, /structures,output = out

b = n_tags(data)
c = n_elements(data.field01)

;if b ge 10 then tags = 2 else tags = 1

plotdata = dblarr(b,c)
i = 0
if i lt b then plotdata(i,*) = data.field01
i = i + 1
if i lt b then plotdata(i,*) = data.field02
i = i + 1
if i lt b then plotdata(i,*) = data.field03
i = i + 1
if i lt b then plotdata(i,*) = data.field04
i = i + 1
if i lt b then plotdata(i,*) = data.field05
i = i + 1
if i lt b then plotdata(i,*) = data.field06
i = i + 1
if i lt b then plotdata(i,*) = data.field07
i = i + 1
if i lt b then plotdata(i,*) = data.field08
i = i + 1
if i lt b then plotdata(i,*) = data.field09
i = i + 1
if i lt b then plotdata(i,*) = data.field10
i = i + 1
if i lt b then plotdata(i,*) = data.field11
i = i + 1
if i lt b then plotdata(i,*) = data.field12
i = i + 1
if i lt b then plotdata(i,*) = data.field13
i = i + 1
if i lt b then plotdata(i,*) = data.field14
i = i + 1
if i lt b then plotdata(i,*) = data.field15
i = i + 1
if i lt b then plotdata(i,*) = data.field16
i = i + 1

tlb = Widget_Base(TITLE = 'CTD', $
        /COLUMN, $
        MBAR = menubar)

fileRow = WIDGET_BASE(tlb, $
```

```
    /ROW, $
    SPACE = 20)

fileTable = WIDGET_TABLE(tlb, $
 /frame, $
; xsize = b, $
;   ysize = 5, $
 scr_Xsize = 640, $
 scr_ysize = 190, $
 alignment = 0, $
; x_scroll_size = 7, $
; y_scroll_size = c, $
 /scroll, $
 /editable, $
 /resizeable_columns, $
 /resizeable_rows, $
 /all_events, $
   VALUE = plotdata, $
    Event_Pro='CTD_Table_Widget_Events', $
   UVALUE = 'tablebut')

; Application Menu Bar

fileMenu = WIDGET_BUTTON(menubar, $
   VALUE = 'File', $
   /MENU)
;fileBut = WIDGET_BUTTON(fileMenu, $
;    VALUE = 'Open', $
;    UVALUE = 'open')
printMenBut = WIDGET_BUTTON(fileMenu, $
   VALUE = 'Print', $
   UVALUE = 'print')
printsetupMenBut = WIDGET_BUTTON(fileMenu, $
   VALUE = 'Print Setup..', $
   UVALUE = 'printsetup')
exitMenBut = WIDGET_BUTTON(fileMenu, $
   VALUE = 'Exit', $
   UVALUE = 'exit', $
   /SEPARATOR)

tableMenu = WIDGET_BUTTON(menubar, $
   VALUE = 'Table', $
   /MENU)
insertrowBut = WIDGET_BUTTON(tableMenu, $
    VALUE = 'Insert Rows', $
;    Event_Pro='CTD_Table_Widget_Events', $
    UVALUE = 'insertrows')
insertcolBut = WIDGET_BUTTON(tableMenu, $
```

```
    VALUE = 'Insert Columns', $
;    Event_Pro='CTD_Table_Widget_Events', $
    UVALUE = 'insertcols')
;settingsBut = WIDGET_BUTTON(plotMenu, $
;    VALUE = 'Change Titles', $
;    UVALUE = 'settings')
;ptrbutton = WIDGET_BUTTON(plotmenu, $
; value = 'PTR Button', $
; UVALUE = 'ptrbut')

helpMenu = WIDGET_BUTTON(menubar, $
   VALUE = 'Help', $
   /MENU, $
   /HELP)
idlhelpBut = WIDGET_BUTTON(helpMenu, $
   VALUE = 'Help on IDL', $
   UVALUE = 'IDLhelp')

WIDGET_CONTROL, tlb, /realize


; find the screen size

device, get_screen_size = scr
if (n_elements(xdim) eq 0) then xdim = fix(scr[0] * 0.8)
if (n_elements(ydim) eq 0) then ydim = fix(scr[1] * 0.5)

; create the draw area that will contain the
; object graphics

objectDraw = WIDGET_DRAW(tlb, $
  XSIZE = xdim, $
  YSIZE = ydim, $
  GRAPHICS_LEVEL = 2, $
  UVALUE = 'draw', $
  ;Event_Pro ='CTD_Resize_Events', $
  /EXPOSE_EVENTS)

; realize the control heirarchy

WIDGET_CONTROL, tlb, /REALIZE

; retrieve the object window ID from the drea area ID

WIDGET_CONTROL, objectDraw, GET_VALUE = oWindow

; add the view to the window
; create the printer object
```

```
thisPrinter = obj_new('IDLgrPrinter')

; create a view

oview = obj_new('IDLgrView', $
 color = [0,0,0], $
 viewplane_rect = [-0.2,-0.2,1.4,1.4], $
 location = [0,0], $
 units = 3)

; create a model

omodel = obj_new('IDLgrModel')

; add the model to the view

oview -> add, omodel

; set up default symbol

psym = Obj_New('IDLgrSymbol', Color=[255,0,0], Size=0.05)

; default fonts

helvetica12pt = Obj_New('IDLgrFont', 'Helvetica', Size=12)
helvetica10pt = Obj_New('IDLgrFont', 'Helvetica', Size=10)
helvetica8pt = Obj_New('IDLgrFont', 'Helvetica', Size=8)
helvetica6pt = Obj_New('IDLgrFont', 'Helvetica', Size=6)

; default title strings
; create default plot title

; create x and y axis titles

thisPlot = Obj_New('IDLgrPlot', Color=[0,255,0])

; create the x and y axes

axiscolor = [255,255,0]

xAxis1 = Obj_New('IDLgrAxis', 0, Color=axiscolor, tickdir = 1, ticklen =
0.01, $
 minor = 4, /exact)

yAxis1 = Obj_New('IDLgrAxis', 1, Color=axiscolor, tickdir = 1, ticklen =
0.01, $
 minor = 4, /exact)
```

```
oModel->add, yAxis1
oModel->add, xAxis1
oModel->Add, thisPlot

; create the information structure

plot = {thisplot:thisplot, $
 helvetica6pt:helvetica6pt, $
 ;xaxistext:xaxistext, $
 ;yaxistext:yaxistext, $
 xaxis1:xaxis1, $
 yaxis1:yaxis1}

; objects structure

obj = {oWindow:oWindow, $
       oview:oview, $
       omodel:omodel}

info = {obj:obj, $
 data:data, $
 plotdata:plotdata, $
 plot:plot}

; pointer to info structures

ptr = ptr_new(info, /no_copy)

; put the information structure into the UVALUE of
; the top-level base

WIDGET_CONTROL, tlb, SET_UVALUE = ptr

; call Xmanager to start up the main event loop

Xmanager, 'ctdplt', tlb;,/no_block

Return
end
--
Regards

David
```


 ** *******
David Mottershead   Phone: +61 2 9949 0234
Manly Hydraulics Laboratory   Fax: +61 2 9948 6185

110b King St, Manly Vale, 2093 email: dmottershead@mhl.nsw.gov.au
SYDNEY, AUSTRALIA WWW: http://www.mhl.nsw.gov.au
********** *******