
Subject: universal structure editor progress
Posted by [Dyer Lytle](#) on Wed, 28 Apr 1999 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

In the "Global Variables in IDL" thread, David Fanning wrote:

```
> Totally unnecessary. I've found that just putting the  
> program in the public domain means that it will come back  
> to you in short order written *exactly* the way you  
> *thought* you were writing it in the first place. :-)
```

Wow, well, let us test this hypothesis!!!

A while ago, I asked this newsgroup if anyone had a universal structure editor (USE) but didn't get anything. I've started building one but don't have time to work on it for the next few months. I'll attach what I have so far to this post and see what happens! Currently, all this is is an attempt at a Universal Structure Display (USD) and it even fails to that properly in many situations. However, it DOES do enough for me to look at stuff in my big nested structures, which is what I wanted to do.

--

-Dyer Lytle in Tucson
Cassini ISS Software Engineer
dyer@lpl.arizona.edu

pro Spreadsheet,array,xnum,ynum,typ

```
; Pop up a spreadsheet widget.
```

```
; Set up the arrays of strings for labeling the X and Y cell numbers.
```

```
for i = 0,xnum-1 do begin
```

```
  if i eq 0 then xstrings = string(format=(i1)',i) $
```

```
  else xstrings = [xstrings,string(format=(i1)',i)]
```

```
endfor
```

```
for i = ynum-1,0,-1 do begin
```

```
  if i eq ynum-1 then ystrings = string(format=(i1)',i) $
```

```
  else ystrings = [ystrings,string(format=(i1)',i)]
```

```
endfor
```

```
print,'xstrings ',xstrings
```

```
print,'ystrings ',ystrings
```

```
; Get the data pixel values, reverse them in Y because the spreadsheet
```

```
; has the origin in the upper left.
```

```
data = reverse(array,2)
```

```

; Make the widget.
spread = widget_base (/column)

sstable = widget_table(spread,value=data,row_labels=ystrings, $
  column_labels=xstrings,column_widths=80,units=0)
; sstable = widget_table(spread,value=data)
buttonbase = widget_base(spread,/row)
donebutton = widget_button (buttonbase, value='Done',Event_Pro='Generic_Exit')

Widget_Control, spread, /Realize
XManager, 'spreadsheet', spread, /No_Block
end

```

```

pro big_cursor, ix, iy, ix0, iy0

```

```

device, get_graphics_function=g_fnc ;save graphics function
device, set_graphics_function=10 ;Use XOR writing mode
if (ix0 GT -1) then begin ;Erase old mark
  plots, [0, ix0-3], [iy0,iy0],/dev
  plots, [ix0-1, ix0+1], [iy0,iy0],/dev
  plots, [ix0+3, !d.x_size-1], [iy0,iy0],/dev
  plots,[ix0,ix0],[0, iy0-3],/dev
  plots, [ix0, ix0], [iy0-1,iy0+1],/dev
  plots,[ix0,ix0],[iy0+3, !d.y_size-1],/dev
endif
plots, [0, ix-3], [iy,iy],/dev
plots, [ix-1, ix+1], [iy,iy],/dev
plots, [ix+3, !d.x_size-1], [iy,iy],/dev
plots,[ix,ix],[0, iy-3],/dev
plots, [ix, ix], [iy-1,iy+1],/dev
plots,[ix,ix],[iy+3, !d.y_size-1],/dev

device, set_graphics_function=g_fnc
end

```

```

function imscale, im, ns

```

```

x = size(im)
if x[0] lt 1 then return,[0.0,0.0]
m = moment(im,sdev=sdev)
me = median(im)
g = where(abs(im-me) lt (ns * sdev),n)

if n gt 0 then return,[min(im(g)),max(im(g))] else return,[0.0,0.0]

```

end

pro ArEdit_Descend,event

Widget_Control, event.top, Get_UValue=info

; Check to be sure that this entry is a valid pointer, or structure, or
; and array, return if not.

t = execute('s = size((*info.array)['+string(info.p)+''])')

if s(s(0)+1) ne 10 and s(s(0)+1) ne 8 and s(0) eq 0 then begin

 t = dialog_message('Sorry, tag is not a pointer or structure or array.')

 return

end

if s(s(0)+1) eq 10 then begin

 t = execute('test = ptr_valid((*info.array)['+string(info.p)+''])')

 if test[0] eq 0 then begin

 t = dialog_message('Sorry, tag refers to an invalid pointer.')

 return

 endif

endif

if s(0) gt 0 and s(s(0)+2) gt 1 then begin

 t=execute('arrayedit,(*info.struct).'+names(info.ypos))

endif else if s(s(0)+1) eq 10 then begin

 t = execute('ss = size((*info.array)['+string(info.p)+''])')

 if ss(ss(0)+1) eq 8 then begin

 t=execute('structedit,(*info.array)['+string(info.p)+''])')

 endif else if ss(0) gt 0 then begin

 t=execute('arrayedit,(*info.array)['+string(info.p)+''])')

 endif else begin

 t=execute('varedit,(*info.array)['+string(info.p)+''])')

 endelse

endif else if s(s(0)+1) eq 8 then begin

 t=execute('structedit,(*info.array)['+string(info.p)+''])')

endif else begin

 t=execute('arrayedit,(*info.array)['+string(info.p)+''])')

endelse

Widget_Control, event.top, Set_UValue=info

end

pro ArEdit_Slide,event

vtyp2=['undefined','byte','integer','long_int','float',\$
'double','complex','string','structure','dbl_cmplx',\$
'pointer','object']

```

Widget_Control, event.top, Get_UValue=info
Widget_Control,info.dataSlide,Get_Value=p
info.p = p
info.pval = (*info.array)[p]

if info.ttyp eq 10 then begin
    ; If pointer
    t = execute('ss = size>(*info.array)['+string(p)+']')
    vs = string(p)+' '+'pointer to '+vtyp2[ss(ss(0)+1)]
endif else if info.ttyp eq 8 then begin
    vs = string(p)+' '+'structure'
endif else if info.ttyp eq 11 then begin
    vs = string(p)+' '+'object'
endif else begin
    vs = string(p)+' '+'string((*info.array)[p])
endelse
Widget_Control,info.dataView,Set_Value=vs
Widget_Control,event.top, Set_UValue=info
end

```

```

pro ArEd_SlideXY, event
vtyp2=['undefined','byte','integer','long_int','float',$
'double','complex','string','structure','dbl_cmplx',$
'pointer','object']

```

```

Widget_Control, event.top, Get_UValue=info

```

```

if event.id eq info.XSlide then begin
    Widget_Control,info.XSlide,Get_Value=x
    s = info.scale
    oldx = info.x
    info.x = x
    Widget_Control,info.PixelView,Set_Value=string((*info.array) [info.x,info.y])
    info.aval = string((*info.array)[info.x,info.y])
    big_cursor, s*info.x, s*info.y, s*oldx, s*info.y
endif else if event.id eq info.YSlide then begin
    Widget_Control,info.YSlide,Get_Value=y
    s = info.scale
    oldy = info.y
    info.y = y
    Widget_Control,info.PixelView,Set_Value=string((*info.array) [info.x,info.y])
    info.aval = string((*info.array)[info.x,info.y])
    big_cursor, s*info.x, s*info.y, s*info.x, s*oldy
endif else if event.id eq info.Z1Text or event.id eq info.Z2Text then begin
    Widget_Control, info.Z1Text, Get_Value = z1
    Widget_Control, info.Z2Text, Get_Value = z2
    s = size(*info.array)

```

```

xnum = s(1)
ynum = s(2)
z1 = float(z1[0]) & z2 = float(z2[0])
xx = !D.WINDOW
Widget_Control, info.dataView, Get_Value = drawid1
wset, drawid1
tv, bytscl(congrid((*info.array), fix(xnum*info.scale), $
fix(ynum*info.scale), $
cubic=-.5), top=!d.n_colors-1, min=z1, max=z2)
wset, xx
big_cursor, info.scale*info.x, info.scale*info.y, -1, -1
endif

```

```

Widget_Control, event.top, Set_UValue=info
end

```

```

pro ArEd_multidim, event
Widget_Control, event.top, Get_UValue=info
if event.id eq info.dimSlide then begin
Widget_Control, info.dimSlide, Get_Value=thisdim
s = size(*info.array)
Widget_Control, info.pickSlide, Set_Slider_Max=s(thisdim+1)
info.crunchdim = thisdim
endif else if event.id eq info.pickSlide then begin
Widget_Control, info.pickSlide, Get_Value=thisplane
info.thisplane = thisplane
endif
Widget_Control, event.top, Set_UValue=info
end

```

```

pro ArEdDim_Descend, event
Widget_Control, event.top, Get_UValue=info
s = size(*info.array)
doit = 'reform((*info.array)[
for i = 0, s(0)-1 do begin
if i eq s(0)-1 then begin
if i eq info.crunchdim then begin
doit = doit+string(info.thisplane)
endif else begin
doit = doit+'*'
endif else begin
if i eq info.crunchdim then begin
doit = doit+string(info.thisplane)+'',
endif else begin
doit = doit+'*',

```

```

    endelse
  endelse
endfor
doit = doit+']]'
t = execute('arrayEdit,'+doit)
Widget_Control,event.top, Set_UValue=info
end

```

```

pro arrayedit,array

```

```

vtyp2=['undefined','byte','integer','long_int','float',$
'double','complex','string','structure','dbl_cmplx',$
'pointer','object']

```

```

; Check the dimensionality of the array.

```

```

s = size(array)
dim = s(0)
if dim eq 1 then begin
  num = s(1)
  ttyp = s(2)

```

```

; Slider/value examination with descend.

```

```

areditWindow = Widget_Base(Title = 'IDL 1D Array Editor', /Column, $
  Mbar=MenuBar, /TLB_Size_Events, $
  TLB_Frame_Attr=8)

```

```

p = 0

```

```

if ttyp eq 10 then begin
  t = execute('ss = size(*array['+string(p)+'])')
  vs = string(p)+' '+'pointer to '+vtyp2[ss(ss(0)+1)]
endif else if ttyp eq 8 then begin

```

```

  vs = string(p)+' '+'structure'
endif else if ttyp eq 11 then begin

```

```

  vs = string(p)+' '+'object'
endif else begin

```

```

  vs = string(p)+' '+'string(array[p])
endif
endelse

```

```

dataView = Widget_Label(areditWindow,/Align_Center,Frame = 2, $
  XSize = 400, Value = vs)

```

```

dataSlide = Widget_Slider(areditWindow,/Drag,Event_Pro='ArEdit_Slide', $
  Maximum = num-1)

```

```

FileMenu = Widget_Button(MenuBar, Value='File', /Menu)

```

```

DecButton = Widget_Button(FileMenu, Value='Descend', $
  Event_Pro = 'ArEdit_Descend')

```

```

ExitButton = Widget_Button(FileMenu, Value='Exit', $
  /Separator, Event_Pro = 'Generic_Exit')

```

```

info = { areditText : 0L,          $ ;
  dataSlide : dataSlide,        $ ; slider widget ID
  dataView : dataView ,        $ ; Label widget ID
  array : ptr_new(array),      $ ; pointer to array
  p : p,                        $ ; slider position
  ttyp : ttyp,                  $ ;
  pval : array[p]              } ;

```

```
Widget_Control, areditWindow, Set_UValue = info
```

```
Widget_Control, areditWindow, /REALIZE
```

```
XManager, 'aredit', areditWindow
```

```
endif else if dim eq 2 then begin
```

```
  xnum = s(1)
```

```
  ynum = s(2)
```

```
  ttyp = s(3)
```

```
; Image display? with X, Y select.
```

```
x = 0
```

```
y = 0
```

```
aval = array[x,y]
```

```
if ttyp eq 10 then begin
```

```
  ; Array of pointers, no edit.
```

```
  return
```

```
endif else if ttyp eq 8 then begin
```

```
  ; Array of structures, no edit.
```

```
  return
```

```
endif else if ttyp eq 11 then begin
```

```
  ; Array of objects, no edit.
```

```
  return
```

```
endif
```

```
if max([xnum,ynum]) le 7 then begin
```

```
  SpreadSheet,array,xnum,ynum,ttyp
```

```
  return
```

```
endif
```

```
c = imscale(array,10.0)
```

```
z1 = float(c[0])
```

```
z2 = float(c[1])
```

```
scale = 400.0/max([xnum,ynum])
```

```
areditWindow = Widget_Base(Title = 'IDL 2D Array Editor', /Column, $
```

```
  Mbar=MenuBar, /TLB_Size_Events, $
```

```
  TLB_Frame_Attr=8)
```

```
rowBase = Widget_Base(areditWindow, /Row)
```

```
dataView = Widget_Draw(RowBase, /Align_Center, XSize = fix(xnum*scale), $
```

```

YSize = fix(ynum*scale), /Button_Events, $
Event_Pro='ArEd_Draw')
YSlide = Widget_Slider(RowBase,/Drag,Event_Pro='ArEd_SlideXY', $
  UValue='yslide',Maximum = ynum-1,/Vertical, $
  YSize = fix(ynum*scale))
XSlide = Widget_Slider(areditWindow,/Drag,Event_Pro='ArEd_SlideXY', $
  UValue='xslide',Maximum = xnum-1, $
  XSize = fix(xnum*scale))
pixelView = Widget_Label(areditWindow,/Align_Left,Frame = 1, $
  XSize = fix(xnum*scale), Value = string(aval))
ZBase = Widget_Base(areditWindow, /Row)
Z1Label = Widget_Label(ZBase,Value='Z1: ')
Z1Text = Widget_Text(ZBase,Value=string(z1),/Editable,XSize=10, $
  Event_Pro = 'ArEd_SlideXY')
Z2Label = Widget_Label(ZBase,Value='Z2: ')
Z2Text = Widget_Text(ZBase,Value=string(z2),/Editable,XSize=10, $
  Event_Pro = 'ArEd_SlideXY')

```

```

FileMenu = Widget_Button(MenuBar, Value='File', /Menu)
ExitButton = Widget_Button(FileMenu, Value='Exit', $
  /Separator, Event_Pro = 'Generic_Exit')

```

```

info = { areditText : 0L,          $ ;
  XSlide   : XSlide,             $ ; X slider widget ID
  YSlide   : YSlide,             $ ; Y slider widget ID
  dataView : dataView,           $ ; Draw widget ID
  pixelView : pixelView,         $ ; Label widget ID
  array    : ptr_new(array),     $ ; pointer to array
  Z1Text   : Z1Text,             $ ; z1 widget ID
  Z2Text   : Z2Text,             $ ; z2 widget ID
  scale    : scale,              $ ; scale
  x        : x,                  $ ; x slider position
  y        : y,                  $ ; y slider position
  ttyp     : ttyp,               $ ;
  aval     : aval                 } ;

```

```
Widget_Control, areditWindow, Set_UValue = info
```

```
Widget_Control, areditWindow, /REALIZE
```

```

xx = !D.WINDOW
Widget_Control, dataView, Get_Value = drawid1
wset,drawid1
tv, bytscl(congrid(array,fix(xnum*scale), fix(ynum*scale), cubic=-.5), $
top=!d.n_colors-1,min=z1, max=z2)
wset,xx
big_cursor, scale*x, scale*y, -1, -1

```

```

XManager, 'aredit', areditWindow
endif else begin
; display size and type, no edit. (or)
; 'select a slice' and pass back to here.
; that is dimension = dimension - 1
numdim = s(0)
ttyp = s(s(0)+1)

; Slider/value examination with descend.
areditWindow = Widget_Base(Title = 'IDL nD Array Editor', /Column, $
    Mbar=MenuBar, /TLB_Size_Events, $
    TLB_Frame_Attr=8)
dimLabel = Widget_Label(areditWindow,/Align_Center, $
    Value = 'array, '+string(numdim)+' dimensions.')

; Choose dimension along which to descend and choose plane/hyperplane.
dimSlide = Widget_Slider(areditWindow,/Drag,Event_Pro='ArEd_multidim', $
    Maximum = numdim-1,Title='Compression Dimension')
pickSlide = Widget_Slider(areditWindow,/Drag,Event_Pro='ArEd_multidim', $
    Maximum = s(1),Title='Hyperplane Choice')

FileMenu = Widget_Button(MenuBar, Value='File', /Menu)
DecButton = Widget_Button(FileMenu, Value='Descend', $
    Event_Pro = 'ArEdDim_Descend')
ExitButton = Widget_Button(FileMenu, Value='Exit', $
    /Separator, Event_Pro = 'Generic_Exit')

info = { placeholder : 0L,          $ ;
    dimSlide : dimSlide,          $ ; dimension slider widget ID
    pickSlide : pickSlide,        $ ; pick plane slider widget ID
    crunchdim : 0,                $ ; dimension to crunch
    thisplane : 0,                $ ; hyperplane choise
    array : ptr_new(array),       $ ; pointer to array
    numdim : numdim,              $ ; number of dimensions
    ttyp : ttyp,                  $ ;
    ph : 0                        };

Widget_Control, areditWindow, Set_UValue = info

Widget_Control, areditWindow, /REALIZE
XManager, 'aredit', areditWindow
endelse

end

```

```

pro varedit,var
  t = dialog_message(string(var))
end

```

```

pro StEdit_Descend,event

```

```

  Widget_Control, event.top, Get_UValue=info
  tnames = tag_names(*info.struct)

```

```

; Check to be sure that this entry is a valid pointer, or structure, or
; and array, return if not.

```

```

t = execute('s = size((*info.struct).'+tnames(info.ypos)+)')
if s(s(0)+1) ne 10 and s(s(0)+1) ne 8 and s(0) eq 0 then begin
  t = dialog_message('Sorry, tag is not a pointer or structure or array.')
  return
end

```

```

if s(s(0)+1) eq 10 then begin
  t = execute('test = ptr_valid((*info.struct).'+tnames(info.ypos)+)')
  if test[0] eq 0 then begin
    t = dialog_message('Sorry, tag refers to an invalid pointer.')
    return
  endif
endif
endif

```

```

if s(0) gt 0 and s(s(0)+2) gt 1 then begin
  t=execute('arrayedit,(*info.struct).'+tnames(info.ypos))
endif else if s(s(0)+1) eq 10 then begin
  t = execute('ss = size((*info.struct).'+tnames(info.ypos)+)')
  if ss(ss(0)+1) eq 8 then begin
    t=execute('structedit,(*info.struct).'+tnames(info.ypos)+' ')
  endif else if ss(0) gt 0 then begin
    t=execute('arrayedit,(*info.struct).'+tnames(info.ypos)+' ')
  endif else begin
    t=execute('varedit,(*info.struct).'+tnames(info.ypos)+)')
  endelse
endif else if s(s(0)+1) eq 8 then begin
  t=execute('structedit,(*info.struct).'+tnames(info.ypos))
endif else begin
  t=execute('arrayedit,(*info.struct).'+tnames(info.ypos))
endelse

```

```

  Widget_Control, event.top, Set_UValue=info
end

```

```

pro EdWin_Event,event

```

```

  Widget_Control, event.top, Get_UValue=info
  if event.type eq 3 then begin

```

```

; Where did they click?
pos = event.offset
columnRow = Widget_Info(info.steditText, Text_Offset_To_XY=pos)
row = columnRow(1)

; Remember this.
info.ypos = row
endif
Widget_Control, event.top, Set_UValue=info
end

```

```

pro Generic_Exit, event
Widget_Control, event.top, /Destroy
end

```

```

pro structedit,struct

```

```

ntags = n_tags(struct)
tnames = tag_names(struct)
maxlenname = max(strlen(tnames))
vtyp = ['undefined',' byte',' integer',' long_int',' float', $
' double',' complex',' string','structure','dbl_cmplx', $
' pointer',' object']
vtyp2=['undefined','byte','integer','long_int','float',$
'double','complex','string','structure','dbl_cmplx',$
'pointer','object']

for i = 0,ntags-1 do begin
t = execute('s = size(struct.'+tnames(i)+'')')
t = execute('val = struct.'+tnames(i))

if s(s(0)+1) eq 10 then begin
t = execute('test = ptr_valid(struct.'+tnames(i)+'')')
if test[0] ne 0 then begin
t = execute('ss = size(*(struct.'+tnames(i)+'')[0])')
endif
endif

if s(0) gt 0 and s(s(0)+1) ne 8 then begin
val = '(array)'
endif else if s(s(0)+1) eq 10 then begin
if test[0] ne 0 then begin
if ss(0) gt 0 then begin
val='(array of '+vtyp2[ss(ss(0)+1)]+')'
endif else begin
val='('+vtyp2[ss(ss(0)+1)]+')'

```

```

endelse
  endif else begin
val='nullpointer'
  endelse
  endif else if s(s(0)+1) eq 11 then begin
    val='object'
  endif else if s(s(0)+1) eq 8 then begin
    val='structure'
  endif else if s(s(0)+1) eq 0 then begin
    val=''
  endif

  spcnt = maxlenname-strlen(tnames(i)) & sppad = ''
  for k = 1, spcnt do sppad = sppad + ' '
  if i eq 0 then begin
    labs = ' '+vtyp[s(s(0)+1)]+' '+tnames(i)
    txt = string(val)
  endif else begin
    labs = [labs,' '+vtyp[s(s(0)+1)]+' '+tnames(i)]
    txt = [txt,string(val)]
  endelse
endfor

; Build a GUI, most of which is a scrollable text field.
steditWindow = Widget_Base(Title = 'IDL Structure Editor', $
  Mbar=MenuBar, /Row, /TLB_Size_Events, $
  TLB_Frame_Attr=8);, XOffset = 200, YOffset = 100)

scrollBase = Widget_Base(steditWindow,/Row,/Scroll, $
  X_Scroll_Size=540,Y_Scroll_Size=600)

FileMenu = Widget_Button(MenuBar, Value='File', /Menu)
DecButton = Widget_Button(FileMenu, Value='Descend', $
  Event_Pro = 'StEdit_Descend')
ExitButton = Widget_Button(FileMenu, Value='Exit', $
  /Separator, Event_Pro = 'Generic_Exit')

stdispText = Widget_Text(scrollBase, XSize = 20, YSize = ntags, $
  Value = labs, $
  Event_Pro = 'EdWin_Event')
steditText = Widget_Text(scrollBase, XSize = 20, YSize = ntags, $
  /Editable, Value = txt, /All_Events, $
  Event_Pro = 'EdWin_Event')

info = { steditText : steditText,      $ ; the text window index number
        ypos       : 0,                $ ; selected position in struct
        struct     : ptr_new(struct),  $ ; pointer to struct
        temp       : 0                  } ; the draw widget identifier

```

Widget_Control, steditWindow, Set_UValue = info

Widget_Control, steditWindow, /REALIZE
XManager, 'stedit', steditWindow

end

File Attachments

1) [structedit.pro](#), downloaded 123 times
