Subject: imagen con ejes
Posted by hernan on Fri, 16 Sep 2005 15:14:37 GMT
View Forum Message <> Reply to Message

Necesito ayuda para modificar este ejemplo (tomado del Zoom in on data,
14 May 1998 07:50) de forma que al escoger una porcion de la imagen
este la muestre con sus valores de eje. Gracias.

```
;************************************************************ *
PRO mapEspectro_Widget_Events, event

; This is the event handler for the draw widget graphics window.
; Deal only with DOWN, UP, and MOTION events.
IF event.type GT 2 THEN RETURN
   ; Get the info structure.
Widget_Control, event.top, Get_UValue=info, /No_Copy
   ; What kind of event is this?
eventTypes = ['DOWN', 'UP', 'MOTION']
thisEvent = eventTypes[event.type]
CASE thisEvent OF
  'DOWN': BEGIN
        ; Turn motion events on for the draw widget.
     Widget_Control, info.drawID, Draw_Motion_Events=1
        ; Create a pixmap. Store its ID. Copy window contents into it.
     Window, /Free, /Pixmap, XSize=info.xsize, YSize=info.ysize
     info.pixID = !D.Window
     Device, Copy=[0, 0, info.xsize, info.ysize, 0, 0, info.wid]
        ; Get and store the static corner of the box.
     info.sx = event.x
     info.sy = event.y
     ENDCASE
  'UP': BEGIN
        ; Erase the last box drawn. Destroy the pixmap.
     WSet, info.wid
     Device, Copy=[0, 0, info.xsize, info.ysize, 0, 0, info.pixID]
     WDelete, info.pixID
        ; Turn draw motion events off.
        ;Clear any events queued for widget.
     Widget_Control, info.drawID, Draw_Motion_Events=0, Clear_Events=1
        ; Order the box coordinates.
     sx = Min([info.sx, event.x], Max=dx)
     sy = Min([info.sy, event.y], Max=dy)
        ; Make sure there was a drag, or just zoom out to
        ; full size.
        print, dx, sx, dy, sy
    IF Abs(dx - sx) LT 10 OR Abs(dy - sy) LT 10 THEN BEGIN
      CONTOUR, INFO.z, /XSTYLE, /YSTYLE
    ENDIF ELSE BEGIN
```

```
        ; Convert the coordinates to map data coordinates.
      !X = info.xscale
      !Y = info.yscale
      !Map = info.map
      latlon = Convert_Coord([sx,dx],[sy,dy], /Device,/To_Data)
      loncenter = ((LATLON[0,1] - LATLON[0,0]) / 2.0) + LATLON[0,0]
      latcenter = ((LATLON[1,1] - LATLON[1,0]) / 2.0) + LATLON[1,0]
        ; Draw the map.
      CONTOUR, INFO.z, XRANGE = [LATLON[0,0],LATLON[0,1]], $
      YRANGE = [LATLON[1,0],LATLON[1,1]], /XSTYLE, /YSTYLE
    ENDELSE
      ; Update the data scaling parameters.
    info.xscale = !X
    info.yscale = !Y
    info.map = !Map
    ENDCASE
  'MOTION': BEGIN
      ; Here is where the actual box is drawn and erased.
      ; First, erase the last box.
    WSet, info.wid
    Device, Copy=[0, 0, info.xsize, info.ysize, 0, 0, info.pixID]
      ; Get the coodinates of the new box and draw it.
    sx = info.sx
    sy = info.sy
    dx = event.x
    dy = event.y
    PlotS, [sx, sx, dx, dx, sx], [sy, dy, dy, sy, sy], /Device, $
      Color=info.boxColor
    ENDCASE
ENDCASE


  ; Store the info structure.
Widget_Control, event.top, Set_UValue=info, /No_Copy
END
 ;------------------------------------------------------------ ----------


PRO mapEspectro

xsize = 500
ysize = 500
nn=384
x=findgen(nn)-(nn-1.)/2.
y=x
z=fltarr(nn,nn)

for i=0,(nn-1) do begin
```

```
    for j=0,(nn-1) do begin
    z(i,j)=sqrt(x(i)^2+y(j)^2)
    endfor
endfor

   ; Create the TLB.
tlb = Widget_Base(Title='Zooming into Map Example Program')
   ; Create the draw widget graphics window. Turn button events ON.
drawID = Widget_Draw(tlb, XSize=xsize, YSize=ysize, Button_Events=1, $
x_scroll_size = 350, y_scroll_size = 350)
    ; Realize widgets and make draw widget the current window.
Widget_Control, tlb, /Realize
Widget_Control, drawID, Get_Value=wid
WSet, wid
   ; Load drawing color and display the initial map projection.
boxColor = !D.N_Colors-2
device, decomposed = 0
PX = !X.WINDOW * !D.X_VSIZE
PY = !Y.WINDOW * !D.Y_VSIZE
SZ = SIZE(z, /dimensions)
print, px, py
TVSCL, z, PX[0], PY[0]
contour,z,x,y, XSTYLE = 1, YSTYLE = 1, $
/DEVICE, /noerase, /nodata, $
POSITION = [PX[0], PY[0], PX[0]+SZ[0]-1, PY[0]+SZ[1]-1]

   ; Create an "info" structure with information to run the program.
info = { wid:wid, $          ; The window index number.
        drawID:drawID, $    ; The draw widget identifier.
        z:z, $              ; La matriz
        pixID:-1, $         ; The pixmap identifier (undetermined
now).
        xsize:xsize, $      ; The X size of the graphics window.
        ysize:ysize, $      ; The Y size of the graphics window.
        sx:-1, $            ; The X static corner of the box.
        sy:-1, $            ; The Y static corner of the box.
        xscale:!X, $        ; The X data scaling parameters.
        yscale:!Y, $        ; The Y data scaling parameters.
        map:!Map, $         ; The map scaling parameters.
        boxColor:boxColor } ; The rubberband box color.
   ; Store the info structure.
Widget_Control, tlb, Set_UValue=info, /No_Copy
   ; Start the program going.
XManager, 'mapEspectro', tlb, /No_Block, $
   Event_Handler='mapEspectro_Widget_Events'
END
```