Subject: FG image change
Posted by Helder Marchetto on Thu, 25 Jun 2015 10:46:20 GMT
View Forum Message <> Reply to Message

Hi,
as I mentioned in previous posts of mine, I would like to have a widget with a window to display images with image(). On top I would like to have markers.

I found two ways of doing this:
1) update the image using the setData method and two variables for x and y coordinates. This is ok, as long as you don't zoom or pan the image (discussed in a previous post).
2) delete all objects and recreate them. This does not have the pan/zoom problem, but is computationally more intensive.

Here are the results that I got by cycling 100 switching images:

Method 1) took 0.027 seconds
Method 2) the time increases linearly by 1.8 ms per cycle. It starts at 0.1 seconds and ends up at 0.3 after 100 iterations. This is adding also about 260 bytes of memory every round (26 kB after 100 iterations). Clearly evidence of some memory leakage somewhere...

Below I have attached the code to show this and my IDL version.

Is there a way in between these two methods that can get the job done?

Thanks,
Helder

```
function ww_mouse_motion_event, win, x, y, keymods
info = win.uvalue
MousePos = win->ConvertCoord(x, y, /device, /to_data)
widget_control, (*info).wBase, tlb_set_title=string(MousePos[0:1],format='("Pos=(",f0.2,",
",f0.2,")")')
return, 1
end

pro testFunctionGraphicsObjects_event, event
;help, event
widget_control, event.top, get_uvalue=info
case event.id of
 (*info).wButtonNext:begin
    (*info).counter = ~(*info).counter
    (*info).iObj->refresh, /disable
    if (*info).counter then begin
      xArr = (*info).d2[0]*dindgen((*info).s2[0])/((*info).s2[0]-1d)
      yArr = (*info).d2[1]*dindgen((*info).s2[1])/((*info).s2[1]-1d)
      (*info).iObj->setData, (*info).img2, xArr, yArr
```

```
      widget_control, (*info).wStatus,  set_value=string([(*info).s2[0],(*info).s2[1],(*info).d2[0],
(*info).d2[1]], format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")')
    endif else begin
      xArr = (*info).d1[0]*dindgen((*info).s1[0])/((*info).s1[0]-1d)
      yArr = (*info).d1[1]*dindgen((*info).s1[1])/((*info).s1[1]-1d)
      (*info).iObj->setData, (*info).img1, xArr, yArr
      widget_control, (*info).wStatus,  set_value=string([(*info).s1[0],(*info).s1[1],(*info).d1[0],
(*info).d1[1]], format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")')
    endelse
    (*info).iObj->scale, /reset
    (*info).iObj->rotate, /reset
    (*info).iObj->translate, /reset
    (*info).iObj->refresh
  end
 (*info).wButtonDeleteNext:begin
    (*info).counter = ~(*info).counter
    (*info).tdd->delete
    (*info).tnd->delete
    (*info).pdd->delete
    (*info).pnd->delete
    (*info).iObj->delete
    obj_destroy, (*info).tdd
    obj_destroy, (*info).tnd
    obj_destroy, (*info).pdd
    obj_destroy, (*info).pnd
    obj_destroy, (*info).iObj
    if (*info).counter then begin
      d = (*info).d2
      s = (*info).s2
      img = (*info).img2
      widget_control, (*info).wStatus,  set_value=string([(*info).s2[0],(*info).s2[1],(*info).d2[0],
(*info).d2[1]], format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")')
    endif else begin
      d = (*info).d1
      s = (*info).s1
      img = (*info).img1
      widget_control, (*info).wStatus,  set_value=string([(*info).s1[0],(*info).s1[1],(*info).d1[0],
(*info).d1[1]], format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")')
    endelse
    (*info).iObj = image(img, current=(*info).oWin, image_dimensions=d, margin=0)
    res = redrawObj((*info).iObj)
    (*info).tdd = res.tdd
    (*info).tnd = res.tnd
    (*info).pdd = res.pdd
    (*info).pnd = res.pnd
  end
  (*info).wButtonLoopNext:begin
    event.id = (*info).wButtonNext
```

```
    times = dblarr(101)
    help,/mem, output=out
    startPos = strpos(out, ':')+1
    endPos   = strpos(out, ',')
    startHeap = long(strmid(out, startPos, endPos-startPos))
    t0 = systime(1)
    for i = 0, 100l do begin
       testFunctionGraphicsObjects_event, event
       t1 = systime(1)
       times[i] = t1-t0
       t0 = t1
    endfor
    help,/mem, output=out
    startPos = strpos(out, ':')+1
    endPos   = strpos(out, ',')
    endHeap  = long(strmid(out, startPos, endPos-startPos))
    p = plot(times,'2r', xTitle='run Nr.', yTitle='time (sec)', title='Redraw, heap
used='+strtrim((endHeap-startHeap)/1024l,2)+' kB')
    print, 'average = '+string(mean(times),format='(f0.3)')+' seconds'
  end
  (*info).wButtonLoopDeleteNext:begin
    tic
    event.id = (*info).wButtonDeleteNext
    times = dblarr(101)
    help,/mem, output=out
    startPos = strpos(out, ':')+1
    endPos   = strpos(out, ',')
    startHeap = long(strmid(out, startPos, endPos-startPos))
    t0 = systime(1)
    help,/mem
    for i = 0, 100l do begin
       testFunctionGraphicsObjects_event, event
       t1 = systime(1)
       times[i] = t1-t0
       t0 = t1
    endfor
    help,/mem, output=out
    startPos = strpos(out, ':')+1
    endPos   = strpos(out, ',')
    endHeap  = long(strmid(out, startPos, endPos-startPos))
    p = plot(times,'2r', xTitle='run Nr.', yTitle='time (sec)', title='Delete and Draw, heap
used='+strtrim((endHeap-startHeap)/1024l,2)+' kB')
    fitRes = linfit(dindgen(101), times)
    ot = text(0.25,0.75, string(fitRes,format='("time = ",f0.3," + ",f0.9," runNr")'), 'b', /norm, target=p)
    op = plot(fitres[0]+fitres[1]*dindgen(101),'2b', overplot=p)
  end
  (*info).wBase:begin
   widget_control, (*info).wDraw, xSize=event.y, ySize=event.y
```

```
  end
  else:print, 'dunno what to do with id ', event.id
endcase
end

function redrawObj, iObj
tdd = text(3.0,0.3,'data defined', 'r', /data, target=iObj)
tnd = text(0.5,0.5,'norm defined', 'b', /norm, target=iObj)
pdd = polygon([3.0,4.0,4.0,3.0],[7.0,7.0,8.0,8.0],'r2', /data, target=iObj, /fill_background,
fill_color="red")
pnd = polygon([0.3,0.4,0.4,0.3],[0.3,0.3,0.4,0.4],'b2', /norm, target=iObj, /fill_background,
fill_color="light steel blue")
return, {tdd:tdd, tnd:tnd, pdd:pdd, pnd:pnd}
end

pro testFunctionGraphicsObjects_cleanup, event
print, 'cleanup'
widget_control, event, get_uvalue=info
ptr_free, info
end

pro testFunctionGraphicsObjects
;read the images
fNames = filepath(['mineral.png','endocell.jpg'], SUBDIRECTORY=['examples','data'])
img1 = read_png(fNames[0])
read_jpeg, fNames[1], img2
;get image sizes
s1 = size(img1, /dimensions)
s2 = size(img2, /dimensions)
;establish image dimensions (real space) as pixel sizes
pxs1 = 30d/s1[0]   ;the x axis is defined as 30 um long
pxs2 = 1d/s2[0]    ;the x axis is defined as  1 um long
d1 = s1*pxs1
d2 = s2*pxs2
;create starting widgets
wBase = widget_base(/row, title='Pos=(  0,  0)', /tlb_size_events)
wDraw = widget_window(wBase, xsize=600, ysize=600,
mouse_motion_handler='ww_mouse_motion_event')
wButtonCol = widget_base(wBase, /col)
wButtonNext = widget_button(wButtonCol, /dynamic_resize, value='next Image')
wButtonDeleteNext = widget_button(wButtonCol, /dynamic_resize, value='delete and next Image')
wButtonLoopNext = widget_button(wButtonCol, /dynamic_resize, value='loop next Image')
wButtonLoopDeleteNext = widget_button(wButtonCol, /dynamic_resize, value='loop delete and
next Image')
wStatus     = widget_label(wButtonCol,  /dynamic_resize, value=string([s1[0],s1[1],d1[0],d1[1]],
format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")'))
widget_control, wBase, /realize
; Retrieve the newly-created Window object.
```

```
widget_control, wDraw, get_value=oWin
iObj = image(img1, current=oWin, image_dimensions=d1, margin=0)
res = redrawObj(iObj)

info = ptr_new({wBase:wBase,$
                wDraw:wDraw,$
                wButtonNext:wButtonNext,$
                wButtonDeleteNext:wButtonDeleteNext,$
                wButtonLoopNext:wButtonLoopNext,$
                wButtonLoopDeleteNext:wButtonLoopDeleteNext,$
                wStatus:wStatus,$
                d1:d1,$
                d2:d2,$
                s1:s1,$
                s2:s2,$
                counter:0b,$
                img1:img1,$
                img2:img2,$
                oWin:oWin,$
                iObj:iObj,$
                tdd:res.tdd,$
                tnd:res.tnd,$
                pdd:res.pdd,$
                pnd:res.pnd})
oWin.uvalue=info
widget_control, wBase, set_uvalue=info
xmanager, 'testFunctionGraphicsObjects', wBase, cleanup =
'testFunctionGraphicsObjects_Cleanup', event_handler = 'testFunctionGraphicsObjects_event',
/no_block
end

IDL> !version
{
   "ARCH": "x86_64",
   "OS": "Win32",
   "OS_FAMILY": "Windows",
   "OS_NAME": "Microsoft Windows",
   "RELEASE": "8.4.1",
   "BUILD_DATE": "Feb 17 2015",
   "MEMORY_BITS": 64,
   "FILE_OFFSET_BITS": 64
}
```

Subject: Re: FG image change
Posted by Helder Marchetto on Fri, 26 Jun 2015 13:07:14 GMT
View Forum Message <> Reply to Message

Hi,
I'm going to answer my own post. After contacting Exelis/Harris support (thanks!), I got great feedback and as it usually goes, things can be pretty easy.
The trick is to set the image scale_factor property to 1 and the image scale_center property to half the image dimensions after using the setData method.
Below is the updated script.
In the script I gave before, there was another error. When defining the x and y axis ranges in the setMethod I divide by nPoints-1. This is wrong and should have been simply nPoints.

The script is below.

Cheers,
Helder

```
function ww_mouse_motion_event, win, x, y, keymods
info = win.uvalue
MousePos = win->ConvertCoord(x, y, /device, /to_data)
widget_control, (*info).wBase, tlb_set_title=string(MousePos[0:1],format='("Pos=(",f0.2,",
",f0.2,")")')
return, 1
end

pro testFunctionGraphicsObjects_event, event
;help, event
widget_control, event.top, get_uvalue=info
case event.id of
 (*info).wButtonNext:begin
    (*info).counter = ~(*info).counter
    (*info).iObj->refresh, /disable
    if (*info).counter then begin
      xArr = (*info).d2[0]*dindgen((*info).s2[0])/(*info).s2[0]
      yArr = (*info).d2[1]*dindgen((*info).s2[1])/(*info).s2[1]
      (*info).iObj->setData, (*info).img2, xArr, yArr
      (*info).iObj.scale_factor=1d
      (*info).iObj.scale_center=(*info).d2/2d
      widget_control, (*info).wStatus,  set_value=string([(*info).s2[0],(*info).s2[1],(*info).d2[0],
(*info).d2[1]], format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")')
    endif else begin
      xArr = (*info).d1[0]*dindgen((*info).s1[0])/(*info).s1[0]
      yArr = (*info).d1[1]*dindgen((*info).s1[1])/(*info).s1[1]
      (*info).iObj->setData, (*info).img1, xArr, yArr
      (*info).iObj.scale_factor=1d
      (*info).iObj.scale_center=(*info).d1/2d
      widget_control, (*info).wStatus,  set_value=string([(*info).s1[0],(*info).s1[1],(*info).d1[0],
(*info).d1[1]], format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")')
    endelse
    (*info).iObj->refresh
```

```
  end
 (*info).wButtonDeleteNext:begin
    (*info).counter = ~(*info).counter
    (*info).tdd->delete
    (*info).tnd->delete
    (*info).pdd->delete
    (*info).pnd->delete
    (*info).iObj->delete
    obj_destroy, (*info).tdd
    obj_destroy, (*info).tnd
    obj_destroy, (*info).pdd
    obj_destroy, (*info).pnd
    obj_destroy, (*info).iObj
    if (*info).counter then begin
      d = (*info).d2
      s = (*info).s2
      img = (*info).img2
      widget_control, (*info).wStatus,  set_value=string([(*info).s2[0],(*info).s2[1],(*info).d2[0],
(*info).d2[1]], format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")')
    endif else begin
      d = (*info).d1
      s = (*info).s1
      img = (*info).img1
      widget_control, (*info).wStatus,  set_value=string([(*info).s1[0],(*info).s1[1],(*info).d1[0],
(*info).d1[1]], format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")')
    endelse
    (*info).iObj = image(img, current=(*info).oWin, image_dimensions=d, margin=0)
    res = redrawObj((*info).iObj)
    (*info).tdd = res.tdd
    (*info).tnd = res.tnd
    (*info).pdd = res.pdd
    (*info).pnd = res.pnd
  end
  (*info).wButtonLoopNext:begin
    event.id = (*info).wButtonNext
    times = dblarr(101)
    help,/mem, output=out
    startPos = strpos(out, ':')+1
    endPos   = strpos(out, ',')
    startHeap = long(strmid(out, startPos, endPos-startPos))
    t0 = systime(1)
    for i = 0, 100l do begin
       testFunctionGraphicsObjects_event, event
       t1 = systime(1)
       times[i] = t1-t0
       t0 = t1
    endfor
    help,/mem, output=out
```

```
    startPos = strpos(out, ':')+1
    endPos   = strpos(out, ',')
    endHeap  = long(strmid(out, startPos, endPos-startPos))
    p = plot(times,'2r', xTitle='run Nr.', yTitle='time (sec)', title='Redraw, heap
used='+strtrim((endHeap-startHeap)/1024l,2)+' kB')
    print, 'average = '+string(mean(times),format='(f0.3)')+' seconds'
  end
  (*info).wButtonLoopDeleteNext:begin
    tic
    event.id = (*info).wButtonDeleteNext
    times = dblarr(101)
    help,/mem, output=out
    startPos = strpos(out, ':')+1
    endPos   = strpos(out, ',')
    startHeap = long(strmid(out, startPos, endPos-startPos))
    t0 = systime(1)
    help,/mem
    for i = 0, 100l do begin
      testFunctionGraphicsObjects_event, event
      t1 = systime(1)
      times[i] = t1-t0
      t0 = t1
    endfor
    help,/mem, output=out
    startPos = strpos(out, ':')+1
    endPos   = strpos(out, ',')
    endHeap  = long(strmid(out, startPos, endPos-startPos))
    p = plot(times,'2r', xTitle='run Nr.', yTitle='time (sec)', title='Delete and Draw, heap
used='+strtrim((endHeap-startHeap)/1024l,2)+' kB')
    fitRes = linfit(dindgen(101), times)
    ot = text(0.25,0.75, string(fitRes,format='("time = ",f0.3," + ",f0.9," runNr")'), 'b', /norm, target=p)
    op = plot(fitres[0]+fitres[1]*dindgen(101),'2b', overplot=p)
  end
  (*info).wBase:begin
   widget_control, (*info).wDraw, xSize=event.y, ySize=event.y
  end
  else:print, 'dunno what to do with id ', event.id
endcase
end

function redrawObj, iObj
tdd = text(3.0,0.3,'data defined', 'r', /data, target=iObj)
tnd = text(0.5,0.5,'norm defined', 'b', /norm, target=iObj)
pdd = polygon([3.0,4.0,4.0,3.0],[7.0,7.0,8.0,8.0],'r2', /data, target=iObj, /fill_background,
fill_color="red")
pnd = polygon([0.3,0.4,0.4,0.3],[0.3,0.3,0.4,0.4],'b2', /norm, target=iObj, /fill_background,
fill_color="light steel blue")
return, {tdd:tdd, tnd:tnd, pdd:pdd, pnd:pnd}
```

```
end

pro testFunctionGraphicsObjects_cleanup, event
print, 'cleanup'
widget_control, event, get_uvalue=info
ptr_free, info
end

pro testFunctionGraphicsObjects
;read the images
fNames = filepath(['mineral.png','endocell.jpg'], SUBDIRECTORY=['examples','data'])
img1 = read_png(fNames[0])
read_jpeg, fNames[1], img2
;get image sizes
s1 = size(img1, /dimensions)
s2 = size(img2, /dimensions)
;establish image dimensions (real space) as pixel sizes
pxs1 = 30d/s1[0]   ;the x axis is defined as 30 um long
pxs2 = 1d/s2[0]    ;the x axis is defined as  1 um long
d1 = s1*pxs1
d2 = s2*pxs2
;create starting widgets
wBase = widget_base(/row, title='Pos=(  0,  0)', /tlb_size_events)
wDraw = widget_window(wBase, xsize=600, ysize=600,
mouse_motion_handler='ww_mouse_motion_event')
wButtonCol = widget_base(wBase, /col)
wButtonNext = widget_button(wButtonCol, /dynamic_resize, value='next Image')
wButtonDeleteNext = widget_button(wButtonCol, /dynamic_resize, value='delete and next Image')
wButtonLoopNext = widget_button(wButtonCol, /dynamic_resize, value='loop next Image')
wButtonLoopDeleteNext = widget_button(wButtonCol, /dynamic_resize, value='loop delete and
next Image')
wStatus     = widget_label(wButtonCol,  /dynamic_resize, value=string([s1[0],s1[1],d1[0],d1[1]],
format='(i4,"x",i4," pix, ",f0.2,"x",f0.2," mm")'))
widget_control, wBase, /realize
; Retrieve the newly-created Window object.
widget_control, wDraw, get_value=oWin
iObj = image(img1, current=oWin, image_dimensions=d1, margin=0)
res = redrawObj(iObj)

info = ptr_new({wBase:wBase,$
            wDraw:wDraw,$
            wButtonNext:wButtonNext,$
            wButtonDeleteNext:wButtonDeleteNext,$
            wButtonLoopNext:wButtonLoopNext,$
            wButtonLoopDeleteNext:wButtonLoopDeleteNext,$
            wStatus:wStatus,$
            d1:d1,$
            d2:d2,$
```

```
            s1:s1,$
            s2:s2,$
            counter:0b,$
            img1:img1,$
            img2:img2,$
            oWin:oWin,$
            iObj:iObj,$
            tdd:res.tdd,$
            tnd:res.tnd,$
            pdd:res.pdd,$
            pnd:res.pnd})
oWin.uvalue=info
widget_control, wBase, set_uvalue=info
xmanager, 'testFunctionGraphicsObjects', wBase, cleanup =
'testFunctionGraphicsObjects_Cleanup', event_handler = 'testFunctionGraphicsObjects_event',
/no_block
end
```
