Subject: Re: Persistent Object Graphics Not Persistent in Resizeable Windows
Posted by davidf on Fri, 06 Jun 1997 07:00:00 GMT
View Forum Message <> Reply to Message

[A copy of this article was sent to the cited author.]

Phil Williams writes:

> There is a suggestion in one of the manuals that retain=2 should not be
> used for draw widgets that are objects (grpahics_level=2). Instead use
> /expose_events keyword and then have something like this
>
> if event.id eq info.drawID then begin
> if event.type eq 4 then info.thisWindow->draw, info.thisView
> endif
>
> in the event handler to draw the window when it's exposed.
>
> I added this to your sample code and the axes appear fine when the
> widget is resized.

Thanks to Phil Williams and RSI for pointing out my problems
with this code. In fact, setting the Retain flag for object
windows is NOT a good idea. Removing it got my code working
fine.

Here is what I heard from RSI about this issue:

> As far as 'retain' goes. If you have are using a widget application
> I would recommend that you always set your retain to 0 then process
> the EXPOSE event in your event handler by calling the draw method
> on your window object.
>
> This is mainly a performance/memory issue. Using retain of 0 allows
> OpenGL to double buffer the window and drawings occur much faster.
> Since you have the window retained already (the view/scene object)
> you can make the determination of when to reissue a draw command.
> With retain=2, OpenGL has a buffer of the graphic, IDL has a buffer
> of the graphic (so it can repair the bits), and you have a 'buffer'
> to the graphic (the object). Under IDL 4.x this wasn't the case,
> so retain of 2 was the best choice (and still is for level 1
> graphics).

Good information to have. I'll update the data base and add a tip
to my web page.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Customizable IDL Programming Courses
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com
IDL 5 Reports: http://www.dfanning.com/documents/anomaly5.html

---

Subject: Re: Persistent Object Graphics Not Persistent in Resizeable Windows
Posted by Phil Williams on Fri, 06 Jun 1997 07:00:00 GMT
View Forum Message <> Reply to Message

David,

There is a suggestion in one of the manuals that retain=2 should not be
used for draw widgets that are objects (grpahics_level=2). Instead use
/expose_events keyword and then have something like this

```
if event.id eq info.drawID then begin
   if event.type eq 4 then info.thisWindow->draw, info.thisView
endif
```

in the event handler to draw the window when it's exposed.

I added this to your sample code and the axes appear fine when the
widget is resized.

Here's the edited code:

```
 ;---------------------------------------------------------- -------

Pro XSurface_Cleanup, tlb

   ; Come here when program dies. Free all created objects.

Widget_Control, tlb, Get_UValue=info
Obj_Destroy, info.thisContainer
END
 ;---------------------------------------------------------- --------

PRO Draw_Event, event
; Event handler to handle draw events.

Widget_Control, event.top, Get_UValue=info, /No_Copy
```

```
; The window was exposed so redraw it!
if event.type eq 4 then info.thisWindow->draw, info.thisView

Widget_Control, event.top, Set_UValue=info, /No_Copy

end

 ;------------------------------------------------------------ -------

PRO XSurface_Resize, event

     ; The only events generated by this simple program are resize
     ; events, which are handled here.

     ; Get the info structure.

Widget_Control, event.top, Get_UValue=info, /No_Copy

    ; Resize the draw widget.

 ;88888888888888888888888888888888888888888888888888888888 888
; Here is the problem. Comment this line out and uncomment the
; two below it to see that the problem comes from resizing the
; window.

info.thisWindow->SetProperty, Dimension=[event.x, event.y]

;info.thisWindow->erase, Color=[0,0,0]
;Wait, 1
 ;88888888888888888888888888888888888888888888888888888888 888

    ; Redisplay the graphic.

info.thisWindow->Draw, info.thisView

    ;Put the info structure back.

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


PRO XSurface, data

    ; Need some data.

Catch, error
IF error NE 0 THEN BEGIN  ; Can't find LoadData.
```

```
   data = DIST(40)
   GOTO, CreateView
ENDIF

IF N_Params() EQ 0 THEN   data = LoadData(2)

CreateView:
Catch, /Cancel

    ; Create a view. Use RGB color. Charcoal background.
    ; The viewplane rectangle extends from -1 to 1 in XY directions.

thisView = OBJ_NEW('IDLgrView', Color=[80,80,80],
Viewplane_Rect=[-1,-1,2,2])

    ; Create a model and add it to the view.

thisModel = OBJ_NEW('IDLgrModel')
thisView->Add, thisModel

    ; Create a wire mesh surface. Make it yellow.

thisSurface = OBJ_NEW('IDLgrSurface', data, Color=[255,255,0])

    ; Create axes for the surface. Color them green.

xAxis = Obj_New("IDLgrAxis", 0, color=[0,255,0], Ticklen=0.1)
yAxis = Obj_New("IDLgrAxis", 1, color=[0,255,0], Ticklen=0.1)
zAxis = Obj_New("IDLgrAxis", 2, color=[0,255,0], Ticklen=0.1)

    ; Add the surface and axes to the model.

thisModel->Add, thisSurface
thisModel->Add, xAxis
thisModel->Add, yAxis
thisModel->Add, zAxis

    ; Get the data ranges for the surface.

thisSurface->GetProperty,XRange=xrange,YRange=yrange,ZRange=zrange

    ; Set the range and location of the axes. In this case,
    ; we are scaling the data into -0.5 to 0.5, so that even
    ; when the surface is rotated, it stays inside the -1 to 1
    ; viewing rectangle. Note that not all values in the Location
    ; keyword are used. (I've put really large values into the
    ; positions that are not being used to demonstate this.) For
    ; example, with the X axis only the Y and Z locations are used.
```

```
xAxis->SetProperty, Range=xrange, Location=[1000.0, -0.5, -0.5]
yAxis->SetProperty, Range=yrange, Location=[-0.5, 1000.0, -0.5]
zAxis->SetProperty, Range=zrange, Location=[-0.5, 0.5, 1000]

    ; Scale the surface and axes into the range -0.5 to 0.5.

xs = [-0.5, 1/(xrange[1]-xrange[0])]
ys = [-0.5, 1/(yrange[1]-yrange[0])]
zs = [-0.5, 1/(zrange[1]-zrange[0])]
thisSurface->SetProperty,XCoord_Conv=xs, YCoord_Conv=ys, ZCoord_Conv=zs
xAxis->SetProperty, XCoord_Conv=xs
yAxis->SetProperty, YCoord_Conv=ys
zAxis->SetProperty, ZCoord_Conv=zs

    ; Rotate model to the standard view.

thisModel->Rotate,[1,0,0], -90  ; To get the Z-axis vertical.
thisModel->Rotate,[0,1,0],  30  ; Rotate it slightly to the right.
thisModel->Rotate,[1,0,0],  30  ; Rotate it down slightly.

    ;Create the widgets to view the surface.

tlb = Widget_Base(Title='Surface Example', Column=1, TLB_Size_Events=1)
drawID = Widget_Draw(tlb, XSize=300, YSize=300, Graphics_Level=2, $
 /expose_events, $
 event_pro = 'draw_event')
Widget_Control, tlb, /Realize

    ; Get the window destination object.

Widget_Control, drawID, Get_Value=thisWindow

    ; Draw the view in the window.

thisWindow->Draw, thisView

   ; Create a container object to hold all the other
   ; objects. This will make it easy to free all the
   ; objects when we are finished with the program.

thisContainer = Obj_New('IDLgrContainer')

   ; Add the view to the container.

thisContainer->Add, thisView

   ; Create an INFO structure to hold needed program information.
```

```
info = { thisContainer:thisContainer, $    ; The object container.
         drawID:drawID, $       ; The draw widget ID
         thisWindow:thisWindow, $          ; The destination window
object.
         thisView:thisView }               ; The view object.

   ; Store the info structure in the UValue of the TLB.

Widget_Control, tlb, Set_UValue=info, /No_Copy

   ; Call XManager. Set a cleanup routine so the objects
   ; can be freed upon exit from this program.

XManager, 'xsurface', tlb, Cleanup='XSurface_Cleanup', /No_Block, $
   Event_Handler='XSurface_Resize'
END

--
 /************************************************************ ********/
  Phil Williams, Ph.D.
  Research Instructor
  Children's Hospital Medical Center    "One man gathers what
  Imaging Research Center                 another man spills..."
  3333 Burnet Ave.                     -The Grateful Dead
  Cincinnati, OH 45229
  email: williams@irc.chmcc.org
  URL: http://scuttle.chmcc.org/~williams/
 /************************************************************ ********/
```

Subject: Re: Persistent Object Graphics Not Persistent in Resizeable Windows
Posted by davidf on Wed, 11 Jun 1997 07:00:00 GMT

View Forum Message <> Reply to Message

Stein Vidar writes:

> David Fanning wrote:
>>
>> I've run into a problem in which I lose all my axes labeling
>> if I resize my draw widget object window before evoking the
>> Draw method of the window. Here is a bug report and some
>> sample code to see the problem. (Those of you who want to
>> see how to add axes to a surface plot, take note. :-))
>
> This posting has kept me scratching my head for quite a
> while.
>

> I've tested the program that was attatched, on a machine
> with !version = { alpha OSF unix 5.0 Apr 28 1997}
> and my problem is that I can't figure out what the problem
> is/was... I can't see any difference between the plot before
> and after resizing... What *was* the problem - did the
> axes disappear, did the axes *values* disappear, or what..

What happened on my machine is that the axis labelling
disappeared, although not the axes themselves, when the
window was resized.

> In toying around with it I even managed to get IDL into
> a state where it didn't process any resizing events at
> all, though I can't reproduce that state... calling
> xmanager didn't do anything (though since this is a non-
> blocking widget I guess it shouldn't..) Hope this doesn't
> happen often...might have been me doing something strange..

I've noticed this myself, but like you, I can't reproduce
it. I've even had random crashes just from closing object
windows, but I can't reproduce that in a reliable way
either. It seems to have gone away now that I know a little
more what I am doing. :-)

> Also, I can't help but feeling that the redraw is quite
> slow compared to "direct" graphics... Is this something
> other people notice as well?

I think in general object redraw will be slower than
direct graphic redraw. This is due to the fact that
objects must be rendered or sent through the graphics
system each time. I have not noticed in the simple
things I have done so far that speed is much of an
issue.

> And what on earth does the retain state have to do with
> whether or not axes are appearing correctly or not..?

Well, exactly! The IDL documentation certainly warns
you that retaining object windows is not recommended,
but they don't tell you that weird things will happen
if you choose to ignore their recommendations.

> Utterly confused...

Welcome to the club! :-)

Although I will say that I think I have the darn

view rectangle thing figured out finally. When I
put an axis on a plot I am about 90% sure where it
is going to show up now. That's progress!

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Customizable IDL Programming Courses
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com
IDL 5 Reports: http://www.dfanning.com/documents/anomaly5.html

---

Subject: Re: Persistent Object Graphics Not Persistent in Resizeable Windows
Posted by Stein Vidar Hagfors H on Wed, 11 Jun 1997 07:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
>
> Folks,
>
> I've run into a problem in which I lose all my axes labeling
> if I resize my draw widget object window before evoking the
> Draw method of the window. Here is a bug report and some
> sample code to see the problem. (Those of you who want to
> see how to add axes to a surface plot, take note. :-))
>

This posting has kept me scratching my head for quite a
while.

I've tested the program that was attatched, on a machine
with !version = { alpha OSF unix 5.0 Apr 28 1997}
and my problem is that I can't figure out what the problem
is/was... I can't see any difference between the plot before
and after resizing... What *was* the problem - did the
axes disappear, did the axes *values* disappear, or what..

In toying around with it I even managed to get IDL into
a state where it didn't process any resizing events at
all, though I can't reproduce that state... calling
xmanager didn't do anything (though since this is a non-
blocking widget I guess it shouldn't..) Hope this doesn't
happen often...might have been me doing something strange..

Also, I can't help but feeling that the redraw is quite

slow compared to "direct" graphics... Is this something other people notice as well?

And what on earth does the retain state have to do with whether or not axes are appearing correctly or not..?

Utterly confused...

Stein Vidar
