Subject: Object Graphics
Posted by Reinhold Kroll on Wed, 23 Jul 1997 07:00:00 GMT
View Forum Message <> Reply to Message

has anybody out there successfully used object graphics
from IDL 5.0?

I put a very simple sine wave to the printer object and
it creates me a 15 MB file, runs quarter of an hour or
so. => completely useless.

Cheers, Reinhold
--

```
                    '''
                   (o o)
 /-------------------------oOO--(_)--OOo---------- ----------\
|                                                             |
|     |      Reinhold Kroll - Instituto de Astrofisica de Canarias   |
|  |/ |/ |/   C/ Via Lactea s/n, 38200 - La Laguna, Tenerife, Spain   |
|  |  |\ |    Phone: (++34)-22-605-290       Fax: (++34)-22-605-210   |
|             kroll@iac.es                  http://www.iac.es/   |
 \------------------------------------------------------- ----------/
```


Subject: Re: Object Graphics
Posted by David Fanning on Tue, 11 May 2004 23:18:22 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> It took a good 30
> minutes to figure out how to draw a single diagonal line.

And you saw it on the display!? You are *so* lucky!! :-)

Cheers,

David

P.S. That little error means that something is totally
screwed up. Maybe you knew that. :-)

I'd try setting the RETAIN keyword to 2 and see if
that helps.

--
David Fanning, Ph.D.
Fanning Software Consulting

Subject: Re: Object Graphics
Posted by Michael Wallace on Wed, 12 May 2004 00:58:00 GMT
View Forum Message <> Reply to Message

>> It took a good 30
>> minutes to figure out how to draw a single diagonal line.
>
>
> And you saw it on the display!? You are *so* lucky!! :-)

Oh, joy. This sounds like I'm going to be in for a fun adventure. But, yes, I saw it on the display. So, is seeing something on the display 30 minutes after you look at your first object graphics code somewhere close to a world record?

> P.S. That little error means that something is totally
> screwed up. Maybe you knew that. :-)

Mr. Obvious strikes again! :-)

> I'd try setting the RETAIN keyword to 2 and see if
> that helps.

Already was. And I can definitely tell you that the Window is not retaining anything. Direct graphics windows just like they're supposed to in the retain retain department.

-Mike

Subject: Re: Object Graphics
Posted by Michael Wallace on Wed, 12 May 2004 01:06:04 GMT
View Forum Message <> Reply to Message

> Direct graphics windows just like they're supposed
> to in the retain retain department.

What?! That makes no sense whatsoever! I need to lay off the caffeine or something. Let me try that sentence again. *ahem* Direct graphics windows work correctly with respect to the retain keyword.

-Mike

Subject: Re: Object Graphics
Posted by David Fanning on Wed, 12 May 2004 01:12:34 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> So, is seeing something on the display 30
> minutes after you look at your first object graphics code somewhere
> close to a world record?

Pretty much. You were smart to choose a line and not
an image. Most people just starting out stare at
the first pixel of their image looming large on
their display for *days* before they figure it out. :-)

>> I'd try setting the RETAIN keyword to 2 and see if
>> that helps.
>
> Already was. And I can definitely tell you that the Window is not
> retaining anything. Direct graphics windows just like they're supposed
> to in the retain retain department.

Well, you can make the argument that retaining an object
graphics window is a gross misuse of the Earth's resources.
Why retain? You have the whole damn graphics hierarchy
already in memory. When you get an EXPOSE event in your
draw widget, just re-draw the graphics.

That's what you *had* to do when object graphics were
first released, although they have become more user
friendly over the years. (To much chagrin in the IEPA,
I can tell you!) You can think of this (if you choose,
of course) as your machine wanting to force you into
good programming practices. Those of us with Windows
machines will be drooling at your good luck! :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object Graphics
Posted by Michael Wallace on Wed, 12 May 2004 02:42:27 GMT

> Well, you can make the argument that retaining an object
> graphics window is a gross misuse of the Earth's resources.
> Why retain? You have the whole damn graphics hierarchy
> already in memory. When you get an EXPOSE event in your
> draw widget, just re-draw the graphics.

window = obj_new('IDLgrWindow', RETAIN=0)

This command works like a charm. No error. No problem. I had a setup file I was using which always set RETAIN to 2 since my work up until now had been in Direct Graphics. And for the curious, here's what RSI has to say about this buried way down in their documentation:

In some situations, IDL cannot provide this backing store.

In IDL Object Graphics, it is almost always best to disable backing store (that is, set the RETAIN property equal to zero). This is because drawing to an off-screen pixmap (which is what happens when backing store is enabled) almost always bypasses any hardware graphics acceleration that may be available, causing all rendering to be done in software. To ensure that windows are redrawn properly, enable the generation of expose events on the WIDGET_DRAW window and redraw the windows explicitly when an expose event is received.

If you are using software rendering (that is, the RENDERER property is set equal to one), IDL will refresh the window automatically regardless of the setting of the RETAIN property.

> That's what you *had* to do when object graphics were
> first released, although they have become more user
> friendly over the years. (To much chagrin in the IEPA,
> I can tell you!) You can think of this (if you choose,
> of course) as your machine wanting to force you into
> good programming practices. Those of us with Windows
> machines will be drooling at your good luck! :-)

Subject: Re: Object Graphics
Posted by Mark Hadfield on Wed, 12 May 2004 03:33:21 GMT

Michael Wallace wrote:
...And for the curious, here's what RSI has
> to say about this buried way down in their documentation:
>
> In some situations, IDL cannot provide this backing store.

>
> In IDL Object Graphics, it is almost always best to disable backing
> store (that is, set the RETAIN property equal to zero). This is because
> drawing to an off-screen pixmap (which is what happens when backing
> store is enabled) almost always bypasses any hardware graphics
> acceleration that may be available, causing all rendering to be done in
> software. To ensure that windows are redrawn properly, enable the
> generation of expose events on the WIDGET_DRAW window and redraw the
> windows explicitly when an expose event is received.

Not true in my experience, though it depends on the graphics card &
settings (and heaven knows what else). With the not-very-cutting-edge
graphics system on my PC (Windows 2000) here are the settings that I
find the best:

    RENDERER = 0 (hardware renderer--it is a little faster
                  than the software one)

    RETAIN = 2

    EXPOSE_EVENTS = RETAIN lt 2

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: Object Graphics
Posted by marc schellens[1] on Wed, 12 May 2004 07:52:23 GMT
View Forum Message <> Reply to Message

Michael Wallace wrote:
>>> It took a good 30 minutes to figure out how to draw a single diagonal
>>> line.
>>
>>
>>
>> And you saw it on the display!? You are *so* lucky!! :-)
>
>
> Oh, joy. This sounds like I'm going to be in for a fun adventure. But,
> yes, I saw it on the display. So, is seeing something on the display 30
> minutes after you look at your first object graphics code somewhere
> close to a world record?
>
>

>> P.S. That little error means that something is totally
>> screwed up. Maybe you knew that. :-)
>
>
> Mr. Obvious strikes again! :-)
>
>
>> I'd try setting the RETAIN keyword to 2 and see if that helps.
>
>
> Already was. And I can definitely tell you that the Window is not
> retaining anything. Direct graphics windows just like they're supposed
> to in the retain retain department.

Concerning the retaining:
I had a similar problem. From the RSI website I found the
following workaround (could not find the tech tip again):

The problem lies within the open gl hardware rendering.
A workaround is to force IDL into software rendering.
To do this:

cd (as user root) to the rsi subdirectory
(usually /usr/local/rsi/)

form there cd to:
idl_6.0/bin/bin.linux.x86/
(or on non-linux: idl_6.0/bin/bin.OS_NAME.ARCHTECTURE_NAME)

and do a
mv gl_driver.so gl_driver.so.off

after restarting IDL retaining worked again for me.

HDH,
marc

Subject: Re: Object Graphics
Posted by Rick Towler on Wed, 12 May 2004 17:25:45 GMT
View Forum Message <> Reply to Message

"Michael Wallace" wrote...

>
> IDL> window = obj_new('IDLgrWindow')
> % OBJ_NEW: Creation of backing store pixmap failed.
> Requesting backing store from the server.

>
> What does this error mean? For what it's worth, this is on Linux,
> specifically Fedora Core 1.
>

As you have discovered, sometimes IDL can't provide backing store and sometimes X won't do it either. I feel the workaround that Marc posted isn't acceptable if you have a decent graphics adapter under the hood and as David said, retaining an OG window is a waste. Object graphics isn't the easiest tool to work with "interactively". You can do it, but I suggest using something like XOBJVIEW to handle the view setup, windowing and drawing if you are just fooling around.

Set RETAIN=0, keep hardware rendering enabled, and make sure you have the best drivers available for your graphics adapter.

> Secondly, is there something, anything out there which actually explains
> what's going on inside the object graphics system? Or at least does
> better than RSI's documentation? I've worked with graphics systems
> before, but never with something quite like this. It took a good 30
> minutes to figure out how to draw a single diagonal line. Oh, well.

Aside from RSI's literature, the only book I know of is Ronn Kling's "Power Graphics with IDL". This is a good introductory text.

Object graphics is built on the openGL API. If you *really* want to know what's going on inside you can read up on openGL. The "Red Book" is the place to start. An older version, perfectly adequate for these purposes, is available online:

http://www.gamedev.net/download/redbook.pdf

You shouldn't need to go this far though. One of the ideas behind the object graphics system is to provide a very powerful 3d rendering engine that is relatively easy to use so don't get too caught up in the openGL stuff.

FWIW, my tip to object graphics newb's is to use XOBJVIEW. Like I said above, XOBJVIEW will handle the view setup which is often the most confusing part of object graphics. Once you have something showing up in XOBJVIEW then you can move on to your own custom view as needed.

-Rick

Subject: Re: Object Graphics
Posted by Michael Wallace on Wed, 12 May 2004 18:36:32 GMT
View Forum Message <> Reply to Message

>> Secondly, is there something, anything out there which actually explains
>> what's going on inside the object graphics system? Or at least does
>> better than RSI's documentation? I've worked with graphics systems
>> before, but never with something quite like this. It took a good 30
>> minutes to figure out how to draw a single diagonal line. Oh, well.
>
>
> Aside from RSI's literature, the only book I know of is Ronn Kling's "Power
> Graphics with IDL". This is a good introductory text.

I located this book, but I can't seem to find any information on it
other than the book exists and you can buy it. What is the general
community opinion of the book? Do you buy the book because it is the
only one which covers object graphics? Or do you buy it because it is a
very useful companion for the object graphics programmer?

Also, how old is the book? And the related question, how much has
object graphics changed since it was written? Basically, how out of
date is it?

> Object graphics is built on the openGL API. If you *really* want to know
> what's going on inside you can read up on openGL. The "Red Book" is the
> place to start. An older version, perfectly adequate for these purposes, is
> available online:

I've done limited work with OpenGL and some with Java3D (which is Java
wrappers above OpenGL), so I'm familiar with the concepts used in the
graphics. My problem is that I'm having quite a time trying to figure
out how IDL has "wrapped" the OpenGL underneath. The IDL -> OpenGL
mapping is what I meant by wanting to understand "what's going on inside
the object graphics system."

-Mike

Subject: Re: Object Graphics
Posted by David Fanning on Wed, 12 May 2004 19:39:53 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

>> Aside from RSI's literature, the only book I know of is Ronn Kling's "Power
>> Graphics with IDL". This is a good introductory text.
>

> I located this book, but I can't seem to find any information on it
> other than the book exists and you can buy it. What is the general
> community opinion of the book? Do you buy the book because it is the
> only one which covers object graphics? Or do you buy it because it is a
> very useful companion for the object graphics programmer?

I learned how to write object graphics programs the old
fashioned way: I spent weeks and weeks making random
changes to my programs until something--anything--
appeared in my display window, then bootstrapped my
way up from there. I could have really used this book. :-)

This book does not claim to be the final word on object
graphics. It is strictly an introduction to the subject
and it does that very well, I think. I don't know how
anyone learns how to use object graphics from the documentation
RSI provides. You will quickly outgrow this book when
you start writing sophisticated programs, but you will have
saved yourself months of hard slogging and will make
faster progress by having a context for additional learning.

> Also, how old is the book? And the related question, how much has
> object graphics changed since it was written? Basically, how out of
> date is it?

My book has an August 2002 publication date. I don't
know if it has been updated since then. As far as I can
see from leafing through it, everything in it is still
relevant and up-to-date. I get it out every couple of
months or so to look something or other up in it.

> I've done limited work with OpenGL and some with Java3D (which is Java
> wrappers above OpenGL), so I'm familiar with the concepts used in the
> graphics. My problem is that I'm having quite a time trying to figure
> out how IDL has "wrapped" the OpenGL underneath. The IDL -> OpenGL
> mapping is what I meant by wanting to understand "what's going on inside
> the object graphics system."

The book I've found most helpful with respect to understanding
the underlying OpenGL is _Computer Graphics: Using Open GL_ by
F.S. Hill, Jr. Very readable and understandable. And it makes
you appreciate just what RSI has done in bringing us object
graphics. :-)

http://tinyurl.com/3x8u4

Cheers,

David

David Fanning, Ph.D.
Fanning Software Consulting
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object Graphics
Posted by Rick Towler on Wed, 12 May 2004 20:15:38 GMT
View Forum Message <> Reply to Message

"Michael Wallace" wrote ...
>>> Secondly, is there something, anything out there which actually explains
>>> what's going on inside the object graphics system? Or at least does
>>> better than RSI's documentation? I've worked with graphics systems
>>> before, but never with something quite like this. It took a good 30
>>> minutes to figure out how to draw a single diagonal line. Oh, well.
>>
>> Object graphics is built on the openGL API. If you *really* want to
know
>> what's going on inside you can read up on openGL. The "Red Book" is the
>> place to start. An older version, perfectly adequate for these
purposes, is
>> available online:
>
> I've done limited work with OpenGL and some with Java3D (which is Java
> wrappers above OpenGL), so I'm familiar with the concepts used in the
> graphics. My problem is that I'm having quite a time trying to figure
> out how IDL has "wrapped" the OpenGL underneath. The IDL -> OpenGL
> mapping is what I meant by wanting to understand "what's going on inside
> the object graphics system."

I'm not sure I see the value in understanding the IDL -> OpenGL mapping in a general sense. If you have an understanding of openGL you can usually make the connections back to IDL fairly easily. If you don't have the experience with openGL then why bother making the connections? I'm certainly not saying it's worthless to do so, I just don't see the value.

RSI intentionally blurs this relationship as they don't want to associate object graphics with a single graphics API. Ronn's book doesn't spend any time on this either. I think that if you want to know, you'll have to go from the openGL side back to IDL.

-Rick

Subject: Re: Object Graphics

>> I've done limited work with OpenGL and some with Java3D (which is Java
>> wrappers above OpenGL), so I'm familiar with the concepts used in the
>> graphics. My problem is that I'm having quite a time trying to figure
>> out how IDL has "wrapped" the OpenGL underneath. The IDL -> OpenGL
>> mapping is what I meant by wanting to understand "what's going on inside
>> the object graphics system."
>
>
> I'm not sure I see the value in understanding the IDL -> OpenGL mapping in a
> general sense. If you have an understanding of openGL you can usually make
> the connections back to IDL fairly easily. If you don't have the experience
> with openGL then why bother making the connections? I'm certainly not
> saying it's worthless to do so, I just don't see the value.

In a general sense, there is no value. What I'm trying to understand is
where IDL has hidden things. For example, earlier today I was looking
for something that'd write the image in the current window or buffer to
a file. After hunting and hunting without turning up anything, I
started reading through all of the properties and methods one by one
until I happened upon an IDLgrWindow method named Read() which returned
and IDLgrImage. Then after reading through the documentation on
IDLgrImage, I didn't find any methods to write the image to a file. I
finally discovered a property called simply, DATA. I had read past this
property several times since I had no clue what 'DATA' meant. What kind
of data is it? What is it doing there? It turns out DATA contains the
rgb values of the image. Obviously! It's so simple! Everyone should
just automagically know that's what DATA is there for.

When it was all said and done, the code to write the image in the
current window turned out to be pretty simple.

```
grImg = grWin -> Read()
grImg -> GetProperty, DATA = data
write_png, 'image.png', data
```

I see this kind of thing happening again and again. I say "I know how
to do such-and-such in OpenGL. Now how to I do it in IDL?" Search,
search, search, dig, dig, dig. I then say, "Eureka! The information I
need is stored in a very well named property called DATA!"

It's really all the syntax and naming of things that's causing me problems.

> RSI intentionally blurs this relationship as they don't want to associate
> object graphics with a single graphics API. Ronn's book doesn't spend any
> time on this either. I think that if you want to know, you'll have to go
> from the openGL side back to IDL.

Yes, and I completely understand the approach. IDL should be simpler to work with and do a lot for you already. Otherwise there'd be no point. I guess the problem is that I don't yet know what all the classes do, what all the properties are for and how the methods are used. What'd be great would be a cheat sheet of the how to do the basics such as save an image, create standard plots, multiplots, etc. And then a cheat sheet to get you started with the advanced visualization stuff.... I can't wait until I can draw that first 3-D mesh model of an asteroid! (I assume this is possible in IDL -- you can do it in OpenGL. ;-))

-Mike

Subject: Re: Object Graphics
Posted by Karl Schultz on Thu, 13 May 2004 14:17:09 GMT
View Forum Message <> Reply to Message

"Michael Wallace" <mwallace.no.spam@no.spam.swri.edu.invalid> wrote in message news:10a2mljqonpogbb@corp.supernews.com...
> I started playing around with Object Graphics today. I've used Java
> objects in IDL plenty of times, but never IDL's own objects. Anyway,
> the first thing I tried to do was create an IDLgrWindow, however IDL said:
>
> IDL> window = obj_new('IDLgrWindow')
> % OBJ_NEW: Creation of backing store pixmap failed.
> Requesting backing store from the server.
>
> What does this error mean? For what it's worth, this is on Linux,
> specifically Fedora Core 1.

This error is generated when a request to create a pixmap fails. If it had succeeded, IDL would render the image into this backing store pixmap as well as the window, and then when an expose event occurs, IDL copies the backing pixmap into the window.

When IDL can't get a pixmap, it asks the server to perform the backing store duties, which doesn't always work. As pointed out in other posts, it is a little more straightforward to just redraw the scene graph on an expose yourself, especially since you can't completely rely on the backing store. I suppose that a really robust app could implement both techniques and let the user select the RETAIN setting. That way, they'd get the benefit of RETAIN=2 if their system can handle it. (It is pretty easy to do this - just redraw the scene graph on an expose event. If RETAIN=2 is on and working, the expose event never comes and the code to redraw the scene graph just doesn't run.)

I run IDL on FC1 too with a Radeon 9700 Pro. What card do you have?

One interesting thing I noted in the XFree86 start up log is a line that said that the maximum offscreen pixmap size was 1280x400. I found this a little disturbing because the card has 128M of memory. So, if you have the same sort of limitation on your X server and you try to create a reasonably sized window with RETAIN=2, say 500x500, the request to create that pixmap will fail. It will fail even if there is enough pixmap memory but the "shape" isn't right.

This isn't a good thing, and I don't know if this is an XFree86 4.3 characteristic, or something related to the DRI and/or the graphics card drivers. My GUESS is that they earmark a huge part of the vid ram for textures, which leaves this "sliver" of free memory after you assign the rest to the other various buffers (front, back, depth, etc). Further, I guess that the hardware on the card can only blit blocks of memory that have the same "shape" between the onscreen color buffer and the offscreen pixmap area. In other words, pixmaps can't be stored in offscreen memory in a linear, contiguous portion of memory nor in an arrangement that "fits" the unsed parts of memory.

There may be some Xfree86/DRI/Driver configurations settings that control this sort of thing, but I have not found them yet. It would probably take some digging over in XFree86-land to find them.

You also mentioned that Direct Graphics retain settings worked as expected. I'm a little confused by this since Direct Graphics does the same sort of thing for RETAIN=2. Were your DG windows smaller than the OG windows? Another possibility is that the max pixmap size restriction can be eased when there are no OpenGL contexts on the card.

Karl

Subject: Re: Object Graphics
Posted by Michael Wallace on Thu, 13 May 2004 15:10:59 GMT
View Forum Message <> Reply to Message

> I run IDL on FC1 too with a Radeon 9700 Pro. What card do you have?

ATI Radeon Mobility 7500 with the standard XFree86 drivers. I haven't bothered trying out the Gatos drivers (http://gatos.sourceforge.net) yet. Do you think using this might make a difference?

> You also mentioned that Direct Graphics retain settings worked as expected.
> I'm a little confused by this since Direct Graphics does the same sort of
> thing for RETAIN=2. Were your DG windows smaller than the OG windows?

I can create a DG window of any size and it retains properly. No matter what size I use for an OG window, it does not retain. I tried bigger windows, smaller windows and same size windows with the same final result: OG windows never retain despite the value of RETAIN; DG windows always retain properly.

However, I did find the source of my error message. On this particular system, I had edited the Idl resource file to "circumvent" one little peculiarity on this system. While I was at it, I stuck in the following line.

Idl.retain: 2

After removing this line, I no longer saw the error message when doing the obj_new('IDLgrWindow') command. Surprisingly, I didn't see the error with obj_new('IDLgrWindow', RETAIN=2). I thought sure that specifying RETAIN would make the error message come back, but I guess not. Nevertheless, the RETAIN problem still exists.

And for what it's worth, this RETAIN issue isn't that big of a deal for me. Most of the OG stuff I will be doing in the future will mostly likely use IDLgrBuffer anyway! ;-)

-Mike
