Subject: Re: Object Graphics: multiple Views of same model
Posted by karl_schultz on Tue, 18 Dec 2001 19:11:41 GMT

View Forum Message <> Reply to Message

David Fanning <david@dfanning.com> wrote in message
news:<MPG.168901ed44c97ffe9897b0@news.frii.com>...

```
>  Martin Downing (martin.downing@ntlworld.com) writes:
>
>>  Ok, todays object graphics question:
>>  Say you have a 3D object model and you want to view it in 2 or 3 orthogonal
>>  directions as you rotate/manipulate it.
>>  Can this be done with a single object instance? Which we would represent :
>>
>>                      grObject
>>                          |
>>                       IDLgrModel_Group
>>                       |                      |
>>  IDLgrModel_Transform1      IDLgrModel_Transform2
>>                       |                      |
>>                 View1                  View2
>>
>>  I guess not as we now have the reverse of IDLs graphics Hierarchy, and our
>>  model_group is not allowed two parents (poor thing!). However it seems to me
>>  a very reasonable thing to want to do, as a graphics model should be
>>  viewable from multiple positions. I cant see how Scenes or Viewgroups can
>>  help, so is the only way to use one view, switching all its settings and the
>>  top level model transform to look like other views before drawing to the
>>  other windows?
>
>  I didn't mean to insult you the other day, Martin.
>  I know perfectly well you know what you are doing with
>  3D graphics, but sometimes I like to overemphasize the
>  point for our readers. :-)
>
>  If I wanted to see two or more views of the same
>  polygon object, I think I would start by create
>  two or more polygon objects that all shared the
>  same data (with the SHARE_DATA keyword). Each
>  object could go into its own model, each model
>  into its own view, and the views could be
>  collected into a scene, that I would display
>  in my window.
```

I think that a better way is to create a single polygon object and
then add it to multiple models using the ALIAS keyword on
IDLgrModel::Add.

You can also add a *model* to multiple models. So, in this application:

IDLgrModel_Transform1->Add, IDLgrModel_Group
IDLgrModel_Transform2->Add, IDLgrModel_Group, /ALIAS

> This scheme allows you to manipulate the models
> independently to get two or more views of the
> same polygon dataset.

As is the case when using ALIAS.

> I have to admit, I've never had occasion to
> use the shared data trick, but it seems to me
> the application you describe is exactly why
> it is there.

I think that it is not used that often.

SHARE_DATA is intended for when you have a huge set of data that you don't want to duplicate if you use it in multiple graphics objects.
The Object Graphics objects all store a copy of the data (e.g., vertex lists) and SHARE_DATA is a way of easing the costs of storing this data.

There are other uses, like "linking" data across multiple objects, so that if you change one, you change them all. Also, it is useful if you want to use the same vertex list for an IDLgrPolygon and IDLgrPolyline.

SHARE_DATA is a way to solve the problem brought up here, but it is just a way to save some storage while making copies of entire objects.

Also, making copies of objects (with or without the storage-saving benefits of SHARE_DATA) would allow you to change attributes like color so that you get a different color in each view, if that is desirable. Using ALIAS is much more efficient, but won't let you change the attributes used in each view, which I think is the desired effect in this situation.

Hope this helps,
Karl