Subject: Re: Transformation of Objects and Models
Posted by ben.bighair on Mon, 18 Aug 2008 11:55:16 GMT
View Forum Message <> Reply to Message

On Aug 18, 6:37 am, Erik <jansse...@gmail.com> wrote:
> Hi all,
>
> I'm working on a piece of code to make the handling of IDLgr objects a
> lot easier (IDL 6.3). The goal is to easily select some visual objects
> like ROI's, Lines and Text and move / resize or rotate them in the
> drawwidget.
>
> I used the translate/rotate/scale functions of the IDLgrROI / IDLanROI
> object a lot and it does exactly what it's supposed to do. But
> unfortunately the other IDLgr objects (like; IDLgrPolyLine, IDLgrText)
> does not have the transformation functions that the ROI object has :-
> (. For example, to move a polyline, I cannot use the code oLine->Translate, tx, ty. Instead I must retrieve and alter the DATA
>
> property. To move a IDLgrText object, this must be done with the
> LOCATION property... and so forth.
>
> To make things easier I expected the IDLgrModel object to supply the
> solution for me, because the model has the same transformation
> functions as a ROI. At first glance, it seems to work. When I add a
> line to a model and give a translate command, the line get moved as
> expected. Same story for IDLgrText and IDLgrROI objects, so I suppose
> this works for any object that can be added to a model.
>
> My complaint however, is that the actual DATA of the IDLgr Objects
> stays the same! When I move a line to the right on my window, I also
> want the Object's X-data to be changed! It seems like the
> transformation of the Model does not do this :-(.
>
> I can understand if the Model is not meant to change this data, but
> why doesn't have all IDLgr objects the same commands for
> transformation? Now I have to type-check every object and execute
> different commands for each type. Very, very annoying if you ask me...
>

Hello,

I should point out that IDLgrROI actually inherits the the Translate,
Scale and Rotate methods from IDLanROI. Obviously the (good) idea
here was to allow the programmer to work with ROIs without lugging all
the display info around. I like the separation of data manipulation
and data display found in IDL's implementation of ROIs. But, to each

his own...

You could easily create your own graphic atoms (lines, polygons, etc.) that INHERIT from IDLgr* atoms. Then you could institute methods like ::Translate, ::Scale, etc. in your own classes. You can then choose to manipulate the actual data OR manipulate the display of the data OR a combination of both (which may drive you to drink.)

CHeers,
Ben