Subject: Re: Propagating properties
Posted by Struan Gray on Mon, 19 Oct 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Stein Vidar Hagfors Haugan, steinhh@ulrik.uio.no writes:

> 1. During Atom::Init, a call is made to type->register_me,self
> That is, the "atom_type" object is responsible of keeping
> track of all atoms of its own kind. (Believe me, there's
> nothing non-object oriented about this!)
> When you say e.g.,
> silicon->setproperty,color=<blue>
> then silicon immediately tells all the atoms of its own kind
> to make the change in their polygons.
>
> 2. The Atom::Draw method needs to be rewritten, to update the
> color of the polygons based on information from a call to
> self.type->getproperty,color=color
>
> Looking at it after writing it down, these two methods are very
> similar to your own suggestion, just rephrased somewhat...

I think you said it more clearly :-)

Having RTFM'd over the weekend I have discovered that in this case there is a third possibility. Because all the properties I want to propagate are to do with how the atoms are displayed, I can use the 'atom_type' as a symbol to be plotted at each vertex of a 3D polyline object. This will also make simple bonds easy to draw since a single 'IDLgrPolyline' can contain many individual (and disconnected) line segments.

I don't know yet how much memory and processing overhead this will entail, if any, or how it will affect data-picking once I start allowing the user to interact with the model. I particularly want the user to be able to create things like point defects and dislocations by altering and inserting atoms, and that may be easier to do if I take care of all the objects explicitly myself. I'll play around with the various ideas and see which works best.

Struan