Subject: IDL Programming Odditites
Posted by davidf on Mon, 24 Jan 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Hi Folks,

As you know, the best way to learn something new about IDL is to post code to this newsgroup. My latest offering, which I made a week or so ago, is no exception. Here is a preliminary list of some new things I learned. I hasten to point out (in the interest of boosting my Microsoft shares) that FEW of these errors were present in my Windows version of IDL. :-)

1. There is an undocumented minim un window size for an IDL object graphics window on all (?) platforms other than Windows. I'm not sure exactly what that size is, but I do know that the window I was creating in my program was too small. This manifests itself as one or both of these errors:

   Program caused arithmetic error: floating divide by 0.
   Program caused arithmetic error: floating illegal operand.

2. There appears to be a bug in some (all?) X Windows devices that prevents object graphics windows from being resized and then having the view displayed properly:

   theWindow->Dimensions=[event.x, event.y]

   The view is always mapped into the old window dimensions, even if you try to update the view mapping. The only work-around is to destroy the old window object and create a new one in the right size.

3. The very useful RSTRPOS function, which searches a string from back to front, has been made obsolete. In IDL 5.3 you should use the STRPOS function with the REVERSE_SEARCH keyword set.

4. Don't bother trying to set the size of a droplist widget after the widget has been created on X displays. The new value is totally ignored. This is bad news for those of us who like to create a professional looking layout in our widget programs.

5. All bets are off if you are using object graphics windows and hardware rendering on 8-bit display devices. Those of

us who have worked with object graphics for a while have realized from the beginning that all bets are pretty much off on *any* 8-bit device, but this is a warning that this is *especially* true when using hardware rendering (the default setting). I'd say that a good rule of thumb if you want to run object graphics on an 8-bit display is to *always* use software rendering. On my machine with 8-bit color and hardware rendering, gray-scale colors appear as extremely ugly shades of blue. :-(

6. The IDL documentation states that the Z-ordering of objects with the same Z value is in the order in which they are added to the model. While this appears to be true in software rendering, it is decidedly NOT true on every machine in which I have tested hardware rendering. If you will be using hardware rendering (the default setting), you must be careful to offset each object by a small Z value. The easiest way to do this is to put overlapping objects in separate models and translate the models by a small Z value.

I know that at least two or three of us are writing object graphics programs, so I thought I might try to save you some time and aggravation. :-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155
