Subject: Re: BSD problems with Draw
Posted by rylek on Wed, 20 Jun 2007 07:48:59 GMT
View Forum Message <> Reply to Message

Hello there!

Sorry for that, that's one of 'historical anomalies' which I hope will be resolved soon. It all started with me putting the relatively rich set of polyline / polygon drawing functions into Draw. To make this working under X11, which natively are unable to paint polygons with holes (AFAIK), I incorporated the SplitPolygon routine currently sitting in the Geom package. Mirek didn't like that because at the time he was writing a short application in which the executable size was crucial because it had to fit on a 1.4 MB diskette and we agreed to drop the SplitPolygon routine and keep only simple polygons working under X (with Mirek's traditional argument that no one but me uses such a silly thing anyway), to which I agreed because WebMap and other of my drawing-intensive applications access the output device via the Tool objects (see Geom/Draw/Plotter.h) where this drawback could be easily fixed.

Now with things like the new image manipulation system or AGG integration under discussion, it might be the right time to either put the SplitPolygon back or, OTOH, to define (with AGG in mind) a generic mechanism for extending the basic Draw interface. This would allow to keep only the basics in the X11 Draw (in this case the simple DrawPolygon method) whereas the more complicated (implementation-wise) functions would be defined e.g. in a hypothetic GeomDraw extension object. However, this brings up the non-trivial question of true portability of Drawings (in the sense of sequences of drawing operations) among various Draws, typically when printing or generating a metafile.

Maybe the right solution would be to have (as Mirek plans it) opaque DrawData blocks in the Drawing stream which only the right Draw extensions would understand; if the target Draw processing the DrawData block didn't understand a block (i.e. if it were a block generated by another Draw extension), it would somehow have to request the registered extension to emulate the block on its output device. More or less, while Windows HDCDraw would understand the DrawPolyPolyPolygon data block and make direct calls to PolyPolygon API to draw it, X11 Draw wouldn't recognize it and force the DrawPolyPolyPolygon registered operation (resp. the registered multipolygon extension) to emulate the complex polygon with basic Draw operations without extensions. Here the extension would naturally perform the SplitPolygon and call DrawPolygon repeatedly to draw the individual parts.

Regards

Tomas