Subject: Re: High Performance Drawing
Posted by cbpporter on Wed, 02 Jan 2008 17:11:04 GMT
View Forum Message <> Reply to Message

OK, so I made some tests in Delphi and it seems that the current computer I'm using has a very poor graphics card (which theoretically should be good enough for simple bliting, being an ATI 9250) and this is the main reason the performance is so bad.

But still, U++ bliting speed is lower than the one in VCL. I still need to do some benchmarks, but this is certainly an area which could theoretically be improved. I looked over Image, and I saw nothing which looks slower, except the fact that the image is always 32 bits and this could be potentially slow on other bit depths (not the case here).

I also ran an older app of mine which had the same issue and in which I used a a combination of smart bliting and xor updates, which is very fast, but I don't have the sources right now and I don't remember how I done it exactly, but I'm sure I will be able to reproduce it and I hope I can apply it to U++.

Quote:Ok I was further playing and found out that you can use DrawingDraw for much faster performance. Basically this solves the problem imo.
Yes, you a right. This was actually to be expected, seeing how DrawingDraw handles stuff, but I need an image buffer, where I can apply custom effects, so DrawingDraw is not good for my current needs.

On a note to Mirek, ImageDraw is NoCopy, it can not be initialized with an Image and it does not even retrieve the size. This pretty much limits it's use and it is not possible to create a persistent backbuffer.

Anyway, I'll continue to reproduce that drawing method from that Delphi app, which manages to draw a bitmap almost as large as the screen resolution and still do mouse based updates in real time even on a Celeron 900Mhz. I really don't want to introduce a DirectX dependency only for this task.