Subject: Re: Draw::DrawImageOp optimization bug
Posted by Tom1 on Mon, 01 Dec 2008 12:50:54 GMT
View Forum Message <> Reply to Message

luzr wrote on Mon, 01 December 2008 13:57
Just a sidenote: I must see stubborn desperately trying to keep this optimization, but:

- some of my existing applications would have troubles without it, this is essential to e.g. printing .jpges directly to the printer, without the need of rescaling them to 500MB.

- this concept plays an important role in the future, when we will need to print svg pictures. I expect that this will work very well with most bussines oriented graphivs.

I now believe that we can handle artifacts issue, at least on printers.

So the problem now is the color mismatch..

Mirek

No need to apologize, you're the chief -- not me.

I faced the same 'oversized printout' problem with images and went around it using ::StretchDIBits() GDI API. This effectively shrinked the amount of data transferred to the printer. I guess there are very few printers really capable of printing 24 or 32 bit colors at the full resolution so the colors visible to the eye are really sums of many adjacent pixels in both horizontal and vertical directions. Therefore, I use lower resolution for color and gray scale images to be transferred to the printer and print only the vector graphics at the full resolution on top of that.

I'm really dependent on this for many of my applications, but I have been able to do it without modifications U++ itself, so I have not been pressuring you to take it in. If it was to be integrated to U++ Draw::, I would prefer an option flag for natively stretched images, instead of U++ based pre-stretching when expanding of images was needed.

// Tom