Subject: Rendering
Posted by Tom1 on Fri, 24 Feb 2012 13:50:25 GMT
View Forum Message <> Reply to Message

Hi,

I decided to continue the rendering conversation outside the RGBA/Color topic, since this clearly does not belong there.

--

My mind goes frequently around graphics rendering, APIs for that and even dreams of hardware acceleration. Not to mention the 'terrifying' changes in the OS platforms.

I really like to think of U++ as the ultimate abstraction layer to all important platforms and technologies which will protect my SW development investment in the years to come and over the platform changes. I'm thinking of Win32/WinRT change and Wayland coming to Linux as examples of this.

Writing and maintaining applications necessitates that we have fixed interfaces that remain available for the years to come. Employing new technologies (behind the scenes from the application's point of view) requires new abstraction layers and potentially new interfaces to gain access to new features while preserving source compatibility for the existing applications.

I admit I do not understand the U++ rendering infrastructure nearly well enough, and therefore my ideas may be simply stupid.

My point with void Paint(Painter &painter); as an option was just to offer access to new Painter grade rendering features in a way that could allow various backends to implement them efficiently later with different backend technologies and on different platforms -- even accelerated some day... perhaps.

The idea is analogous to what Draw does currently with GDI grade graphics.

Maybe I see the big picture all wrong (it has happened to me before ...)

How do you see the future on this subject?

Best regards,

Tom