Subject: Re: Using Qtf forgets control font
Posted by cbpporter on Sun, 31 Aug 2008 11:39:46 GMT
View Forum Message <> Reply to Message

luzr wrote on Sun, 31 August 2008 11:46Sorry, "FontZoom" is "general term". The methods are The idea behind this is that there is some basic font size. All dialog layouts are designed with this basic font size in mind - and these are units shown in layout designer.

Then there is "GetZoomText" - a standard text whose size is known in basic font size (by basic font we consider here the "normal font" used for most texts in GUI).

Then, when U++ app is started, the same text is measured in host platform "normal GUI font" and this new size / "standard size" is taken as "FontZoom ratio".

This is the best solution I was able to invent... In practice, it works pretty well.

OK, basically you have Csize where you get the current size using StdFont of the zoom text, and Dsize, which is initialized with a hardcoded measurement of the same text and same font under what you consider zoom 1:1 zoom ration. And you use their ration to adjust positions.

Then you must know what which font StdFont() is .

I also noticed that max between Csize and Dsize. This probably explains why when I lowered my PPI and I go really small text, the layout was not shrunk, but when I increase PPI, layouts grow..

Quote:
I had to deprecate ScreenInfo, as it is inherently MT unsafe...

You should be able to get the same thing using

ScreenDraw info(true);

ScreenDraw worked fine. Thank you! Actually, I was getting the assert because I used PromptOK in a Paint method. I shouldn't have done that.

Can I use ScreenDraw to draw on the screen as with Win API GetDC(0)? If yes, then probably I should extract the info I need at application startup and then discard the ScreenDraw instance.

I think I have everything that I need to add HeightPt() and GetHeightPt() to Font.

Also, except font antialiasing, is there any other missing tag from Qtf that is present in a font description?