Subject: Re: 16 bits wchar
Posted by cbpporter on Sun, 03 Aug 2008 12:51:49 GMT
View Forum Message <> Reply to Message

I've finished Label too and that's about it for the immediate support that I need for CJK. I fixed the previous problem with characters not being drawn by using a hardcoded font name for those problematic ranges. I guess Windows font support is not perfect either .A better solution would be to determine if the font can display the character, and if not, change the font by probably using a list determined at application startup. But I'm afraid that it isn't that simple to do with current font rendering methods and we should get back to it at the next text output engine refactoring (maybe when we do it for Linux, where it is more needed).

I only had to update a couple of U++ functions, and I wrote different encoding conversion functions which I explicitly call instead of the standard ones to limit my changes to specific parts of code and let the rest use the defaults.

I will probably need an edit control updated also, but for now I'm pretty happy with my over 13000 unique characters displayed, so full JIS support.

BTW, the Core2000 font available on the Internet has a number of broken codepoints, drawing the wrong characters in several cases. It is pretty hard to notice unless you know what to look for, so if anybody is using it, try out "HAN NOM A" instead, which hasn't shown any error up to now.

The question is what now. Since I'm happy with my fixes and nobody else seems to have needs regarding CJK support, I could just rename the couple of functions I modified and override Paint in a control that inherits from Label and thus keep my changes local and become U++ version agnostic. Of course, I will release a package in Bazaar for those who for some particular reason need more than Unicode 1.1 support, but a fair warning is due: my changes are strongly biased towards Japanese characters, so Chinese or Korean specific issues might still exist.

Or I could merge my changes with my installed version of U++, use it for a while to see if there are other problems (Qtf and edit controls are sure to not enjoy surrogate pairs) and continue researching how to best migrate U++ entirely to the new scheme. I'm only going to do this if you want these changes and if you want them relatively soon, i.e. in 1-2 devs. If not, I'll go with variant one because I still need to implement EUC-JP encoding support, for which I need huge conversion tables .