Subject: Re: Porting SystemDraw to Frambuffer
Posted by mirek on Wed, 16 Dec 2009 13:37:37 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Wed, 16 December 2009 05:19
where the BufferPainter can be initialized with an ImageBuffer
which in turn should somehow represent the /dev/fb0.
-->problem: the fb0 can have really weired dimensions, bitwisths per color, strides and the like..is the ImageBuffer capable to cope with that?

No.

Quote:
or do we need to have the double buffering as only solution (where the normal ImageBuffer 24bit is overlayed/calculated on tob of the weired fb0 layout)?

Yes. BTW, ImageBuffer is 32bit RGBA.

Quote:
we need to provide a full implementation of own drag and drop, which can be significantly simpler i think, because we dont drag between multiple windows od Win32/X11, but simply inside the TopWindow (which is between Ctrls)

we need to provide a full replacement for Clipboard stuff, how does this work?

IMO, it is in fact much easier to start from the scratch than implementing Clips in X11/Win32.

There is nothing magic, you have raw binary data and string that represents the type of data. Perhaps use filesystem to store both and you are done... (with clipboard anyway, D&D will need a bit more).

Quote:
we need to provide a full replacement of the TimerThread (which is used heavily by Ctrls, it also needs to be reentrant (means a new occurance of timer event is handled either by interrupting a possibly currently still working timerthread procedure, or by spawning another thread to handle this, which leads either to application lock (due to repeated reentrance) or to thread explosion if handling callbacks are misdimensioned somehow.

There is no TimerThread in CtrlCore And thing is pretty simple generally in fact. All you need to do is limit waiting for input events to some max timeout in message loop and to know current system time (one way or another). Then call TimerProc at the end of event processing or if there is timeout.

Quote:

is there any description (besides the code itself) available in terms of how Clip, Drag and Drop, the timer thread and the GUI thread are set up (basic ideas should be enough)

I hope that above is helpful for starters, ask for more details as you go.

Quote:
the GUI flag, which is used currently can be abandoned maybe, because it is evaluated only in POSIX environment, in WIN32 GUI is somewhat implicit, as soon as one uses the package CtrlCore, one uses gui stuff in Win32.

GUI is because Win32 - console vs GUI apps have different linker options - that is the original problem.

Quote:
in Linux, one should be able to specify GUI X11 or GUI FB somehow.. but this is small peanuts. backwards compatibility can be done imlicitly by defining a standard flag.

Well, that is interesting problem I guess GUI FB and X11 as default or something like that.

Quote:
PS: maybe we can really use this effort to restruct code to make porting in future a lot esier.

Well, it is not so messy, but it is true that over those 10 years of development, thing are getting more and more patched.

Anyway, in reality, I am not huge believer of fresh starts there. If nothing else, it would put all of my apps in jeopardy...

Also, it is questionable that you can gain too much more clarity there. Some of stuff in CtrlCore is pretty nasty, but that is partly because host-platform details are pretty nasty too...

Mirek