Subject: Re: Porting (Mac OS X) and "reference application" idea
Posted by daveremba on Tue, 12 Jul 2011 22:15:54 GMT
View Forum Message <> Reply to Message

OK, your messages make it very clear & helpful.
No high level widgets.
I understand that in summary,
we're aiming for a low level port
at the System Draw level, etc.
(like from Windows to Linux).

Quote:Actually, I do not think so. There is only one possible path and it is the same as current Linux and Windows backends...

Well within that path,
I am thinking of looking into the following
sub-paths:

1) X11 approach - develop in GCC
continue to use X11 code for drawing,
fix the observed artifacts (flat buttons and
uninitialized areas behind menus), and
add the needed code to get top windows recognized
by MacOS as true application processes
(most work appears to be done
since UPP already has X11/Linux done, but
X11 can be complex and emulator on
MacOS may not be 100% compliant)

2) OpenGL approach - develop in GCC
implement low level drawing code using OpenGL,
and use a simple library like GLUT for window
system, keybd, and mouse, events
(simple and high performance, but not sure
if GLUT behaves well with MacOS)
This is the approach used by:
http://www.openframeworks.cc/about

3) Objective-C++ approach - develop in Xcode
implement low level drawing, window, and
event code using Obj-C calls into MacOS (probably low level
Cocoa calls) hooked into SystemDraw
and other UPP libs.
(probably the most work, but maybe the
best visual result, and flexible towards
more compatbility with Apple in the future).

4) do whatever they do approach - develop in GCC
continue to look at other frameworks (wxWidgets, Firefox,
Qt, FLTK, Tcl/Tk) and see what they do;
borrow from their code libraries if possible.
wxWidgets uses approach (3) above

Here is an interesting comparison of
UI toolkits (from a wx perspective):
http://wiki.wxwidgets.org/WxWidgets_Compared_To_Other_Toolki ts
wxWidgets has a pretty clear Cocoa library
in their source tree; unlike Firefox it
builds standalone apps rather hooked into
the Gecko framework (browser).

FLTK has an architecture that is most similar
to UPP - only a small portion of the code is
platform specific. It draws its own
widgets using low level 2D drawing primitives
in OpenGL (as in approach (2) above).
Pros: it looks the same on all platforms,
Cons: it never looks like other native apps
on any platform.
This code may be helpful to fix problems
in approach (1 & 2) above.
Source code is here:
http://www.fltk.org/software.php?VERSION=1.3.0&FILE=fltk /1.3.0/fltk-1.3.0-source.tar.gz

Firefox source is browsable here:
(to see how they do things, as you suggested Mirek)
http://mxr.mozilla.org/firefox/source/

The Firefox to Cocoa code is here:
http://mxr.mozilla.org/firefox/source/widget/src/cocoa/

I will do some tests/learning, and
report back in a few days, and
I will see how far the previous Mac
port effort got.

It seems that getting apps to work in the
X11 emulator is still worthwhile as the
easiest first step. (approach 1)

I spoke to my project client and they also do
want a working MacOS front end as well.

-Dave