Subject: Re: best way to draw text/fonts on MacOS from C/C++
Posted by daveremba on Wed, 27 Jul 2011 19:31:27 GMT
View Forum Message <> Reply to Message

Quote:Could we do without nib eventually?
Yes, but maybe 2000 lines of code is needed:
(comparing to FLTK)

If there is no nib file then it seems one
must create their own top level window and
process event handling (using NS... obj-C API),
and draw text and/or geometry (using CG... and CT...
C++ APIs) at minimum.

Rather than end up writing a stand alone app
(even using theide), I think it makes sense to either
a) fit new MacOSX code into the Rainbow framework,
or b) connect the UPP low level drawing code
using CtrlCore mods.

probably option a is the recommended path.

Do the test and example applications work
for Rainbow with UPP on Windows?

I'd have to guess at what UPP should do with
some of these window & app events, maybe using
other UI kits as a guide.

MacOSX question (for me): what is difference
between a "key" window, "main" window,
and an "active" window?
what's the correct way to send the events into UPP?

from fl_cocoa.mm
window & application events:
- (void)windowDidMove:(NSNotification *)notif;
- (void)windowDidResize:(NSNotification *)notif;
- (void)windowDidResignKey:(NSNotification *)notif;
- (void)windowDidBecomeKey:(NSNotification *)notif;
- (void)windowDidBecomeMain:(NSNotification *)notif;
- (void)windowDidDeminiaturize:(NSNotification *)notif;
- (void)windowDidMiniaturize:(NSNotification *)notif;
- (void)windowWillClose:(NSNotification *)notif;

- (void)anywindowwillclosenotif:(NSNotification *)notif;
- (NSApplicationTerminateReply)applicationShouldTerminate:(NSApplication*)sender;
- (void)applicationDidBecomeActive:(NSNotification *)notify;
- (void)applicationWillResignActive:(NSNotification *)notify;
- (void)applicationWillHide:(NSNotification *)notify;
- (void)applicationWillUnhide:(NSNotification *)notify;
- (id)windowWillReturnFieldEditor:(NSWindow *)sender toObject:(id)client;

device events:
+ (void)prepareEtext:(NSString*)aString;
- (id)init;
- (void)drawRect:(NSRect)rect;
- (BOOL)acceptsFirstResponder;
- (BOOL)acceptsFirstMouse:(NSEvent*)theEvent;
- (BOOL)performKeyEquivalent:(NSEvent*)theEvent;
- (void)mouseUp:(NSEvent *)theEvent;
- (void)rightMouseUp:(NSEvent *)theEvent;
- (void)otherMouseUp:(NSEvent *)theEvent;
- (void)mouseDown:(NSEvent *)theEvent;
- (void)rightMouseDown:(NSEvent *)theEvent;
- (void)otherMouseDown:(NSEvent *)theEvent;
- (void)mouseMoved:(NSEvent *)theEvent;
- (void)mouseDragged:(NSEvent *)theEvent;
- (void)rightMouseDragged:(NSEvent *)theEvent;
- (void)otherMouseDragged:(NSEvent *)theEvent;
- (void)scrollWheel:(NSEvent *)theEvent;
- (BOOL)handleKeyDown:(NSEvent *)theEvent;
- (void)keyDown:(NSEvent *)theEvent;
- (void)keyUp:(NSEvent *)theEvent;
- (void)flagsChanged:(NSEvent *)theEvent;
- (NSDragOperation)draggingEntered:(id < NSDraggingInfo >)sender;
- (NSDragOperation)draggingUpdated:(id < NSDraggingInfo >)sender;
- (BOOL)performDragOperation:(id <NSDraggingInfo>)sender;
- (void)draggingExited:(id < NSDraggingInfo >)sender;
- (NSDragOperation)draggingSourceOperationMaskForLocal:(BOOL)isLocal;

Dave
