Subject: Re: WinAPI UNICODE question
Posted by cbpporter on Sat, 22 Nov 2008 10:36:02 GMT
View Forum Message <> Reply to Message

OK, so I was about 20% through Draw and I was starting to feel the good old "this is getting really old really fast". For every WinAPI reference I have to replace the type with the underlying one, both in header and C++, and often I have to insert harmless but ugly casts. Following this method I can't see myself finishing this, not to speak about merging with SVN anytime someone commits. Auto merge is pretty smart, but it has it's limitations.

So I need a better solution and I found one: a variant of the pimpl idiom. All Hxxxxx WinAPI types are pointers, and they receive their memory from some source (so it bypasses theproblem with pimp: the need to allocate extra memory), so a forward declaration in in function declaration is more than sufficient. So all I have to do is insert a forward struct and a typedef in Core.h somewhere early, and thanks to the very permissive forward declaration and typedef rules provided by the C++ standard (stuff which I usually would consider better not to compile), I can leave pretty much all U++ header and .cpp files unmodified. This will make merges a lot easier.

So ignore all my previous suggestion (except the MACROs from WinAPI). I think this is the right way to handle this. Very few changes to existing code base, yet still platform independent interfaces.

It has the slight disadvantage that something like HINSTANCE h; will compile anywhere, but it's just a pointer and you can't really introduce platform dependency with an opaque pointer, unless you really want to.

It even maintains the 15% compilation time decrease under MSC when compiling Core tutorials. I need to get more packages done before I can say how this percent will change when compiling larger code base.

Now I have to undo my changes and experiment with this approach. If somebody is interested, I can post here a Win32 independent version of Core once I consider it stable, and later Draw.