Subject: Re: U++ talk
Posted by amrein on Wed, 03 Sep 2008 16:52:47 GMT
View Forum Message <> Reply to Message

Quote:
OK, now I see where are you heading. Anyway, this is even more complex that what we wanted to do. And, let me say, perhaps a little bit more risky too.

Yes.
I don't know how to generate complex doxygen doc without including the documentation back into the code.

Quote:
OTOH, it appears that you would like to do some work on .so etc.

Maybe, as part of this process, you can do exactly this.

I mean, it appears that ".so" work will have to somehow take existing uppsrc and via some nice tool transform it to .so (or, rather, something with makefile that produces .so).

The T++ contains enough information (ok, barely enough, but it is manageable) to put the comments back to the code even now. So do it!

Makefile already done, just for testing. I modified the upp-2008.1 Makefile to see how much disk space a lib.so could take.

At present, I'm trying to learn U++. Doc is not enough to be a good U++ programmer. I really need to build a few small applications (statically linked). I'm still a real noob in U++.

When I will feel ready, I will work on TheIDE source code to generate Makefile able to produce dynamic libraries from uppsrc packages and certainly also on U++ missing documentation.

Quote:
Or, maybe, you can export doxygen to separate files, that is even easier.

BTW, to put things into perspective and understand each other, if you are speaking about Doxygen, I was thinking about putting docs into sources, that is why confusion. Exporting Doxygen formatted docs from T++ is IMO completely unrelated topic and perhaps a good idea, if it can be done with relatively low costs.

Mirek

I think you mean Topic++ will still use external files but TheIDE will be able to show them directly side by side in the code editor, right?

Quote:
There is no way I as a normal user am installing MySQL, Sqlite3 and the other client so that I can install a .so for U++.

No need to. Same goes for Qt, wxWindows, Gnomedb, Kde, Gambas, ...

Quote:
On possible layout would be: one for Core and other non GUI stuff, one For CtrLib and all control and draw, and one for each SQL dialect.

The idea in the sent list (other thread) was to have several libs. Each directory in level 3 could be a library:

./source/upp/core
./source/upp/gui/core/x11
./source/upp/gui/core/win
./source/upp/gui/core/osx
./source/upp/gui/core/directfb
./source/upp/gui/library
./source/upp/gui/dialogs/x11
./source/upp/gui/dialogs/win
./source/upp/gui/dialogs/osx
./source/upp/network
./source/upp/database
./source/upp/database/oracle
./source/upp/database/odbc
./source/upp/database/sqlite
./source/upp/database/mysql
./source/upp/database/postgresql
./source/upp/dbus
./source/upp/opengl
./source/upp/xml
./source/upp/sound
./source/upp/video
./source/upp/html
./source/upp/script
./source/upp/richtext
./source/theide/...
./source/unstable/my_unstable_ide/
(...)

Core, Gui, Network, Database, Opengl, XML could be dynamic libraries. Upp could be an all in one library too.

How to have X11 or directFb back end is a good question. Build several libraries like wxWindows: gui-x11, gui-directfb, gui-gtk? Other solution: gui-core load its screen plugin (x11, directfb, linuxfb, javascript, ...) at runtime.

QtEmbedded does this for Qtopia: if you don't provide a special switch the lib try to connect to the Linux framebuffer, if not it connect to the framebuffer simulator. At compile time, you can remove the ability to connect to different backend and select the main one.