Subject: Re: User lists of "bad" naming of classes, functions etc in U++...
Posted by amrein on Fri, 29 Aug 2008 21:52:14 GMT
View Forum Message <> Reply to Message

Quote:
Quote:
- All files and directories in U++ should use lower-case names because of portability issue between Windows and Linux (Linux files names are case sensitive).

Well, we had some minor problems in the past with this, but in reality, it is not an issue.

For the U++ team perhaps. For a large project with hundred header file, the portability is lost and you need to edit your .cpp + rename a few .h (need to resync .cpp #include statements with your .h names).

Without a strict layout, if I get a package from someone using Windows only and want to compile it to build a rpm or a deb package, I will need to add fix just for this. And for each release, I will have to update those patches as soon as something has changed in one of those #include statements or if there are new .h.

Quote:
Quote:
- "Assembly" "Nest" and "Package" should be just "Collection" "Directory" "Package". "Assembly" and "Nest" words are complicated and should die. I don't hate birds but I'm still confused by those names

Hehe, in fact, the only one I suggested to rename is "Package". IMO, the better name would be "Unit"

Where is my gun to shot myself? LOL.

You mean, you would like it to be more complicated? Please don't. I beg you. Really. Don't over complicate Ultimate++.

Your toolkit, your baby, is the best toolkit I ever saw. For me, It could replace GTK, QT, wxWindows... It's simple. It just needs to be simpler, not the contrary. It gave me back the desire to code.

I'm not a company. I'm alone. I'm in vacation. I'm just trying to do what I like to do. I'm not an everyday programmer. Programming is just an hobby. My company do not sell software. They are not around applications and libraries. They use mainly PHP/Javascript/MySQL/Linux for servers and Linux+Windows for clients. That's all. No need for U++.

What I mean is please, don't do this error. I know we are not lemmings (I love this game) but

really, you shouldn't go this way, but the other one, the simplification one.

Quote:
As for "Nest" vs "Directory", I am afraid that it would clash with normal directories, leading to various mistunderstanging. E.g. "Package is directory. Packages as grouped in Directory".

But you are not the first one to complain about it, so we might still find a better name.

You have the same problem with the old naming. That won't change.

When someone says "A package", it's like opening an HTML file. Anyone understand "open a specific file in a directory. This file contain all information about how to link all together and how to build and show the final target".

Quote:
Quote:
- First TheIDE window (when you open TheIDE). It shows two rows "Assembly" and "Packages". Should be "Directories" and "Packages" and the main title should be "Select main package from "Default Collection".

Ehm, what is "Default Collection"?

The one TheIDE should use the first time you open it.
The idea of Collection is to break "Assembly", "Nest" and "Packages" into "Collections", "Directories" and "Packages".

When you open a "Directory", you get all Packages there. When you open a "Collection", you get all "Directories" with "Packages" you would like to works together.

Example: you want to create "TheIDE 2.0". You don't want to mess with the old TheIDE directory. You create a new Collection and you add all directories except the old TheIDE one. That way, your new TheIDE won't conflict with the old one and you will still be able to get all other packages functionalities.

Quote:
Quote:
- All external dependencies from thirds party, those not created by U++ team but distributed with U++, should be in a separated directory "3rdparty" for easy directory understanding and easy license attribution.

They are separated in "plugin". OK, over time, the problem appeared that some "inhouse" packages got there too...

Now I understand why I couldn't appreciate the main purpose of this directory.

Quote:
Quote:
- RichText is not RichTextFormat (RTF from MS). I was confused first when I saw RichText. Should be replaced by HTML implementation or perhaps real RTF or OASIS (OpenDocument file format from OpenOffice also known as ODF).

RichText is just data container for rich texts. It can be now created (imported, loaded) from RTF or QTF. I would be happy to have HTML and ODF (and MS Office) support as well, but it is a lot of work - that is the real constraint.

Note that QTF was created because HTML lacks true typography and other formats are too hard to be coded manually. You certainly would not want to create text in message boxes in ODF....

Qt use HTML tag in their help. A few years ago, I translated all Qtopia 1.7.x + OPIE help and applications.

Quote:
Quote:
- Topic++ and other tools like that should have their own .exe. At least, for big enterprise or just contributors wanting to just create documentation, translation, ... or people wanting to use "make" without calling TheIDE each time.

The main beauty of Topic++ will be the tight integration with the code. I cannot see how to do that without having it in TheIDE.

One thing I really wanted to have (and it was told several times) is non-GUI "umk".

Let me explain what I think is one of the most missing application for programmers. An application answering those questions:
- How can I have the documentation directly into my code
- How can I see where are undocumented classes or functions
- How can I read good documentation about the API without messing with the entire source code

Doxygen gives a part of the answer. Topic++ gives another part.
My idea is to have Topic++ for internal code documentation (but IN the .h/.cpp) and use doxygen to create the result.

Quote:
Quote:
- In TheIDE, the main window, just after the first one (i.e "Main package selection"). In the top left frame, main packages dependencies should be separated from main package with a clickable

borderless button in grey saying "Dependencies list" (click on it, and select other dependencies).

That is fine suggestion. OTOH, I do not see this as critical feature, in fact I think that for complex overview and dependencies management, users should learn to invoke Package organizer...
