Subject: Re: Ideas on U++ as library
Posted by mirek on Sun, 20 Dec 2015 10:40:13 GMT
View Forum Message <> Reply to Message

Well, I suggest starting with allocator....

It is the least controversial thing and relatively simple to 'extract'.

Technically, there is one problem though to be resolved: When thread exits, heap cleanup function has to be called.

That was easy to do in U++, where we have our own Thread class, but will need to be resolved for standalone allocator.

It looks like pthread_key_create resolves issue in Posix (maybe...), but I do not see equivalent in Win32 :(Worst scenario - it has to be documented.... :)

Other than that, the endproduct that I would like to produce here is single 'amalgamated' .c file (like sqlite3). You include it in .cpp or add to project and you get your application running faster, using less memory. Or include in .c and get 'super_malloc', 'super_free' functions. With something #define JUST_DECLARATIONS, you can also include it in header.

So the task would be mostly about

- convert current code in C++ to be .c compilable. Tedious, but possible
- in the process, add more comments, including algorithm overview
- create 'amalgamating export' utility that will take current Core and export amalgamated allocator code as single .c file (or maybe some other extension)
- create website section for it, publish :)

Mirek