Subject: Re: New packages announcement
Posted by piotr5 on Sun, 14 Jun 2009 19:53:09 GMT
View Forum Message <> Reply to Message

I think the c-extension is a bad idea. there are a lot of things possible in c, from objects and abstract interfaces, right down to iterators and whatever programming-paradigm. what we really need is a replacement for bison and yacc. what do you plan to put into your c-language, cbpporter? what do you feel is missing in c? as I am saying, bison or yacc are supposed to create c-files, and a slow incremental non-breaking development would fit those tools better than any c-slang. whenever some file or script needs to be parsed, there is a major problem with redundancy within the sources of a parsing-program. solve that! c is really not missing anything (except maybe for type-checking, but his would break code and is already covered by c++). please show me that I am wrong. from the point of view of assembler there are several things one could add to c. for example an inlined 2-log realized in assembler would be nice -- unfortunately not every machine-language has the fitting op-code. and what I really miss about c is a way to tell the compiler how it should optimize my code. for example if the compiler doesn't know of the low-level load-zero optimization (i.e. whenever a register has to contain a zero-value then it's faster to xor the register with itself), if it really does load an explicit "0" from memory to the registers whenever it is asked to, then there is no possibility in c to tell the compiler to apply that optimization! in such a case one is forced to switch to a different compiler as the c-standard has no possibility of achieving an influence on pre-processing and post-processing of the program. I'm just not sure how such a intrusion into the compiler's competences could be implemented...

as for tcc, it's really a good idea. a good application would be to create a graphical front-end to a c-interpreter. preferably in the style of upp with the possibility to browse the various libraries and headers and maybe occasional sources of them, and to look up things in various documentations. if only I would understand upp-sources better, I could transform it into such a beast. instead of source-packages there would be library-packages and program-interfaces (for already running programs, in the style of a debugger). still, I'm not ready yet, and I don't have enough time. I really can't help now...