Subject: Re: User lists of "bad" naming of classes, functions etc in U++...
Posted by Coder on Sun, 31 Dec 2006 03:20:40 GMT
View Forum Message <> Reply to Message

From my experience working on large programming teams, having the variable scope prefixed to variables name helps make code a lot easier to read, and allows for different scopes to use the same name variable (less variable name conflicts). It helps in readability in that in looking at a piece of code, you don't have to search for the declaration of a variable in order to determine if it is a local, member, inherited member, static or global variable, and also in trying to understand how a member is used and doing a search you don't accidentally come across a local variable named the same as a member. The ones used in Hungarian Notation are s_* (s_Var) for static variables, g_* (g_Var) for global variables, m_* (m_Var) for class/struct member variables. Since member variables are most often used, sometimes I've seen code where they don't use an underscore, m* (mVar), or they only use underscore _* (_Var). I've also seen some use a_* for arguement variables, this is also useful in reducing variable name conflicts. Also, in general classes/structs with no or few functions and mostly public data, don't have to use scope prefixes.

I think type prefixes are less important, but can reduce variable name conflicts in some cases, or (if there is no quick help) it allows others to quickly understand what a variable is with out looking up it's declaration.

To Werner,
That is useful but not in a way that addresses much of the above. It still requires looking things up, and the information it returns has type and global, instance, class symbol. It doesn't appear to work for local variables. Plus it adds additional information that is kind of confusing about its locality.

There are similar things you can have in Visual Studio, one of them you hover the mouse and it tells you the type and what it is a member variable (if it is a member at all).

But really nothing beats being able to look at variables and know their scope, with out having to look it up. It may also be useful if there were some how a way to color code variables based upon their scope. But you still run into one of the other issues, the variable name conflicts.