Subject: Re: Should the pick semantics be changed?
Posted by piotr5 on Fri, 07 Mar 2014 15:23:50 GMT
View Forum Message <> Reply to Message

I agree, imho this is a bug in gcc (or maybe also in c++11 standard): every return statement should pass the arguments as rvalue to the return-type constructor since the return statement is leaving scope and thereby its parameters have the same properties as all implicite rvalue objects: unlikely to remain (unless static), should not be altered except for purposes of picking, definitely a temporary value.

as for pick and clone I am unsure if we need both. this is the actual question of this thread: which of the two should we drop? in std-c++11 clone is implicitely done, always the default. in u++ the default is picking and clone would need to be stated. a good choice since picking is more efficient.

looking at this new idea with clone() I guess my fears about diverging from the standard has been soothed: if I want to use std::vector as drop-in replacement for Upp::Vector, I could specialize a clone() function for it. so I would be happy to see a clone function in Core, maybe already specialized for std library containers.

btw, I heard that according to the standard compilers are free to remove calls to the initializer of an object, so also from that side pick-construction can be realized. I haven't observed that behaviour though...