Subject: Re: Should the pick semantics be changed?
Posted by mirek on Fri, 07 Mar 2014 15:44:44 GMT
View Forum Message <> Reply to Message

piotr5 wrote on Fri, 07 March 2014 10:23I 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.

I am not so sure about this. I would say that what compiler requires here is a copy from lvalue to temporary object...

Quote:
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.

Well, I got interested in this and after about 4 hours I have ide running with new notation. It is committed into branches/cpp11...

That being done, I would say that it is actually kind of nice to see all places where picking happens. Also, valid argument about unexpected behaviour (at least for beginners) ends with this.

Also, without mandatory 'pick', you need a lot of of constructors/operators= in the class. With 'pick', you need just
2 (or 3 if you want to support clone).

More good news: I am 90% sure that I can make clone and pick work in c++03: that would mean that we can have common sources for both, with the only difference that when compiled with C++11 active, you get things enforced.