Subject: Re: C++11
Posted by dolik.rce on Sun, 02 Dec 2012 15:04:20 GMT
View Forum Message <> Reply to Message

Lance wrote on Sun, 02 December 2012 15:08I think eventually U++ should get rid of pick_ and make use of rvalue reference instead. rvalue reference solves the same problem pick_ saught to solve and is standard compliant, and behaves more consistent across compilers: I believe pick_ is #define'd to different things on MSVC from on g++, and to avoid conflicts, U++ has to introduce a dummy parameter for deep copy semantics.
Hi Lance,

If I understand both picking and rvalue references correctly, there is still a very valid reason to keep using picking: Rvalue references work ONLY on temporary objects. In U++, you can use picking on any object and even reuse it, assuming you clean it up after it has been picked, typically by calling Clear() or by assigning content from another object. I think this can not be done simply with rvalues.

Also, I believe that introduction of move semantics to C++11 standard will actually make it easier to explain U++ pick to new programmers, because they will already be familiar with the concept. We should just adapt the introductory documentation to explain how pick constructor is similar to move constructor, what is different and how U++ further extends this concept.

Honza

EDIT: After bit more reading, I found you can actually make it work with any object, using std::move... Right now, that seems downright ugly and hackish to me I guess I need to read even more before I can make my mind whether the switch to move semantics is a good or bad idea.