Subject: How would you design a good copy/move semantics system?
Posted by cbpporter on Mon, 15 Dec 2014 14:08:38 GMT
View Forum Message <> Reply to Message

This is more of a question for people deeply familiar to the way copy-constructing works.

I was thinking of a system where:
  - each class can have a copy and a move optionally
  - copy works like the default copy constructor, except for classes where deep copies are needed
  - move works pretty much the way it works in U++, but only destroys data that would be copied by a deep copy
  - calling move on a class that does not have a move implemented will do a copy
  - the rules apply based on class depth

This are the principles. I need to also do an implementation that has as low overhead as possible performance wise and that does not look particularly ugly.