Subject: Re: Problem with Vector::Add (pick/clone semantics)
Posted by Novo on Sat, 10 Aug 2019 04:33:51 GMT
View Forum Message <> Reply to Message

Actually, it is possible to move data from std::initializer_list with a little hack:

```
template <typename T>
struct Foo {
 Foo(std::initializer_list<T> init) {
  for(const T& i : init)
   v.Add(pick(const_cast<T&>(i)));
 }
// Foo(std::initializer_list<T> init) {
//  for(const T& i : init)
//   v.Add(i);
// }

 Vector<T> v;
};

struct Boo : Moveable<Boo> {
 Boo() {}
 Boo(const Boo&) = default;
 Boo(Boo&&) = delete;
};

CONSOLE_APP_MAIN
{
 Foo<Vector<int>> f = {{1}};
// Foo<Boo> f = {Boo()};
}
```

The problem is that this will require all types to have a move constructor.
A move constructor can be detected via std::is_move_constructible, but I couldn't figure out how to apply SFINAE to a constructor.

IMHO, all this code complexity is unnecessary in this case.