Subject: Re: Problem with Vector::Add (pick/clone semantics)
Posted by Novo on Fri, 09 Aug 2019 18:12:20 GMT
View Forum Message <> Reply to Message

I personally would say that this is not a bug. This is a feature. :)
clone was intentionally removed from Add to prevent implicit cloning.
Basically, std::initializer_list will create a temporary const object and after that it will force you to create another copy of it. This is an unnecessary allocation.
U++ is warning you about that and offering you other tools like
```
 VectorMap<String, Vector<String>> MY_MAP;
 MY_MAP.Add("s1", Vector<String>{"s11", "s12", "s13", "s14"});
```

In this case objects will be moved.

Ideally, it would be great to have a set of overloaded operators VectorMap& operator()(const K& k, const T& v)

More details on this problem can be found here.
A comment to this article has an interesting code snippet:
```
template<std::size_t N>
Vec(T(&&a)[N])
  : _vect(std::make_move_iterator(std::begin(a)), std::make_move_iterator(std::end(a)))
{}
```
Extra braces needed though, but somebody may find this more idiomatic:
```
Vec<int> v {{1, 2}};
```