Subject: Map with unique keys and transfer semantics?
Posted by Infausto on Fri, 13 May 2016 03:09:32 GMT
View Forum Message <> Reply to Message

Looking at docs i cannot find any map container that allows only unique keys and have transfer semantics.

ValueMap supports replace of value by key lockup, but it has no transfers semantics.

OTOH VectorMap don't support replacing value by key lockup, but has transfers sematincs.

Any hybrid map that supports this two requirements?

Thanks in advance.

Subject: Re: Map with unique keys and transfer semantics?
Posted by Lance on Sat, 14 May 2016 03:28:30 GMT
View Forum Message <> Reply to Message

If VectorMap otherwise satisfy your need, I would use VectorMap as it appears to be more economic. You have control over the VectorMap object, so you can enforce uniqueness by yourself. If its going to be exposed to somebody else, enclose it in a class and expose Add method that would either replace existing value (value as in key-value pair) or throw or return false...

The fact VectorMap allow multiple record to have the same key (kind of like std::multimap) should not prevent you from enforcing uniqueness as you have full control over its object.

Just my 2 cents.

Subject: Re: Map with unique keys and transfer semantics?
Posted by Lance on Sat, 14 May 2016 04:26:23 GMT
View Forum Message <> Reply to Message

for an idea of what I explained

```
#include <Core/Core.h>

using namespace Upp;

template <class K, class T>
struct UMap : VectorMap<K, T>{
 T&      Add(const K& k, const T& x)          { return value[FindAdd(k)]=x; }
 T&      AddPick(const K& k, T rval_ x)       { return value[FindAdd(k)]=pick(x); }
```

```
 T&      Add(const K& k)                 { return value[FindAdd(k)]=K(); }
};

CONSOLE_APP_MAIN
{
 UMap<int, String> map;
 String s="Hello, world!";


 map.Add(1)="Hello";
 DUMPC(map);
 // unfortunately String doens't have move assignment(xfer) operator
 // so the following downgraded to Add
 map.AddPick(1, pick(s));
 DUMPC(map);
 map.Add(1,"Unique Vector Map");
 DUMPC(map);
}
```

Subject: Re: Map with unique keys and transfer semantics?
Posted by Infausto on Sun, 15 May 2016 07:11:53 GMT
View Forum Message <> Reply to Message

Smart as simple. Good tip, Lance. Now, according to documentation, the AMap<class K, class T, class V, class HashFn>::FindAdd() method returns negative if the key is not found. I suppose that de docs are outdated because names a param that is no there:

int FindAdd(const K& k)
...
x Key to find.
h Precomputed hash value.
Return value Position of element or a negative value if element is not in AMap.

http://www.ultimatepp.org/src$Core$AMap$en-us.html

Anyway, many thanks!

Subject: Re: Map with unique keys and transfer semantics?
Posted by Lance on Sun, 15 May 2016 12:21:53 GMT
View Forum Message <> Reply to Message

Suppose the document is right, ie. FindAdd returns negative if it added a new entry, you can simply add a call to abs(), or you can even compose a method doing what the current FindAdd is

doing. The code and logic is open, the "key, value" are protected memebers of AMap so that inherited classes can access freely. Not much work added really.