Subject: Avoid Copy when adding a Tuple to a VectorMap
Posted by Xemuth on Sun, 06 Jun 2021 01:39:43 GMT
View Forum Message <> Reply to Message

Hello,

I'm working with a VectorMap<int, Tuple<int, A>>
class A don't provid a copy constructor, only a Move constructor.
I would like to add a new Tuple to my vector map by moving a fresh class A.
But I don't succeed... Can someone help me ? (I would like to avoid creation of a class to replace tuple in this specific case)

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

using namespace Upp;

class A{
 public:
  A(int e) : d_e(e){}
  A(A&& a){d_e = a.d_e;}
  int d_e;
};

CONSOLE_APP_MAIN
{
 VectorMap<int, Tuple<int, A>> myVector;
 myVector(1, pick(Tuple<int,A>(1, 5)));
 Cout() << myVector.Get(1).b.d_e << EOL;
 //Error : call to implicitly deleted copy constructor of 'A'
}
```

Also, why there is no Create(Args...) function in VectorMap ? (like the one in ArrayMap)

Subject: Re: Avoid Copy when adding a Tuple to a VectorMap
Posted by jjacksonRIAB on Fri, 13 Aug 2021 08:52:28 GMT
View Forum Message <> Reply to Message

I realize this is an old post and you've already figured it out, but is there any reason you can't do

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

using namespace Upp;

class A : Moveable<A> {
    public:
        A(int e) : d_e(e){}
//  A(A&& a){d_e = a.d_e;}
    int d_e;
};

CONSOLE_APP_MAIN
{
    VectorMap<int, Tuple<int, A>> myVector;
    myVector(1, pick(Tuple<int,A>(1, 5)));
    Cout() << myVector.Get(1).b.d_e << EOL;
    //Error : call to implicitly deleted copy constructor of 'A'
}
```

Subject: Re: Avoid Copy when adding a Tuple to a VectorMap
Posted by jjacksonRIAB on Mon, 11 Oct 2021 05:19:12 GMT
View Forum Message <> Reply to Message

I just realized I didn't really answer your question, but this might help with what you're doing:

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

using namespace Upp;

class A : Moveable<A> {
    public:
        A(int e) : d_e(e){}
        int d_e;
};

CONSOLE_APP_MAIN
{
    VectorMap<int, Tuple<int, A>> myVector;
    myVector.AddPick(1, MakeTuple(1, A(5)) );

    Cout() << myVector.Get(1).b.d_e << EOL;
}
```

I know AddPick uses move constructor but I'm not sure what you get out of it, although you may find MakeTuple useful from a clarity standpoint.