Subject: Re: Deepcopying One container
Posted by mirek on Mon, 02 Nov 2009 10:14:28 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Mon, 02 November 2009 03:37Hi,

I've met following problem with polymorphic classes in One containers. Consider following code:#include <Core/Core.h>
using namespace Upp;

```
class A{
public:
 virtual void DoSmthng(){Cout()<<"I'm A.\n";};
};
class B:public A{
public:
 virtual void DoSmthng(){Cout()<<"I'm B.\n";};
};

CONSOLE_APP_MAIN{
 One<A> a=new A;
 One<A> b=new B;
 One<A> c;

 c<<=a;
 c->DoSmthng();

 c<<=b;
 c->DoSmthng();
```

The output of this is I'm A.
I'm A.It surprised me at first, but after looking in the implementation of operator<<=, I understood that this is to be expected (that is not a bug).

The question is: Is there some workaround to make a copy of One without loosing the information about the type it stores? I mean to make it work same way as if you do

```
 c<<=a;
 c->DoSmthng();

 c<<=b;
 c->DoSmthng();
```

but without a and b beeing picked. Is that even posible?

Thanks for any responses.
Regards,
Honza

Surprisingly, yes, we can provide polymorphic copies - by overloading DeepCopyNew.

That can be simplified by using PolyDeepCopyNew, using virtual
Copy method.

In reality, I have never really used polymorphic deep copy, it looks a little bit tricky to me. What is your usage scenario?

Mirek
