Subject: Re: Basic questions about u++
Posted by mirek on Wed, 07 Oct 2009 17:14:04 GMT

irtech wrote on Wed, 07 October 2009 12:47mrjt wrote on Wed, 07 October 2009 18:14
By wrapping the new/delete calls in an object (one that has already been thoughly tested) you are able to utilise C++ inbuilt destruction mechanics and avoid the error-prone call to delete. There isn't any 'magic' going on outside of using templates in a clever way

As far as I'm concerned the Upp memory management philosophy is to never use new/delete. There is almost always a better way .

Ok thanks with your explanation and what I've read in overview now I understood the resource management philosophy of U++!

except one thing !
Ok your example is about a simple int but hw about a big object like a class? Then you certainly need copy constructor which seems to be the motive not to use stdlib.

Actually, in most cases, you do NOT need copy constructor.

Quote:
I mean for example I want to specifically copy value of a sibling object when copy constructor is called or I want to increment a variable inside a class inside copy constructor so I know how many times it has been copied.

Well, you have 3 kinds of entities:

- "object" (I lack better word) classes like File, Window etc... These usually do not have copy.

- "value" types like int, String, Color etc... These usually have full deep copy semantics

- and, U++ specific "containers". These act as sort of structural glue binding everything together.

And these containers, to workaround possibly missing copy contructors in cointained objects, have so called "pick transfer semantics":

http://www.ultimatepp.org/srcdoc$Core$pick_$en-us.html

which is the most "alien" thing in U++.

In reality, we could probably even live without pick, it is sort of similar in importance to "break" statement in C(++/#)/Java. But it makes live easier.

Mirek