Subject: Re: Basic questions about u++
Posted by mr_ped on Wed, 07 Oct 2009 19:28:53 GMT
View Forum Message <> Reply to Message

The C++ compiler does call destructors when the object goes out of scope. The resources used by that object are not freed magically, you have to write the destructor code. Then it is "magically" called whenever that object goes out of scope.

Probably everything you will use from U++ does already have proper destructor, so you can write it as Mirek posted, you just create new object on heap, and when you go out of scope, you know it was destroyed properly.

In case you use "new" in code (and really can't avoid it, which is usually quite easy with U++ way of doing things), you have either to detect it in destructor and call appropriate "delete" (a good idea which keeps your custom classes to behave same as U++ things) or you have to watch out during that scope and call "delete" by hand whenever it is needed (going out of scope) (quite error prone).

C++ of course calls ctors/dtors automagically (inserting those calls at proper places during compilation) just like you would expect it (well, almost) (see C++ standard), so if you manage your resources this way, you can avoid many common bugs which usually arise from new/delete in function code.
But in the end it's yours (and U++) code responsibility to free everything correctly. If you do it in U++ way, you will very rarely have to think about it, it will work almost "alone".
But still you should understand how it works and why, so you don't allocate for example 3 big memory blocks at the same time (same scope), when you need just 1 at a time and then you can release it and allocate another one (but this is also problem for GC languages, if you are way too much careless, you will degrade performance of your code tenfold, and GC will not save you).