Subject: Re: C++ FQA
Posted by Mindtraveller on Tue, 13 Nov 2007 01:41:38 GMT
View Forum Message <> Reply to Message

luzrnot only are .iml compressed using zlib, but even more importantly, several images are always compressed in a single block More I know, more I like it

cbpporterWell these execution freezes are not worse than in JVN or .NET platforms. Actually, they can even be shorter. I would like to see some real-life samples of GC performance, not just speculation or my personal experience. Have you ever used a bigger .NET or JVM application.Just don`t get me wrong: I develop nearly real-time applications (not RTA actually at all when we discuss Windows issues). I do work with actual hardware devices with a number of protocols. It all runs under highly truncated version of Windows, which doesn`t know much of system-hanging device drivers like CD-ROM ones. So, generally we have no big problems with OS latensy on protocol timeouts like 50-500 msecs.
This way neither Java or .NET, nor similar "heavy" platforms can be used (usually those industrial computers are not that quick as Pentium3/4 to support virtual machines, and memory installed may be below even 64/128 MB).
But I need to use GUI as much as time-critical code working with hardware devices. It all of course is devided into different threads etc. The thing I need most - is efficiency and predictability, afterwards - ease of use.
This all situation makes GC or other hanging solutions totally inacceptable for my tasks. Huge memory consumption is inacceptable too.
Besides, I don`t like complex solutions in a situation when simple solution can be applied without high cost. So that U++ satisfies me by a number of criteria, becoming succesful replacement for Borland C++ Builder.

You can throw rocks at me, but I don`t see necessarity of having general purpose GC when you have experience with constructing/destructing of objects. It all may be good at learning-style languages like basic, but well-organized code generally knows where and when to delete objects - more of that, you may choose a moment to do costly memory operations - when they are the mostly invisible for user (knowing program logic). I just don`t believe that GC can determine these (most effective) moments automatically.

Also.
Speaking about reference counting and GC in such a general way makes hard to think of what is really better. Maybe it would be more efficient to discuss some real-life class to be more spicific and sure in reasons. Just a thought.