

Re: std::map question

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.cpp/2004-08/2124.html

From: David Hilsee (davidhilseenews_at_yahoo.com)

Date: 08/15/04

Date: Sat, 14 Aug 2004 21:49:48 -0400

"Flzw" <flownz@wanadoo.fr> wrote in message
news:cflku0\$3bl\$1@news-reader4.wanadoo.fr...

> > *It would be better not to *work around* this but to redesign the
> > implementation of CObject. The way it is right now, destruction of one
> > instance will invalidate all copies that exist somewhere else. This does
> > not only make the use of containers impossible but is somewhat dangerous
> > anyway.*
>
> *True.*
>
> *But I thought std containers were efficient. Here with this line, it
creates
> 4 instances of the object and deletes three... Is it really worth it ?*
>
> *I mean, I will switch to std::map <string, CObject*> from std::map
<string,
> CObject> and handle new / delete myself I won't need to introduce shared
> pointers and it will be MUCH faster (to code and to run).*

If you are allocating memory dynamically and it is possible for exceptions to be thrown, it can be difficult to guarantee leak-free code unless there is an object that is responsible for managing the memory. If you take the "easy" route, then you will probably write code that can leak memory in certain cases. Also, reference counted pointers can provide other benefits, because they can alleviate the need to explicitly manage the lifetime of an object, making the code simpler and more likely to be correct. I doubt that your predictions about the improvement in the efficiency of the code will be realized, especially when you consider that the string is probably managing a dynamically allocated buffer and the object is managing a handle to an OS resource. The overhead of the smart pointer will most likely be negligible when compared to the rest of the code. Profile it and see if it's something to be concerned about.

--
David Hilsee