

Re: How to give selective access to the methods in a class?

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.programmer/2006-08/msg00807.html>

- *From:* "toton" <abirbasak@xxxxxxxxx>
 - *Date:* 11 Aug 2006 06:55:49 -0700
-

Eric Sosman wrote:

toton wrote:

Eric Sosman wrote:

toton wrote On 08/09/06 10:24,:

[...]
How much
effective is this in long run? i.e will JIT
make the object contains
side by side? or they will be scattered? (boils
down to the question,
array holds the object itself or just the
reference ?)

The instances exist "somewhere else," and the ArrayList
holds references to them. The instances might be scattered
or might be grouped together; they might even move around
to
different memory locations at different times. That's the
JVM's worry, not yours: The reference still leads to the
instance, no matter where it happens to be located.

Yes, its JVM's worry. I know that. My worry is that, do JVM really
worry to keep frequently accessed objects in nearby places?

Not that I know of -- but then, I've never bothered to look.
Also, there is more than one JVM, and different JVMs have different
strategies for managing memory.

Re: How to give selective access to the methods in a class?

The second thing what I wanted to know, for Vector or ArrayList type of random access container, how much effective the load factor or initial capacity, as the container doesn't hold the object itself. One need to choose them judiciously for STL containers, when they contains the objects itself, otherwise frequent memory copy occurs. One can see a visible performance difference there. Is that kind of fine tuning is necessary for JCF Vector or ArrayList, (or default value of 10 works fine irrespective of ArrayList size?) as it stores the reference only, the increase in size