arrayList和vector的区别

1. Vector & ArrayList

1） Vector的方法都是同步的(Synchronized),是线程安全的(thread-safe)，而ArrayList的方法不是，由于线程的同步必然要影响性能，因此,ArrayList的性能比Vector好。

2） 当Vector或ArrayList中的元素超过它的初始大小时,Vector会将它的容量翻倍,而ArrayList只增加50%的大小，这样,ArrayList就有利于节约内存空间。

2. Hashtable & HashMap

Hashtable和HashMap它们的性能方面的比较类似 Vector和ArrayList，比如Hashtable的方法是同步的,而HashMap的不是。

3. ArrayList & LinkedList

ArrayList的内部实现是基于内部数组Object[],所以从概念上讲,它更象数组，但LinkedList的内部实现是基于一组连接的记录，所以，它更象一个链表结构，所以，它们在性能上有很大的差别：

从上面的分析可知,在ArrayList的前面或中间插入数据时,你必须将其后的所有数据相应的后移,这样必然要花费较多时间，所以,当你的操作是在一列 数据的后面添加数据而不是在前面或中间,并且需要随机地访问其中的元素时,使用ArrayList会提供比较好的性能；而访问链表中的某个元素时,就必须从链表的一端开始沿着连接方向一个一个元素地去查找,直到找到所需的元素为止，所以,当你的操作是在一列数据的前面或中 间添加或删除数据,并且按照顺序访问其中的元素时，就应该使用LinkedList了。

如果在编程中，1、2两种情形交替出现,这时,你可以考虑使用List这样的通用接口,而不用关心具体的实现，在具体的情形下,它的性能由具体的实现来保证。

4. 配置集合类的初始大小

在Java集合框架中的大部分类的大小是可以随着元素个数的增加而相应的增加的，我们似乎不用关心它的初始大小,但如果我们考虑类的性能问题时,就一定要考虑尽可能地设置好集合对象的初始大小,这将大大提高代码的性能。

比如,Hashtable缺省的初始大小为101,载入因子为0.75,即如果其中的元素个数超过75个,它就必须增加大小并重新组织元素，所以,如果你 知道在创建一个新的Hashtable对象时就知道元素的确切数目如为110,那么,就应将其初始大小设为110/0.75=148,这样,就可以避免重 新组织内存并增加大小。