Difference between ArrayList and LinkedList in Java

BY CHAITANYA SINGH | FILED UNDER: JAVA COLLECTIONS

ArrayList and LinkedList both implements List interface and their methods and results are almost identical. However there are few differences between them which make one better over another depending on the requirement.

ArrayList Vs LinkedList

1) **Search**: ArrayList search operation is pretty fast compared to the LinkedList search operation. get(int index) in ArrayList gives the performance of O(1) while LinkedList performance is O(n).

Reason: ArrayList maintains index based system for its elements as it uses array data structure implicitly which makes it faster for searching an element in the list. On the other side LinkedList implements **doubly linked list** which requires the traversal through all the elements for searching an element.

2) **Deletion**: LinkedList remove operation gives O(1) performance while ArrayList gives variable performance: O(n) in worst case (while removing first element) and O(1) in best case (While removing last element).

Conclusion: LinkedList element deletion is faster compared to ArrayList.

Reason: LinkedList's each element maintains two pointers (addresses) which points to the both neighbor elements in the list. Hence removal only requires change in the pointer location in the two neighbor nodes (elements) of the node which is going to be removed. While In ArrayList all the elements need to be shifted to fill out the space created by removed element.

3) **Inserts Performance**: LinkedList add method gives O(1) performance while ArrayList gives O(n) in worst case. Reason is same as explained for remove.

4) **Memory Overhead**: ArrayList maintains indexes and element data while LinkedList maintains element data and two pointers for neighbor nodes hence the memory consumption is high in LinkedList comparatively.

There are few **similarities between** these classes which are as follows:

1. Both ArrayList and LinkedList are implementation of List interface.
2. They both maintain the elements insertion order which means while displaying ArrayList and LinkedList elements the result set would be having the same order in which the elements got inserted into the List.
3. Both these classes are non-synchronized and can be made synchronized explicitly by using Collections.synchronizedList method.
4. The iterator and listIterator returned by these classes are fail-fast (if list is structurally modified at any time after the iterator is created, in any way except through the iterator's own remove or add methods, the iterator will throw a ConcurrentModificationException).

When to use LinkedList and when to use ArrayList?

1) As explained above the insert and remove operations give good performance (O(1)) in LinkedList compared to ArrayList(O(n)). Hence if there is a requirement of frequent addition and deletion in application then LinkedList is a best choice.

2) Search (get method) operations are fast in Arraylist (O(1)) but not in LinkedList (O(n)) so If there are less add and remove operations and more search operations requirement, ArrayList would be your best bet.

References:

- **ArrayList documentation**
- **LinkedList Javadoc**

Enjoyed this post? Try these related posts

1. **How to empty an ArrayList in Java**
2. **Remove all mappings from Hashtable example – Java**
3. **HashMap in Java with Example**
4. **Replace Vector elements using index – Java example**
5. **How to sort Hashtable in java**
6. **Java – Convert Vector to ArrayList example**

Comments

krishna kumar gupta says
JUNE 19, 2014 AT 7:38 AM

really great explanations

Reply

Subhankar Adhikary says
JULY 7, 2014 AT 7:09 PM

I am new in java. I want to know basic difference between ArrayList and LinkList. This is nice article. Nice Job. Nice explained. Its really helps me.

Reply

vairam says
SEPTEMBER 12, 2014 AT 3:06 PM

very gud explanation. Superb.

Reply

enadun says

SEPTEMBER 28, 2014 AT 4:17 PM

Thanks a lot. Fully satisfied answer !

Reply

Anshuman Dwivedi says
NOVEMBER 12, 2014 AT 3:51 AM

You said –
The iterator and listIterator returned by these classes are fail-fast (if list is structurally modified at any time after the iterator is created, in any way except through the iterator's own remove or add methods.
How are you going to add element via iterator ?

Reply

Subbareddy says
DECEMBER 22, 2014 AT 1:46 PM

Hi,
This post is really good. But, In the above I have one doubt. while searching arraylist follows O(1) because of index. At the same time why arraylist will not follow same one while deletion.
regards,
Subbareddy

Reply

Neel Mehta says
JANUARY 20, 2015 AT 6:12 AM

ArrayList doesn't follow the same for delete since even though it can find the element at the given index fairly quickly, it has to shift all the elements at the later indices back one so that the empty index of the deleted element gets filled.

Reply

Mfily says
JUNE 25, 2015 AT 9:52 PM

It's really a nice post but i want to know the definition of array list and linked list I can't get the definition.

Reply

Prince Abhijeet says
FEBRUARY 20, 2017 AT 5:00 PM

You see: ArrayList has indexes and LinkedList has pointer to next node. So deleting an item of ArrayList will cause next items to shift left so shifting takes more time. Whereas in case of LinkedList when an item is deleted there is no need to shift left the next items, only thing is needed is to point the pointer to the next node.

Reply

Anonymous007 says
JUNE 20, 2017 AT 5:33 AM

An arrayList stores data at contiguous memory locations, so when deletion operation is performed it creates an empty space in arrayList. This empty space has to be occupied again by performing large number of shift operations. And hence DELETION from arrayList does not give O(1).

Reply

robert says
AUGUST 8, 2015 AT 10:17 AM

why isn't array list implemented to insert an item to the end -> so no reindexing would be needed and the insert operation would be O(1) ? If I wanted to delete an item from linked list, wouldn't it be needed to search for the item firstly (so it would be O(n)) ?

Reply

Michael says
NOVEMBER 24, 2015 AT 8:16 AM

Dear Robert, this is what I asked myself, too. I looked into the code and remove has O(n). If you use iterator for remove or add, then you have O(1).

See Stackoverflow: **http://stackoverflow.com/questions/322715/when-to-use-linkedlist-over-arraylist**

Reply

Paras Mehta says
DECEMBER 2, 2015 AT 1:19 PM

Isn't array list add is O(1) as it adds element at the end. Insertion order is maintained. So add should be O(1) only....

Reply

Wasim says
JULY 5, 2016 AT 1:03 PM

Superb explanation :)

Reply

Ilia says
JULY 28, 2016 AT 11:24 AM

Great, thanks! And what about editing the data? Which one is better if you change it a lot? Like if the type is integer and you constantly increase the values of the items. Sorry, if the question is stupid.

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

POST COMMENT

Java Tutorial

- Java Tutorial
- OOPs Concepts

Java Collections

- ArrayList
- LinkedList
- ArrayList vs LinkedList