## List in Java

A **list in Java** is a collection for storing elements in sequential order

Sequential order means the first element, followed by the second element, followed by the third element, and so on.

A good Realtime example of a list is a line of train bogies on a track: To get to the fourth bogie from the first bogie, we have to go through the second and third bogies in that order.

Java list is a sub-interface of the collection interface that is available in java.util package.

public interface List<E> extends Collection<E>

 It is an ordered collection where elements are maintained by index positions because it uses an index-based structure.

where duplicate elements are allowed.

 four concrete subclasses. They are ArrayList, LinkedList, Vector, and Stack. These four subclasses implement the list interface.

## Features of List Interface in Java

1)The list allows storing duplicate elements in Java. JVM differentiates duplicate elements by using ‘index’. Index refers to the position of a certain object in an array. It always starts at zero.

For example, assume that there is a list with size 10. Suppose the first element is ‘a’ at zero index position and the second element is also ‘a’ which is at 9th position. Thus, there are two ‘a’ elements in the list at positions 0 and 9 respectively.

So, JVM will differentiate between both elements in the list based on their numeral position of the index. Therefore, the index is very useful and plays an important role to differentiate objects.

2. In the list, we can add an element at any position.

3. It maintains insertion order. i.e. List can preserve the insertion order by using the index.

4. It allows for storing many null elements.

5. Java list uses a resizable array for its implementation. Resizable means we can increase or decrease the size of the array.

6. Except for LinkedList, ArrayList, and Vector is an indexed-based structure.

7. It provides a special Iterator called a ListIterator that allows accessing the elements in the forward direction using hasNext() and next() methods.

In the reverse direction, it accesses elements using hasPrevious() and previous() methods. We can add, remove elements of the collection, and can also replace the existing elements with the new element using ListIterator.

## Java List Initialization

After creating a list, we need to initialize the list by adding elements to it. There are three methods to initialize the list. They are as follows:

* Using Arrays.asList
* Using Normal way
* Using Anonymous Inner class

## When to use List?

1. List can be used when we want to allow or store duplicate elements.  
2. It can be used when we want to store null elements.  
3. When we want to preserve my insertion order, we should go for list.

++++++++++++++++++++++++++++++++++++++++++++++++

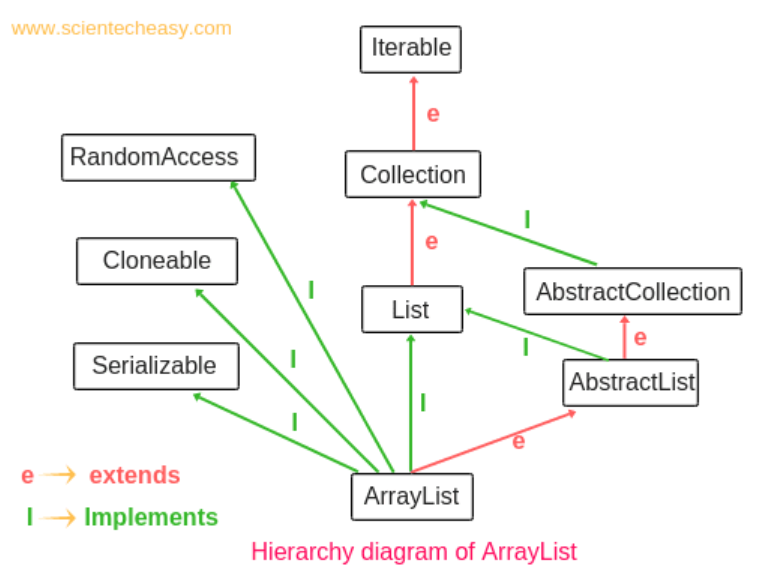
**ArrayList in Java**

 is a resizable array that can grow or shrink in the memory whenever needed. It is dynamically created with an initial capacity.

It means that if the initial capacity of the array is exceeded, a new array with larger capacity is created automatically and all the elements from the current array are copied to the new array.

Elements in ArrayList are placed according to the zero-based index. That is the first element will be placed at 0 index and the last element at index (n-1) where n is the size of ArrayList.

The capacity of ArrayList does not shrink automatically. When elements are removed from the list, the size of array list can be shrunk automatically but not capacity.



ArrayList class implements the [List interface](https://www.scientecheasy.com/2020/09/java-list-interface.html/) and extends AbstractList (Abstract class) which implements List interface.

Java ArrayList class also implements 3 marker interfaces: Random Access, Cloneable, and Serializable.

A marker interface is an interface that does not have any methods or any member variables. It is also called an empty interface because of no field or methods.

ArrayList class implements a random access interface so that we can access any random element at the same speed. For example, suppose there is a group of one crore objects in the array list. Assume that the first element is x, 10th element is y, and 1st crore element is z.

Now assume that first element x can be accessed within only 1 sec.  Due to the implementation of random access interface, the 10th element and 1st crore element can also be accessed within 1 sec.

if our frequent operation is retrieval operation then ArrayList is the best choice.

 A serializable interface is a marker interface that is used to send the group of objects over the network. It is present in the java.io package.

## Features of ArrayList in Java

**1. Resizable-array:** ArrayList is a resizable array or growable array that means the size of ArrayList can increase or decrease in size at runtime. Once ArrayList is created, we can add any number of elements.

**2. Index-based structure:** It uses an index-based structure in java.

**3. Duplicate elements:** Duplicate elements are allowed in the array list.

**4. Null elements:** Any number of null elements can be added to ArrayList.

**5. Insertion order:** It maintains the insertion order in Java. That is insertion order is preserved.

**6. Heterogeneous objects:** Heterogeneous objects are allowed everywhere except TreeSet and TreeMap. Heterogeneous means different elements.

**7. Synchronized:** ArrayList is not synchronized. That means [multiple threads](https://www.scientecheasy.com/2020/08/creating-multiple-threads-in-java.html/) can use the same ArrayList objects simultaneously.

**8. Random Access:** ArrayList implements random access because it uses an index-based structure. Therefore, we can get, set, insert, and remove elements of the array list from any arbitrary position.

**9. Performance:** In ArrayList, manipulation is slow because if any element is removed from ArrayList, a lot of shifting takes place.

For example, if an array list has 500 elements and we remove 50th elements then the 51st element will try to acquire that 50th position, and likewise all elements. Thus, it consumes a lot of time-shifting.

Once ArrayList is reached its maximum capacity, the ArrayList class automatically creates a new array with a larger capacity.

**New capacity** = (current capacity\*3/2) + 1 = 10\*3/2 + 1 = 16

After creating a new array list with a larger capacity, all the existing elements are copied into the new array list and then adds the new element into it. ArrayList class reassigns the reference to the new array objects

The old default array list with the collection of objects is automatically gone into the garbage collection

### How do we manually increase or decrease current capacity of ArrayList?

**1. ensureCapacity():** This method is used to increase the current capacity of ArrayList. Since the capacity of an array list is automatically increased when we add more elements. But to increase manually, ensureCapacity() method of ArrayList class is used.  
**2. trimTosize():** The trimTosize() method is used to trim the capacity of ArrayList to the current size of ArrayList.

ArrayList<String> list = new ArrayList<String>(); // Here, list can hold 10 elements.(Default initial capacity).

  list.ensureCapacity(20); // Now it can hold 20 elements.

list.trimTosize();

## When to use ArrayList in Java?

 We want to store duplicate elements.

 We want to store null elements.

 It is more preferred in Java when getting of the element is more as compared to adding and removing elements.

 We are not working in the multi-threading environment in Java because ArrayList is non-synchronized.