List<E> interface :
---

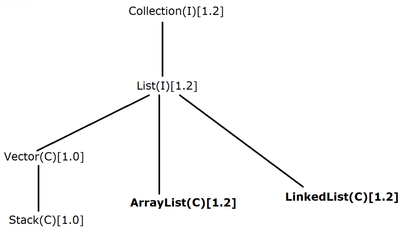
* It is the sub interface of Collection interface which is introduced from JDK 1.2V.

* It stores the element on the basis of **index.**

* It accepts duplicate and null values.

* Internally It uses indexing technique so we can sort List<E> interface implemented classes by using Collections.sort(List<E> list) method.

List interface Hierarchy :



List interface methods :
---

* As we know List interface stores the element on the basis of index so It also provides index related methods. The following are the methods of List<E> interface.

1) public boolean isEmpty() :- Verify whether List is empty or not

2) public void clear() :- Will clear all the elements, Basically List will become empty.

3) public int size() :- To get the size of the Collections(Total number of elements are available in the collection)

4) public void add(int index, E element) :- Insert the element based on the index position.

5) public boolean addAll(int index, Collection c) :- Insert the Collection based on the index position

6) public E get(int index) :- To retrieve the element based on the index position

7) public Object set(int index, Object o) :- To override or replace the existing element based on the index position.

8) public Object remove(int index) :- remove the element based on the index position

9) public boolean remove(Object element) :- remove the element based on the object element, It is the Collection interface method extended by List interface

10) public int indexOf() :- index position of the element

11) public int lastIndex() :- last index position of the element

12) public Iterator iterator() :- To fetch or iterate or retrieve the elements from Collection in forward direction only.

13) public ListIterator listIterator() :- To fetch or iterate or retrieve the elements from Collection in forward and backward direction.

Behavior of List<E> interface specific classes : [Vector<E>, Stack<E>, ArrayList<E> and LinkedList<E>]
---

* It stores the elements based on the index position.

* It can accept duplicate, null, homogeneous and hetrogenous types of object.

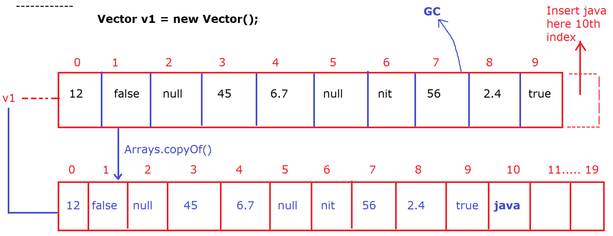
* It stores everything in the form of object.

* By using generic <> concept we can eliminate compilation warning as well as we can hold hetrogeneous types of data by using Object class.

* Iterator(Forward) and ListIterator(Forward + Backwars and index wise) both the iterator will work with List interface.

* List interface few classes are DYNAMICALLY GROWABLE (Vector + ArrayList)

Example :



How many ways we can fetch the **Collection Object** from Collections ?
---

* We have total 11 ways to fetch the Collection object from Collections :

1) By using toString() of respective class [JDK 1.0V]

2) By using Ordinary for loop [JDK 1.0V]

3) By using for-each loop [JDK 1.5V]

4) By using Enumeration interface [JDK 1.0V]

5) By using Iterator interface [JDK 1.2V]

6) By using ListIterator interface [JDK 1.2V]

7) By using Spliterator interface [JDK 1.8V]

8) By using for-each Method [JDK 1.8V]

9) By using Method Reference (::) [JDK 1.8V]

10) By using stream() method of Collection interface [JDK 1.8V]

11) By using parallelStream() method of Collection interface [JDK 1.8V]

Among all these 11 ways, Enumeration, Iterator, ListIterator, Spliterator are the **cursors so It can move from one direction to another**

Theory of 11 ways + Program :
