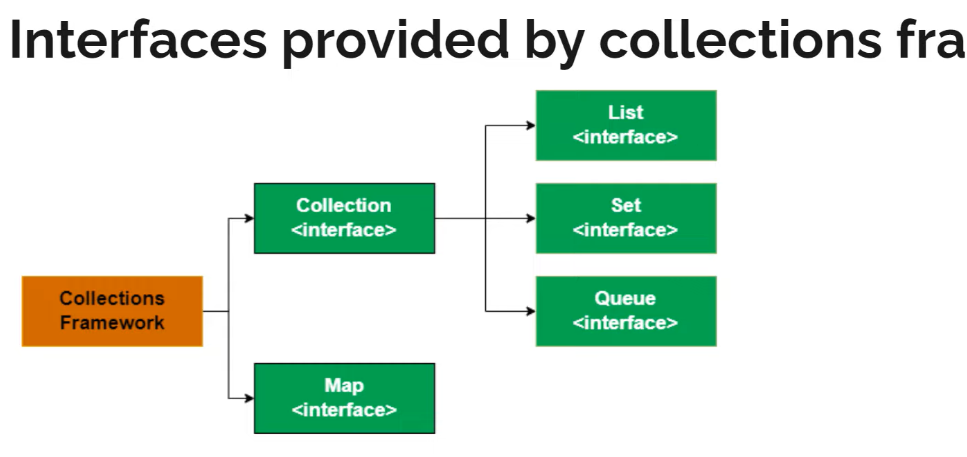
Collections

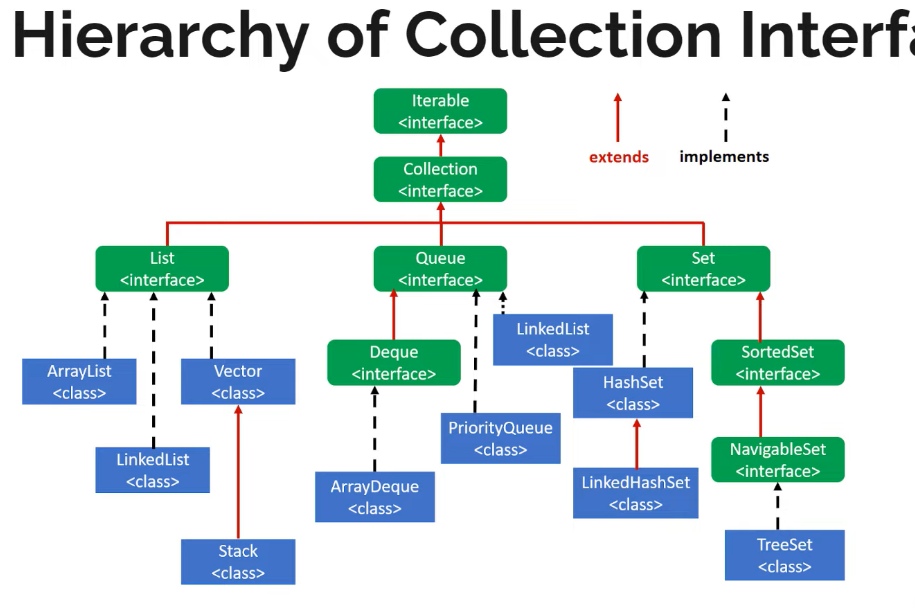
Collection is a framework provided by java.

In that framework interfaces and class are stored by set of instructions.

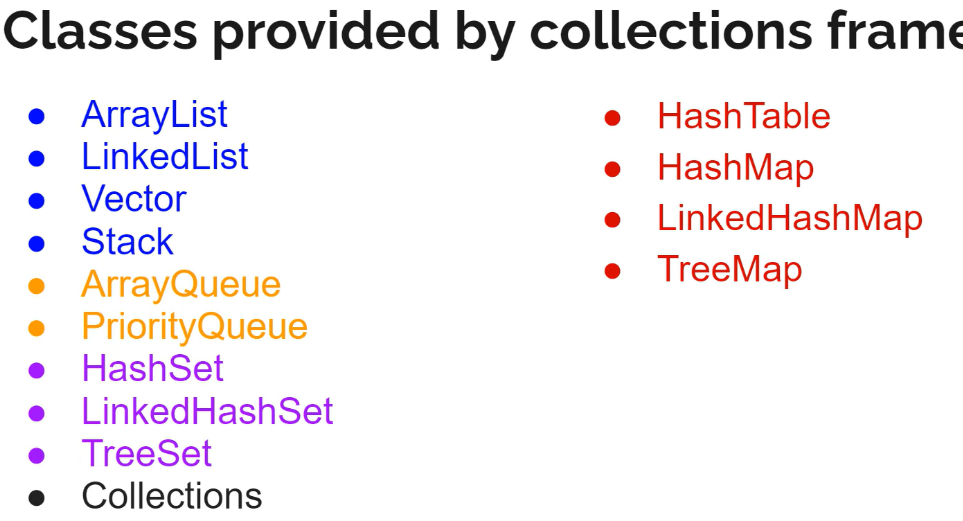
Interface provided by collection framework

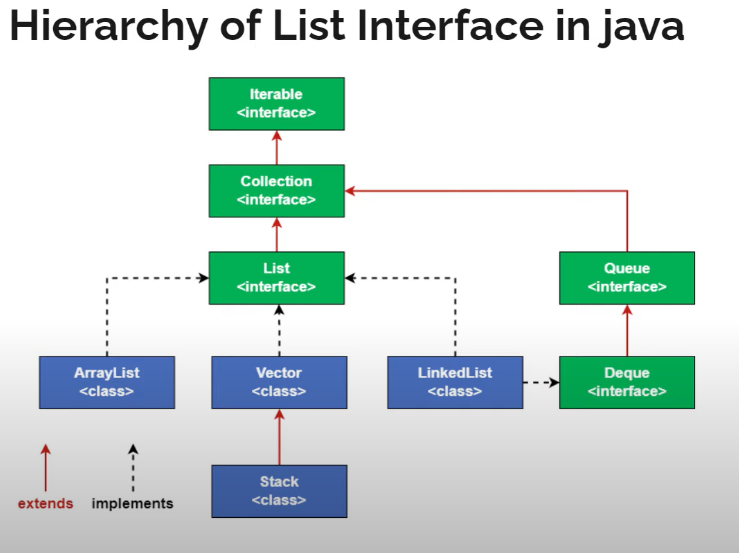


Collection Interface structure



Class provided by collection frameworks





List

Vector:

* It is used for, in requirement if values are increasing randomly we use vector or huge number of values is there means we use vector concept.
* If we create a vector by default vector size is 10. And it is increased by 100% incrimination.
* E.g.:

If we created vector it size is 10. After adding 11 element vector size becomes 20. Here 100% increment is happening.

* Vector allows duplicates.
* Allows nulls
* Follows insertion order
* Sorting order not follows
* Follows synchronized.

E.g.: In ticket counter three members are asking question to one person then that person tells first come in line and solves one by one issue is called synchronize.

* Vector supports multi-threading, vector solve line by line thread.
* Vector introduced in 1.0 version
* Later java developers found disadvantages and introduce collections 1.2 version.
* And in collections there is disadvantage so developers introduce Generic concept.
* E.g. If we won't use generic concept another developer may use code and he can add some another data type in list. After executing one exception error throws. To avoid this we use generic concept.

Array List

* It is used for, if requirement has limited data then we can go with array list.
* If we create an array list by default size is 0. After adding elements to an array then size created with 10.
* E.g.: if we created arraylist it size is 0. After adding element size becomes 10 and after adding 11 element arraylist size becomes 15. Here 50% incrimination is happening.
* Array list allows duplicates.
* Allows nulls
* Follows insertion order
* Sorting order not follows

Note:

* If any list follows insertion order sorting order won't follows.
* If list follows sorting order then insertion order won't follows.
* ArrayList won't follows synchronized but by using collection class we can convert into synchronized.

Difference between vector and arraylist?

* In vector there is some disadvantages to our come those arrayList is introduced.
* If we create a vector list and if not added any elements then array memory will be wasted, to avoid that arraylist introduced.
* Here after creating arraylist size value is 0. After adding elements only size created.
* Here based on our requirement we need to select list.

Stack

* Stack is a class in java which implements the list interface and extends the vector class and represents the LIFO principles

LIFO--> Last In First Out -->Represent stack

FIFO-->First In First Out -->Represent Queue

* In vector array elements are stored horizontal but stack array elements store vertical direction (for our imagination).
* In stack we can't create an instance by declaring size. We can create only an empty stack.
* Stack introduce in 1. Version.

Special method available in stack

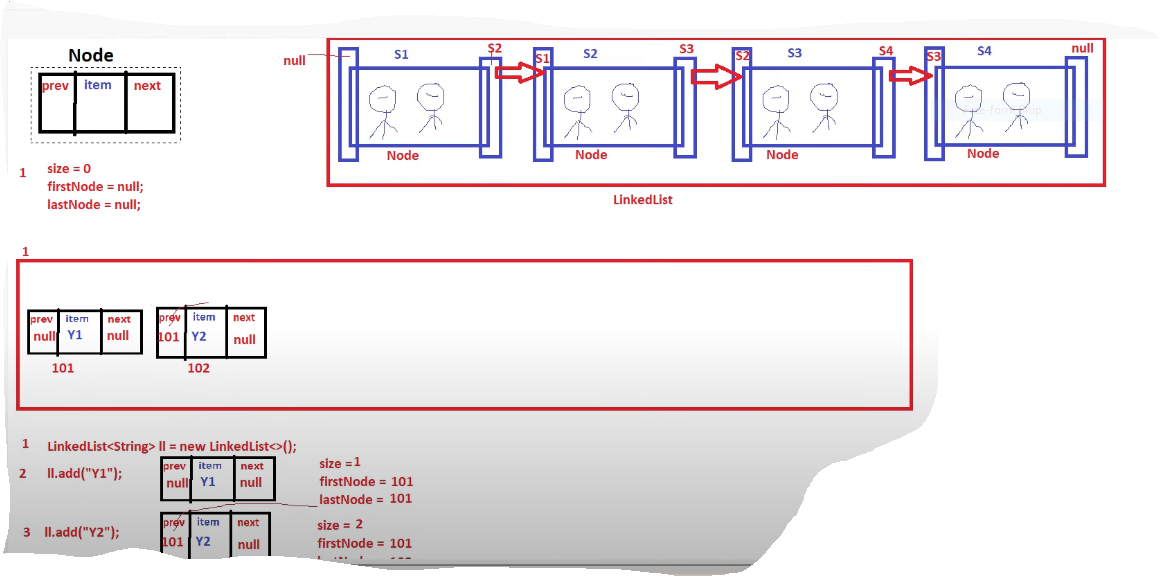
* Push
* Used to add elements to list same as add method.
* Pop
* Used to remove the last/top element.
* It display in console and remove the element from list.
* peek
* Used to find the last/top element (LIFO)
* It display in console but element won’t remove from list.
* search
* Search method used to display element position.
* Counting position start from top and count number start from 1.
* Empty
* Check list is empty or not and return Boolean type.

Important points:

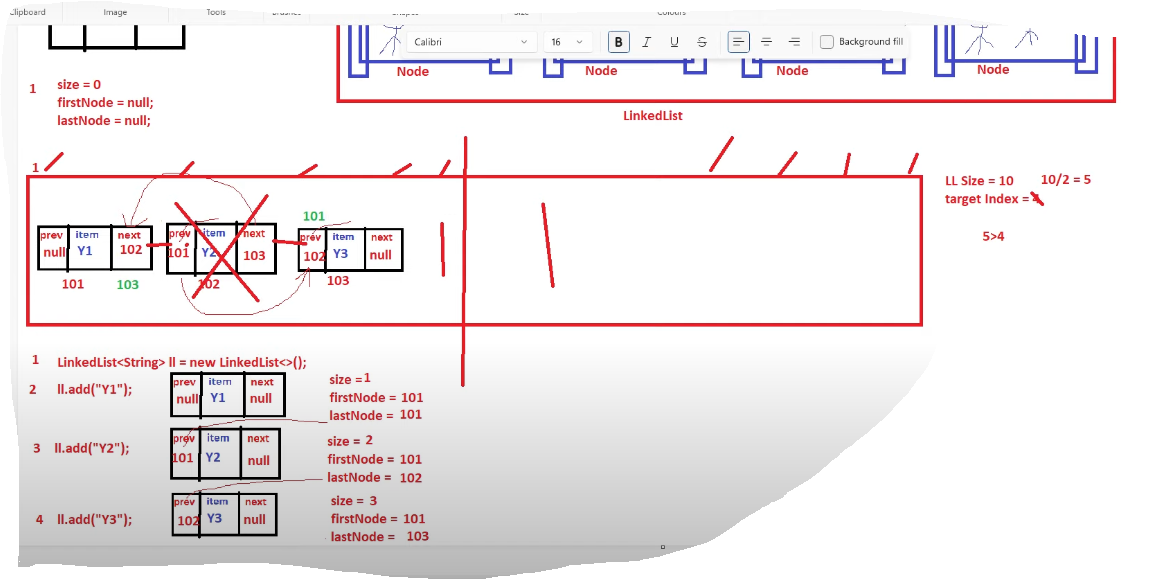
* Default/initial capacity is 10.
* Allows duplicates and null vales.
* Follows insertion order.
* Won’t follows sorted order.
* We can access random elements but we should not use it is a rule.
* Vector is synchronised because stack is super class to vector.

Linked List

* LinkedList implements list interface and queue interface (which extends deque interface).
* If we create an instance with List keyword then we can access only list related methods. Means we can't access queue methods.
* While creating instance left side with keyword List, Collections, queue, and deque we can create.
* In LinkedList we create an object called Node.
* In Node class first compartment have pervious element address, second compartment data/item, third compartment next these types of object we are linking then a linkedlist forms.
* By default linkedlist size is 0 because here there is no concept of array. Linkedlist is based on object implementation.



* Here first we creating a linked list instance. Then automatically a Node object is created. If we are adding an element to list for that data object node is creating.
* In that Node class first compartment have pervious element address, Second compartment data/item, in third compartment next object id we be present. Because to link the one Node to another node is concept of LinkedList.
* For 1st Node previous element address always null because before 1st Node there is nothing present.
* And for last Node next object id is Null.
* For every time one different id is generated for data



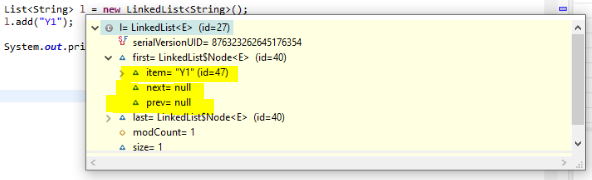
* If we are using get method to get an element linkedlist have good concept.
* E.g. Size =10, and we need 4th element then list separate by 2

10/2 =5

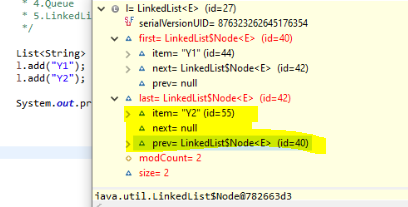
We want 4th element so it start checking directly left side.

* If we are adding element by specifying with index, it will add and create one id. That id added in next compartment which is present before Node.

Let’s see Example



Here created one linked list and to that list added one element to list. We can see prev and next are filled with Null.



Here added one more element to list then for y1 next is changed to 42.

