**Java List**

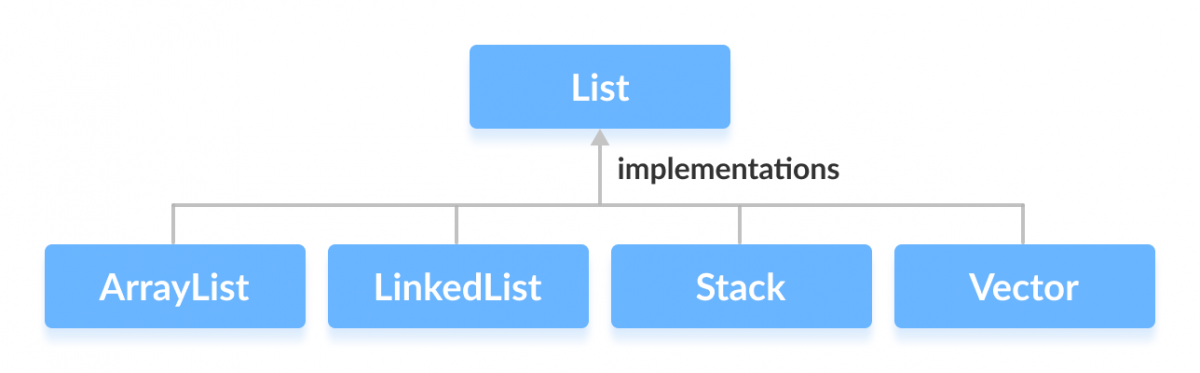
In Java, the List interface is an ordered collection that allows us to store and access elements sequentially. It extends the Collection interface.

## Classes that Implement List

Since List is an interface, we cannot create objects from it.

In order to use functionalities of the List interface, we can use these classes:

* ArrayList
* LinkedList
* Vector
* Stack



These classes are defined in the Collections framework and implement the List interface.

## How to use List?

In Java, we must import java.util.List package in order to use List.

// ArrayList implementation of List

List<String> list1 = new ArrayList<>();

// LinkedList implementation of List

List<String> list2 = new LinkedList<>();

Here, we have created objects list1 and list2 of classes ArrayList and LinkedList. These objects can use the functionalities of the List interface.

## Methods of List

The List interface includes all the methods of the Collection interface. Its because Collection is a superinterface of List.

Some of the commonly used methods of the Collection interface that's also available in the List interface are:

* add() - adds an element to a list.
* addAll() - adds all elements of one list to another
* get() - helps to randomly access elements from lists (fetch)
* iterator() - returns iterator object that can be used to sequentially access elements of lists
* set() - changes elements of lists
* remove() - removes an element from the list
* removeAll() - removes all the elements from the list
* clear() - removes all the elements from the list (more efficient than removeAll())
* size() - returns the length of lists
* toArray() - converts a list into an array
* contains() - returns true if a list contains specified element (Serching)

## Implementation of the List Interface

**1. Implementing the ArrayList Class**

import java.util.List;

import java.util.ArrayList;

class Main {

public static void main(String[] args) {

// Creating list using the ArrayList class

List<Integer> numbers = new ArrayList<>();

// Add elements to the list

numbers.add(1);

numbers.add(2);

numbers.add(3);

System.out.println("List: " + numbers);

// Access element from the list

int number = numbers.get(2);

System.out.println("Accessed Element: " + number);

// Remove element from the list

int removedNumber = numbers.remove(1); //Position

System.out.println("Removed Element: " + removedNumber);

}

}

**Output**

List: [1, 2, 3]

Accessed Element: 3

Removed Element: 2

To learn more about ArrayList, visit Java ArrayList.

**2. Implementing the LinkedList Class**

import java.util.List;

import java.util.LinkedList;

class Main {

public static void main(String[] args) {

// Creating list using the LinkedList class

List<Integer> numbers = new LinkedList<>();

// Add elements to the list

numbers.add(1);

numbers.add(2);

numbers.add(3);

System.out.println("List: " + numbers);

// Access element from the list

int number = numbers.get(2);

System.out.println("Accessed Element: " + number);

// Using the indexOf() method

int index = numbers.indexOf(2);

System.out.println("Position of 2 is " + index);

// Remove element from the list

int removedNumber = numbers.remove(1); //index number

System.out.println("Removed Element: " + removedNumber);

}

}

**Output**

List: [1, 2, 3]

Accessed Element: 3

Position of 3 is 1

Removed Element: 2

To learn more about LinkedList, visit Java LinkedList.

## Java List vs. Set

Both the List interface and the Set interface inherits the Collection interface. However, there exists some difference between them.

* Lists can include duplicate elements. **However, sets cannot have duplicate elements.**
* Elements in lists are stored in some order. However, elements in sets are stored in groups like sets in mathematics.