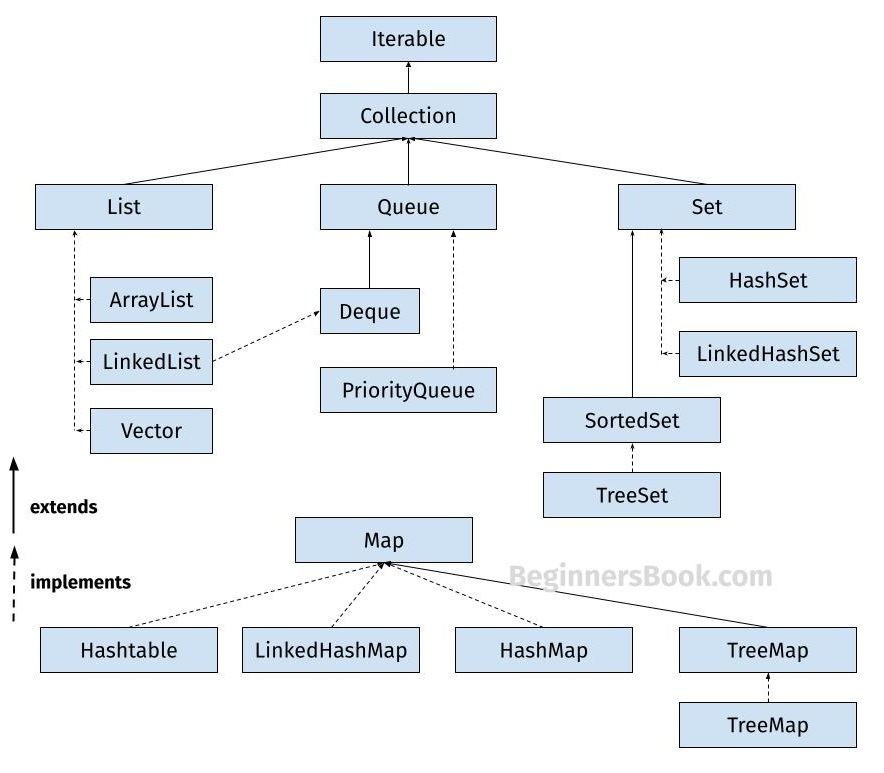
**/Collections in Java**

The **Java Collections Framework** is a collection of interfaces and classes, which helps in storing and processing the data efficiently.



## **1. List :-**

A List is an ordered Collection (sometimes called a sequence). Lists may contain duplicate elements.

* ArrayList.
* LinkedList.
* Vector. 🡪Stack

### 1.1 ArrayList

ArrayList is a popular alternative of [arrays in Java](https://beginnersbook.com/2013/05/java-arrays/). It is based on an Array **data structure**. ArrayList is a resizable-array implementation of the List interface.

### 1.2 LinkedList

LinkedList is a linear data structure. However LinkedList elements are not stored in contiguous locations like arrays, they are linked with each other using pointers.

### 1.3 Vector

Here is the list of all the tutorials published on the Vector.

### 1.4 Stack

Stack class extends Vector class, which means it is a subclass of Vector. Stack works on the concept of Last In First Out (LIFO). The elements are inserted using push() method at the end of the stack, the pop() method removes the element which was inserted last in the Stack.

## **2. Set :-**

A Set is a Collection that cannot contain duplicate elements. There are three main implementations of Set interface: HashSet, TreeSet, and LinkedHashSet.

* HashSet.
* LinkedHashSet.
* SortedSet 🡪TreeSet.

### 2.1 HashSet

[HashSet](https://beginnersbook.com/2013/12/hashset-class-in-java-with-example/) which stores its elements in a hash table, is the best-performing implementation. HashSet allows only unique elements.

### 2.2 LinkedHashSet

Unlike HashSet, the [LinkedHashSet](https://beginnersbook.com/2013/12/linkedhashset-class-in-java-with-example/) maintains insertion order.

### 2.3 TreeSet

[TreeSet](https://beginnersbook.com/2013/12/treeset-class-in-java-with-example/) stores elements in a red-black tree. It is substantially slower than HashSet. TreeSet class implements SortedSet interface, which allows TreeSet to order its elements based on their values.

## **3. Map :-**

A Map is an object that maps keys to values. A map cannot contain duplicate keys. There are three main implementations of Map interfaces: HashMap, TreeMap, and LinkedHashMap.

### 3.1 HashMap

**HashMap:** HashMap is like HashSet, it doesn’t maintain insertion order and doesn’t sort the elements in any order. Refer [this guide](https://beginnersbook.com/2013/12/hashmap-in-java-with-example/) to **learn HashMap in detail**.

### 3.2 TreeMap

**TreeMap:** It stores its elements in a red-black tree. The elements of TreeMap are sorted in ascending order. It is substantially slower than HashMap.

### 3.3 LinkedHashMap

**LinkedHashMap:** It maintains insertion order, the key & value pairs maintained the insertion order.

**4. Queue :-**

Queue (First-In-First-Out) Characteristics:

FIFO principle (First-In-First-Out) .

Elements added at the rear, removed from the front .

Basic queue operations only.

**4.1 Deque:-**

Deque (Double-Ended Queue) Characteristics:

Double-ended - operations at both ends.

Can function as both Queue and Stack.

More flexible than regular Queue.

**4.2 PriorityQueue:-**

Priority-based ordering, not insertion order.

Elements processed based on priority (natural ordering or custom comparator)

Not thread-safe \,\,

Implemented as a priority heap Implementation.