**Difference between Map, List, Set, Queue in java collection Interface and there Usage**

# **Difference between list set and map in java?**

**[List, Set and Map](https://beginnersbook.com/java-collections-tutorials/" \o "Java Collections Framework Tutorials" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)** are the interfaces which implements Collection interface. Here we will discuss difference between List Set and Map in Java.

## **List Vs Set Vs Map**

1) ****Duplicity:**** List allows duplicate elements. Any number of duplicate elements can be inserted into the list without affecting the same existing values and their indexes.  
Set doesn’t allow duplicates. Set and all of the classes which implements Set interface should have unique elements.  
Map stored the elements as key & value pair. Map doesn’t allow duplicate keys while it allows duplicate values.

2)****Null values:**** List allows any number of null values.  
Set allows single null value at most.  
Map can have single null key at most and any number of null values.

3) ****Order:**** List and all of its implementation classes maintains the insertion order.  
Set doesn’t maintain any order; still few of its classes sort the elements in an order such as LinkedHashSet maintains the elements in insertion order.  
Similar to Set Map also doesn’t stores the elements in an order, however few of its classes does the same. For e.g. TreeMap sorts the map in the ascending order of keys and LinkedHashMap sorts the elements in the insertion order, the order in which the elements got added to the LinkedHashMap.

4) ****Commonly used classes:****  
List: **[ArrayList](https://beginnersbook.com/2013/12/java-arraylist/" \o "ArrayList in java with example programs – Collections Framework" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)**, **[LinkedList](https://beginnersbook.com/2014/08/java-linkedlist-class/" \o "Java – LinkedList Class" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)** etc.  
Set: **[HashSet](https://beginnersbook.com/2013/12/hashset-class-in-java-with-example/" \o "HashSet Class in Java with example" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)**, **[LinkedHashSet](https://beginnersbook.com/2013/12/linkedhashset-class-in-java-with-example/" \o "LinkedHashSet Class in Java with Example" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)**, **[TreeSet](https://beginnersbook.com/2013/12/treeset-class-in-java-with-example/" \o "TreeSet Class in Java with example" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)**, SortedSet etc.  
Map: **[HashMap](https://beginnersbook.com/2014/08/java-hashmap-class/" \o "Java – HashMap Class" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)**, **[TreeMap](https://beginnersbook.com/2013/12/treemap-in-java-with-example/" \o "TreeMap in Java with Example" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)**, WeakHashMap, **[LinkedHashMap](https://beginnersbook.com/2013/12/linkedhashmap-in-java/" \o "LinkedHashMap in Java" \t "https://beginnersbook.com/2015/01/difference-between-list-set-and-map-in-java/_blank)**, IdentityHashMap etc.

## **When to use List, Set and Map in Java?**

1) If you do not want to have duplicate values in the database then Set should be your first choice as all of its classes do not allow duplicates.  
2) If there is a need of frequent search operations based on the index values then List (ArrayList) is a better choice.  
3) If there is a need of maintaining the insertion order then also the List is a preferred collection interface.  
4) If the requirement is to have the key & value mappings in the database then Map is your best bet.

**Introduction:**

Most real-world applications work with collections like files, variables, records from files, or database result sets. The Java language provides a set of collection frameworks that you can use to create and manage various types of object collections. This blog describes the most common collection classes and how to start using them.

**List:**

The list is an ordered collection, also known as a sequence. Because the list is organized, you have complete control over where list items are placed in a list. One thing to note here is that the Java list collection can only contain objects.

### **Declare a List:**

### A generic way

 List listOfStrings = new ArrayList();

* Using the Diamond operator

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

Here the object type is not specified during ArrayList instantiation. This is because the type of the class to the right of the expression must match the type on the left. Note that the ArrayList java object is assigned to a variable of the list type. In Java programming, you can assign one type of variable to another, as long as the assigned variable is a superclass or interface implemented by the assignment variable.

### **List Methods:**

* To place a list item on a list using add() method

List<Integer> listOfIntegers = new

ArrayList<>(); listOfIntegers.add(Integer.valueof(100));

Note that the add() method adds elements to the end of the list.

To ask how big the list is, call size(). Here in the above example to get the current list size we will call,

 listOfIntegers.size();

To retrieve an item from the list call get() and pass it the index of the item you want. For example if you want to get the first item from the listOfIntergers, then you will specify it like this:

 listOfIntegers.get(0);

To go through all the records in a list you will iterate through the collection. You can do that easily because list implements the java.lang.Iterable.  
   
When java.lang.Iterable is implemented by a collection it is called an Iterative variable collection. Here you will begin at one end and work on the collection, item by item until you’ve finished processing all the items.To get each item in a list, you can do the following:

 for (Integer i : listOfIntegers){

System.out.print(“Integer value is : ” +i);

}

**Set:**

Java [Set](https://vibrantpublishers.com/blogs/blogs-on-programming/data-structures-for-sets" \t "https://vibrantpublishers.com/en-in/blogs/blogs-on-programming/_blank) is a collection construct that, by definition, contains unique elements — that is, no duplicates. The Java Set collection can only contain objects, and it is not an ordered list, which means it does not care about the order of the elements. Because the set is an interface, you cannot instantiate it directly.

**Types of Java Set:**

In Java, there are three implementations available for a set. They are HashSet, LinkedHashSet & TreeSet.

Declare a Set:

 Set<Integer> setOfNumbers = new HashSet<>();

This Hashset is the most widely used version of a Set which gives a unique ordered list.

 Set<String> setOfNames = new LinkedHashSet<>();

The only difference in LinkedHashSet with HasSet is that it orders the elements based on insertion order.

 Set<Integer> setOfNumbers = new TreeSet<>();

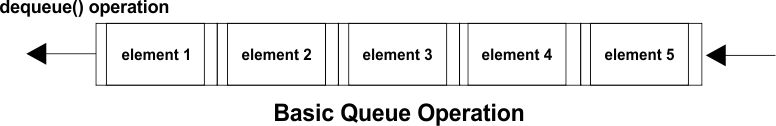
In a TreeSet the ordering takes place based on the values of the inserted element. It follows a natural ordering by default.

**Queue:**

Java [Queue](https://vibrantpublishers.com/blogs/blogs-on-programming/know-your-queue-data-structures-in-60-seconds" \t "https://vibrantpublishers.com/en-in/blogs/blogs-on-programming/_blank) is a collection that works on FIFO (First In First Out) principle. The elements that are added first will be removed first from the queue. LinkedList and Priority Queue are the most common types of Queue implementations in Java.

**Basic Queue Operations**The common operations that can be performed in a queue are addition, deletion & iteration. Like other collections here also we can find out the queue size & length.

The enqueue() method will add an element at the back of the queue and the dequeue() method will remove the item which is at the front of a given queue.



**Map:**

The map is a convenient collection construct that you can use to associate one object (key) with another object (value). As you can imagine, the key of the Map must be unique and can be used later to retrieve values. Different implementations of the Map are HashMap, TreeMap, LinkedHashMap, etc. HashMap Java is the common Map type used by programmers.  
   
**Declare a Map:**  
A Map can be declared using the Diamond Operator as given below:

 Map<Integer, String> sampleMap = new HashMap<>();

Here, the Integer will be the ‘Key’ and Sting will be the ‘Value’.

**Basic Map Operations:**  
The basic operations that can be performed in a Map are:

Put the content in Map

Get content from Map

Get the key set for Map – use it to iterate.

