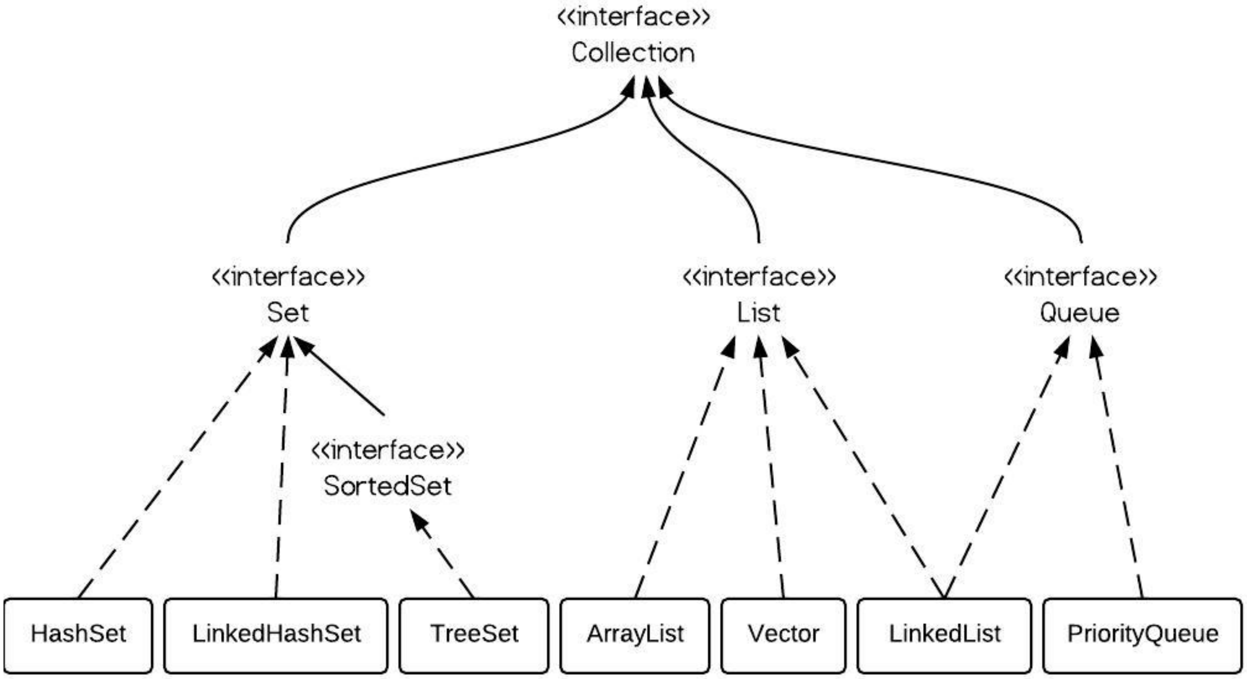
Set Interface

Set interface extends Collection interface. In a set, no duplicates are allowed. Every element in a set must be unique. We can simply add elements to a set, and finally we will get a set of elements with duplicates removed automatically.



HashSet, LinkedHashSet and TreeSet are the implementations of Set interface which does not allow duplicate elements. In this tutorial we will see the differences between them.

Duplicate Elements

- All the Set implementations does not allow duplicate elements

Order of Elements

- HashSet does not guarantee any order
- LinkedHashSet maintain insertion order
- TreeSet maintain sorting order

Null Elements

- HashSet and LinkedHashSet allows only one null
- TreeSet does not allow null

Thread-Safe

- All three are not thread-safe

Iterator

- Fail-Fast Iterator is returned by HashSet, LinkedHashSet and TreeSet.

Check for Duplicate Elements

- HashSet and LinkedHashSet uses equals() method
- TreeSet uses compareTo() method

Differences between java.util.HashSet vs java.util.LinkedHashSet vs java.util.TreeSet in java>

	Property	*java.util.HashSet*	*java.util.LinkedHashSet*	*java.util.TreeSet*

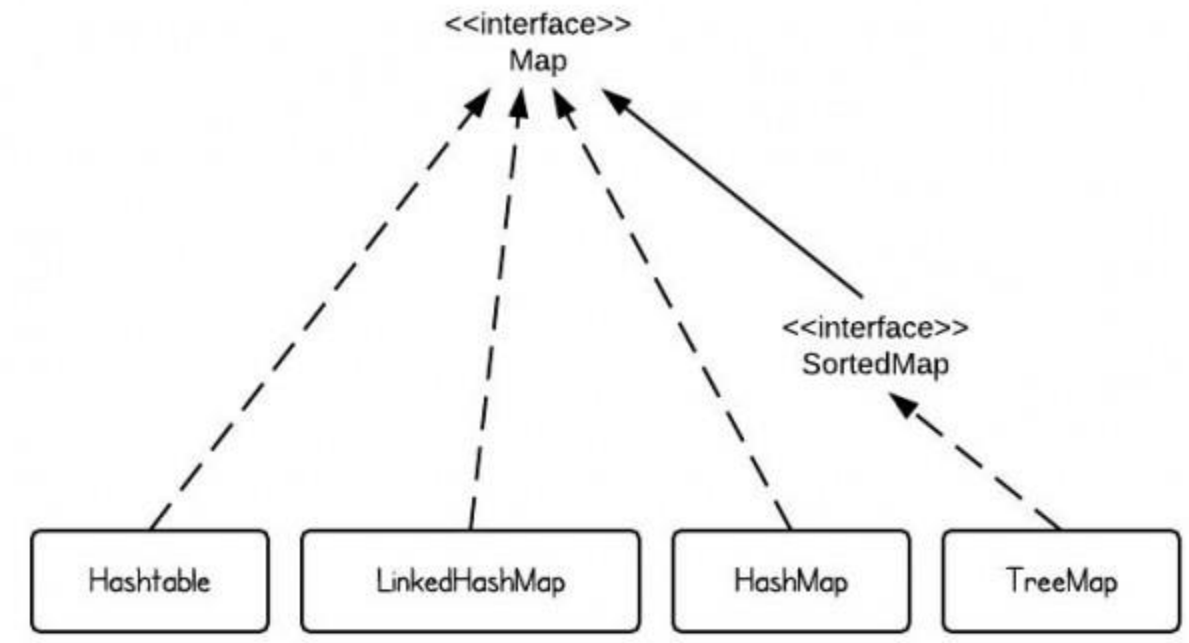
1	Insertion order	java.util.HashSet does not maintains insertion order in java.
		Example in java >
		`set.add("b");`
		`set.add("c");`
		`set.add("a");`
		Output >
		`No specific order`
			java.util.LinkedHashSet maintains insertion order in java.
			Example in java >
			`set.add("b");`
			`set.add("c");`
			`set.add("a");`
			Output >
			`b`
			`c`
			`a`
				java.util.TreeSet is sorted by natural order in java.
				Example in java >
				`set.add("b");`
				`set.add("c");`
				`set.add("a");`
				Output >
				`a`
				`b`
				`c`

2	Null elements	HashSet allows to store **one null** in java**.**	LinkedHashSet allows to store **one null** in java.	TreeSet does **not** allows to store **any null** in java. Any attempt to add null throws runtimeException (NullPointerException).
3	Data structure internally used for storing data	For storing elements HashSet internally uses HashMap.	For storing elements LinkedHashSet internally uses LinkedHashMap.	For storing elements TreeSet internally uses TreeMap.
4	Introduced in which java version	java.util.HashSet was introduced in second version of java (1.2) i.e. **JDK 2.0**	java.util.LinkedHashSet was introduced in second version of java (1.4) i.e. **JDK 4.0**	java.util.TreeSet was introduced in second version of java (1.2) i.e. **JDK 2.0**
5	Implements which interface	HashSet implements **java.util.Set** interface.	LinkedHashSet implements **java.util.Set** interface.	TreeSet implements **java.util.Set** **java.util.SortedSet** **java.util.NavigableSet** interface.

Similarity in java.util.HashSet vs java.util.LinkedHashSet vs java.util.TreeSet

	Property	**java.util.HashSet vs** **java.util.LinkedHashSet vs** **java.util.TreeSet**
1	Iterator	iterators returned by HashSet, LinkedHashSet and TreeSet all of them are *fail-fast in java.*
2	Implements	HashSet, LinkedHashSet and TreeSet all implements java.util.Set interface in java.
3	Duplicate elements	HashSet, LinkedHashSet and TreeSet *all of them does* ***not allow to store duplicate elements*** in java.

Map Overview



There are 4 commonly used implementations of Map in Java SE - HashMap, TreeMap, Hashtable and LinkedHashMap. If we use one sentence to describe each implementation, it would be the following:

- HashMap is implemented as a hash table, and there is no ordering on keys or values.
- TreeMap is implemented based on red-black tree structure, and it is ordered by the key.
- LinkedHashMap preserves the insertion order
- Hashtable is synchronized, in contrast to HashMap.

This gives us the reason that HashMap should be used if it is thread-safe, since Hashtable has overhead for synchronization.

Differences between java.util.HashMap vs java.util.Hashtable vs java.util.LinkedHashMap vs java.util.TreeMap

	Property	***HashMap***	***Hashtable***	***LinkedHashMap***	***TreeMap***
1	Insertion order	HashMap does not maintains insertion order in java.	Hashtable does not maintains insertion order in java.	LinkedHashMap maintains insertion order in java.	TreeMap is sorted by natural order of keys in java.
2	Performance	HashMap is not synchronized, hence its operations are **faster** as compared to Hashtable.	Hashtable is synchronized, hence its operations are **slower** as compared HashMap. If we are working not working in multithreading environment jdk recommends us to use HashMap.	LinkedHashMap must be used only when we want to maintain insertion order. **Time and space overhead** is there because for maintaining order it internally uses **Doubly Linked list**.	TreeMap must be used only when we want sorting based on natural order. Otherwise sorting operations cost performance. (Comparator is called for sorting purpose)
3	Null keys and values	HashMap allows to store **one null key** and **many null values** i.e. many keys can have	Hashtable does **not allow to store null key or null value**. Any attempt to store null key or value throws runtimeException	LinkedHashMap allows to store **one null key** and **many null values** i.e. any key can have null value in java.	TreeMap does **not allow to store null key but allow many null values**. Any attempt to store null key throws runtimeException

		null value in java.	(NullPointerException) in java.		(NullPointerException) in java.
4	Implements which interface	HashMap implements **java.util.Map**	Hashtable implements **java.util.Map**	LinkedHashMap implements **java.util.Map**	TreeMap implements **java.util.Map** **java.util.SortedMap** **java.util.NavigableMap**
5	Implementation uses?	HashMap use **buckets**	Hashtable use **buckets**	LinkedHashMap uses **doubly linked lists**	TreeMap uses **Red black tree**
6	Complexity of put, get and remove methods	O(1)	O(1)	O(1) **overhead** of updating **Doubly Linked list** for maintaining order it internally uses.	O(log(n))
7	Introduced in which java version?	HashMap was introduced in second version of java i.e. **JDK 2.0**	Hashtable was introduced in first version of java i.e. **JDK 1.0** But it was refactored in java 2 i.e. JDK 1.2 to implement the Map interface, hence making it a member of member of the Java Collections Framework.	LinkedHashMap was introduced in fourth version of java i.e. **JDK 4.0**	TreeMap was introduced in second version of java i.e. **JDK 2.0**