University Institute of Computer Science and Application (UICSA) RDVV, Jabalpur

By:

Mr. Naresh Ku Sondhia

Collection Framework

- **Collections:-**
- 1. Collection can hold both homogeneous data and heterogeneous data
- 2. Memory wise collections are good. Recommended to use.
- 3. Performance wise collections are not recommended to use .

Collections:-

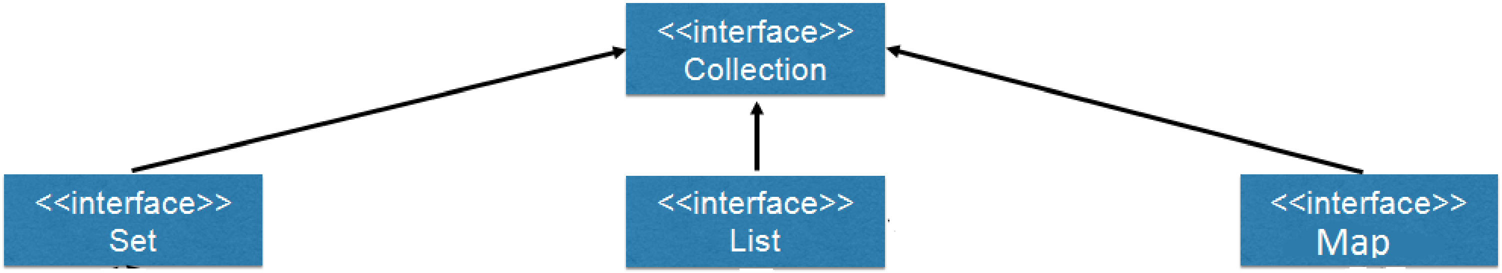
- If we want to represent group of as a single entity then we should go for collection.
- If we want to send multiple object then we should use collection.

Collection framework we having 9 key interfaces:-

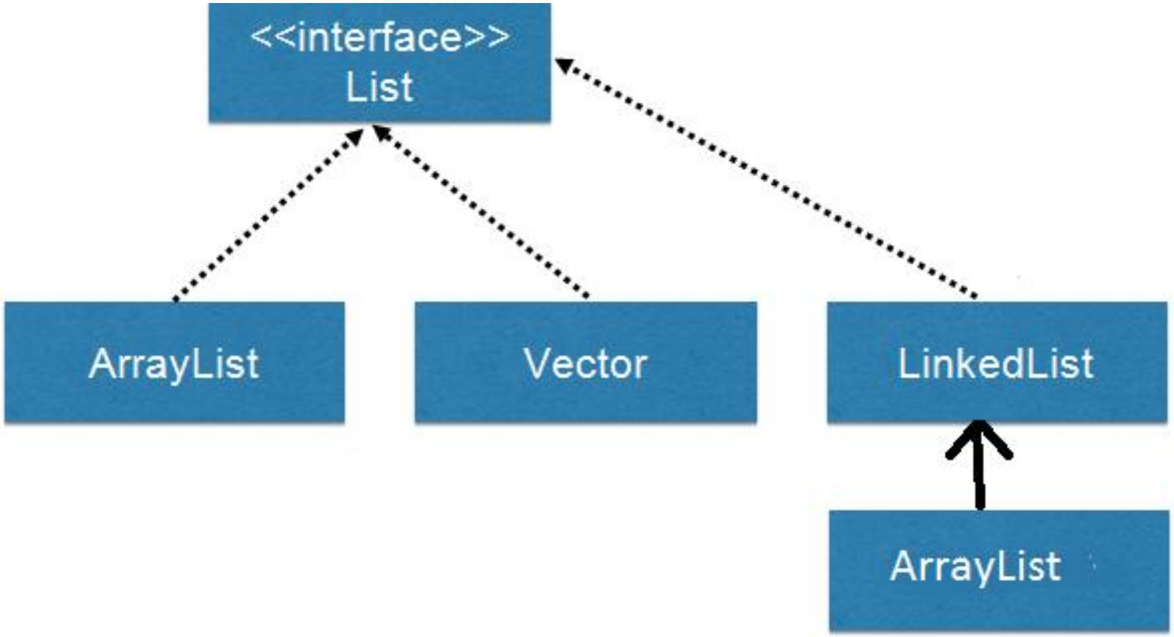
- Collection
- List
- Set
- SortedSet
- NavigablaSet
- Queue
- Map
- SotedMap
- NavigableMap

Collection:-

Collection Interface



List:-



ArrayList :- class ArrayList extends AbstractList implements List

the collection classes stores only objects but we are passing primitives these primitives are automatically converts into objects is called autoboxing.

Introduced in 1.2 version.

- ArrayList supports dynamic array that can be grow as needed.it can dynamically increase and decrease the size.
- Duplicate objects are allowed.
- Null insertion is possible.
- Heterogeneous objects are allowed.
- The under laying data structure is growable array.
- Insertion order is preserved

LinkedList:-

Class LinkedList extends AbstractSequentialList implements List, Deque, Queue

- Introduced in 1.2 version
- Duplicate objects are allowed
- Null insertion is possible
- Heterogeneous objects are allowed
- The under laying data structure is double linked list.

Vector:- (legacy class introduced in 1.0 version)

- Introduced in 1.0 v legacy classes.
- Duplicate objects are allowed
- Null insertion is possible
- Heterogeneous objects are allowed
- The under laying data structure is growable array.
- Insertion order is preserved.
- Every method present in the Vector is synchronized and hence vector object is Thread safe.

Stack:- (legacy class introduced in 1.0 version)

- It is a child class of vector
- Introduce in 1.0 v legacy class
- It is designed for LIFO(last in fist order)

Cursors:-

The main purpose of the constructors is to retrieve the data from the collection objects.

There are three types of cursors present in the java language.

- Enumaration
- Iterator
- ListIteator

Enumaration:-

- It is used for only legacy classes(Vector,Stack)
- Based on above reason the enumeration cursor is not a universal cursor
- By using this cursor it is possible to read the data only it not possible to update the data an not possible to delete the data.
- By using elements method we are getting enumaration object.

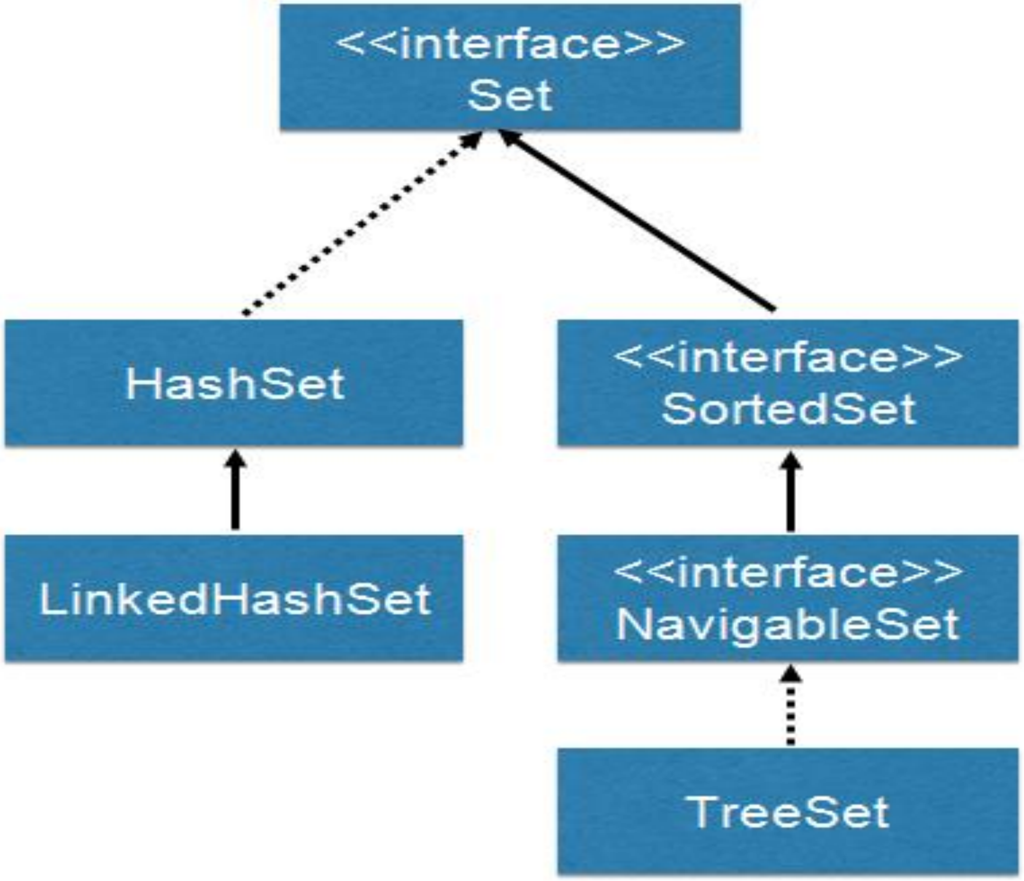
Iterator:-

- it is universal cursor we can apply any type of collection class.
- By using this it is possible to read the data and remove the data.
- We can use iterator() method to get the iterator object.

LIstIterator:-

- It is applicable for only list type of objects.
- By using this it is possible to read the data upate the data and delete data also.
- By using listIterator() method we are getting LIstIterator object

Set:-


<<interface>>
Set
HashSet
<<interface>>
SortedSet
LinkedHashSet
<<interface>>
NavigableSet
TreeSet

HashSet:-

- Introduced in 1.2 v
- Duplicate objects are not allowed if we are trying to insert duplicate values then we won't get any compilation errors an won't get any Execution errors simply add method return false.
- Null insertion is possible
- Heterogeneous objects are allowed
- The under laying data structure is hashTable.
- Insertion order is not preserved.

LinkedHashSet:-

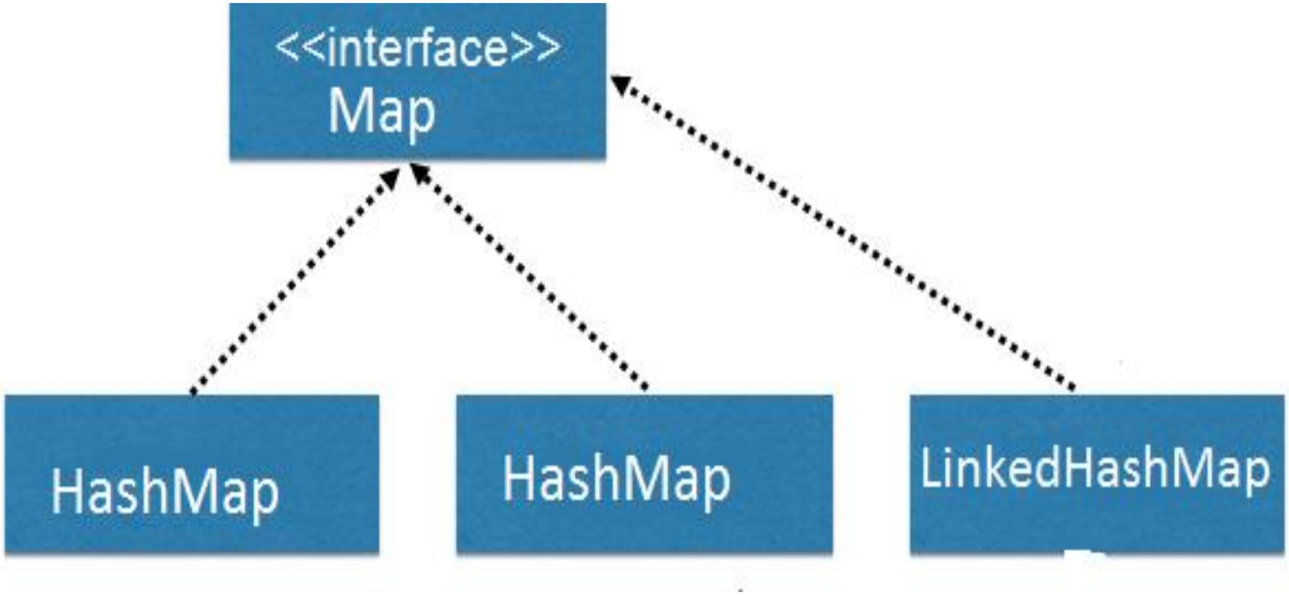
- Introduced in 1.4 v
- Duplicate objects are not allowed if we are trying to insert duplicate values then we wont get any compilation errors an won't get any Execution errors simply add method return false.

- Null insertion is possible
- Heterogeneous objects are allowed
- The under laying data structure is LinkedList & HashTable.
- Insertion order is preserved.
- It is a child class of HashSet.

TreeSet:-

- The underlying data Structure is Balenced Tree.
- Insertion order is not preserved it is based some sorting order.
- Heterogeneous data is not allowed.
- Duplicate objects are not allowed
- Null insertion is possible only once.

Map:-



Map:-

- Map is a child interface of collection.
- Up to know we are working with single object and single value where as in the map collections we are working with two objects and two elements.
- The main purpose of the collection is to compare the key value pairs and to perform necessary operation.
- The key and value pairs we can call it as map Entry.
- Both keys and values are objects only.
- In entire collection keys can’t be duplicated but values can be duplicate.

HashMap:-

- It used to hold key value pairs
- Underlying data Structure is HashTable.
- Duplicate keys are not allowed but values can be duplicated.
- Insertion order is not preserved.
- Null is allowed for key (only once)and allows for values any number of times.
- Every method is non-synchronized so multiple Threads are operate at a time hence permanence is high.

HashTable:-

- It is a legacy class introduced in the 1.0 version.
- Every method is synchronized hence only one thread is allow to access.
- The performance of the application is low.
- Null insertion is not possible if we are trying to insert null values we are getting NullPointerException.

LinkedHashMap:-

- It used to hold key value pairs
- Underlying data Structure is HashTable & LinkedList.
- Duplicate keys are not allowed but values can be duplicated.
- Insertion order is preserved