**Java LinkedHashSet**

**Introduction**

A LinkedHashSet is an ordered version of HashSet that maintains a doubly-linked List across all elements. Use this class instead of HashSet when you care about the iteration order. When you iterate through a HashSet the order is unpredictable, while a LinkedHashSet lets you iterate through the elements in the order in which they were inserted.when cycling through LinkedHashSet using an iterator, the elements will be returned in the order in which they were inserted.

**LinkedHashSet constructors**

LinkedHashSet() // Default constructor

LinkedHashSet(Collection c) // It creates LinkedHashSet from collection c

LinkedHashSet( int capacity) // it creates LinkedHashSet with initial capacity mentioned

LinkedHashSet(int capacity, float loadFactor) // This creates LinkedHashSet with capacity and load factor

**LinkedHashSet Methods**

|  |  |
| --- | --- |
| **Method** | **Purpose** |
| public boolean add(Object o) | Adds an object to a LinkedHashSet if already not present in HashSet. |
| public boolean remove(Object o) | Removes an object from LinkedHashSet if found in HashSet. |
| public boolean contains(Object o) | Returns true if object found else return false |
| public boolean isEmpty() | Returns true if LinkedHashSet is empty else return false |
| public int size() | Returns number of elements in the LinkedHashSet |

**Java Code**

[view plaincopy to clipboardprint?](http://www.w3resource.com/java-tutorial/java-linked-hashset.php)

1. **package** linkedHashSet;


5. **import** java.util.LinkedHashSet;

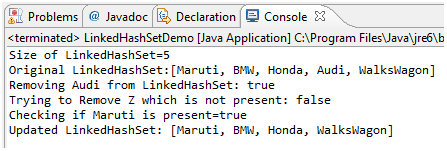

9. **public** **class** LinkedHashSetDemo {
11. **public** **static** **void** main(String[] args) {
13. LinkedHashSet<string> linkedset = **new** LinkedHashSet<string>();
15. // Adding element to LinkedHashSet
17. linkedset.add("Maruti");
19. linkedset.add("BMW");
21. linkedset.add("Honda");
23. linkedset.add("Audi");
25. linkedset.add("Maruti"); //This will not add new element as Maruti already exists
27. linkedset.add("WalksWagon");


31. System.out.println("Size of LinkedHashSet=" + linkedset.size());
33. System.out.println("Original LinkedHashSet:" + linkedset);


37. System.out.println("Removing Audi from LinkedHashSet: " + linkedset.remove("Audi"));
39. System.out.println("Trying to Remove Z which is not present: "
41. + linkedset.remove("Z"));
43. System.out.println("Checking if Maruti is present=" + linkedset.contains("Maruti"));
45. System.out.println("Updated LinkedHashSet: " + linkedset);


49. }
51. }
53. </string></string>

**Output:**



We can user Iterator object to iterate through our collection. While iterating we can add or remove objects from collection. Below program demonstrate use of iterator in LinkedHashSet collection.

**Java Code**

[view plaincopy to clipboardprint?](http://www.w3resource.com/java-tutorial/java-linked-hashset.php)

1. **package** linkedHashSet;

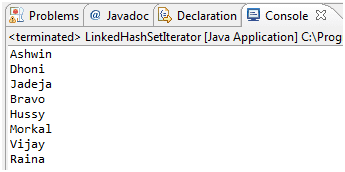

5. **import** java.util.Iterator;
7. **import** java.util.LinkedHashSet;
9. **import** java.util.Set;


13. **public** **class** LinkedHashSetIterator {


17. **public** **static** **void** main(String[] args) {
19. Set<string> myCricketerSet = **new** LinkedHashSet<string>();
21. myCricketerSet.add("Ashwin");
23. myCricketerSet.add("Dhoni");
25. myCricketerSet.add("Jadeja");
27. myCricketerSet.add("Bravo");
29. myCricketerSet.add("Hussy");
31. myCricketerSet.add("Morkal");
33. myCricketerSet.add("Vijay");
35. myCricketerSet.add("Raina");


39. Iterator<string> setIterator = myCricketerSet.iterator();
41. **while**(setIterator.hasNext()){
43. System.out.println(setIterator.next());
45. }
47. }
49. </string></string></string>

**Output**



**Summary**

* LinkedHashSet provides collection of unique objects
* LinkedHashSet is unsorted and non-indexed based collection class
* Iteration in LinkedHashSet is as per insertion order