**LinkedHashMap in Java**

[**JAVA COLLECTIONS**](http://beginnersbook.com/category/java-collections/)

[**LinkedHashMap**](http://docs.oracle.com/javase/7/docs/api/java/util/LinkedHashMap.html) is a Hash table and linked list implementation of the Map interface, with predictable iteration order. This implementation differs from HashMap in that it maintains a doubly-linked list running through all of its entries. This linked list defines the iteration ordering, which is normally the order in which keys were inserted into the map (insertion-order). In last few tutorials we have discussed about [**HashMap**](http://beginnersbook.com/2013/12/hashmap-in-java-with-example/) and [**TreeMap**](http://beginnersbook.com/2013/12/treemap-in-java-with-example/). This class is different from both of them:

* HashMap doesn’t maintain any order.
* TreeMap sort the entries in ascending order of keys.
* LinkedHashMap maintains the insertion order.

Let’s understand the LinkedHashMap with the help of an example:

import java.util.LinkedHashMap;

import java.util.Set;

import java.util.Iterator;

import java.util.Map;

public class LinkedHashMapDemo {

public static void main(String args[]) {

// HashMap Declaration

LinkedHashMap<Integer, String> lhmap =

new LinkedHashMap<Integer, String>();

//Adding elements to LinkedHashMap

lhmap.put(22, "Abey");

lhmap.put(33, "Dawn");

lhmap.put(1, "Sherry");

lhmap.put(2, "Karon");

lhmap.put(100, "Jim");

// Generating a Set of entries

Set set = lhmap.entrySet();

// Displaying elements of LinkedHashMap

Iterator iterator = set.iterator();

while(iterator.hasNext()) {

Map.Entry me = (Map.Entry)iterator.next();

System.out.print("Key is: "+ me.getKey() +

"& Value is: "+me.getValue()+"\n");

}

}

}

**Output:**

Key is: 22& Value is: Abey

Key is: 33& Value is: Dawn

Key is: 1& Value is: Sherry

Key is: 2& Value is: Karon

Key is: 100& Value is: Jim

As you can see the values are returned in the same order in which they got inserted.