HashMap in Java with Example

BY CHAITANYA SINGH | FILED UNDER: JAVA COLLECTIONS

HashMap is a Map based collection class that is used for storing Key & value pairs, it is denoted as HashMap<Key, Value> or HashMap<K, V>. This class makes no guarantees as to the order of the map. It is similar to the Hashtable class except that it is unsynchronized and permits nulls(null values and null key).

It is not an ordered collection which means it does not return the keys and values in the same order in which they have been inserted into the HashMap. It does not sort the stored keys and Values. You must need to import `java.util.HashMap` or its super class in order to use the HashMap class and methods.

HashMap Example in Java:

In this example we have demonstrated almost all the important methods of HashMap class.

```
import java.util.HashMap;
import java.util.Map;
import java.util.Iterator;
import java.util.Set;
public class Details {

   public static void main(String args[]) {

      /* This is how to declare HashMap */
      HashMap<Integer, String> hmap = new HashMap<Integer, String>();

      /*Adding elements to HashMap*/
      hmap.put(12, "Chaitanya");
      hmap.put(2, "Rahul");
      hmap.put(7, "Singh");
      hmap.put(49, "Ajeet");
      hmap.put(3, "Anuj");

      /* Display content using Iterator*/
      Set set = hmap.entrySet();
      Iterator iterator = set.iterator();
      while(iterator.hasNext()) {
         Map.Entry mentry = (Map.Entry)iterator.next();
         System.out.print("key is: "+ mentry.getKey() + " & Value is: ");
         System.out.println(mentry.getValue());
      }

      /* Get values based on key*/
      String var= hmap.get(2);
      System.out.println("Value at index 2 is: "+var);
      /* Remove values based on key*/
      hmap.remove(3);
      System.out.println("Map key and values after removal:");
      Set set2 = hmap.entrySet();
      Iterator iterator2 = set2.iterator();
      while(iterator2.hasNext()) {
          Map.Entry mentry2 = (Map.Entry)iterator2.next();
          System.out.print("Key is: "+mentry2.getKey() + " & Value is: ");
          System.out.println(mentry2.getValue());
       }

   }
}
```

Output:

```
key is: 49 & Value is: Ajeet
key is: 2 & Value is: Rahul
key is: 3 & Value is: Anuj
key is: 7 & Value is: Singh
key is: 12 & Value is: Chaitanya
Value at index 2 is: Rahul
Map key and values after removal:
Key is: 49 & Value is: Ajeet
Key is: 2 & Value is: Rahul
Key is: 7 & Value is: Singh
Key is: 12 & Value is: Chaitanya
```

HashMap Class Methods

Here is the list of methods available in HashMap class. I have also covered examples using these methods at the end of this post.

1. **void clear()**: It removes all the key and value pairs from the specified Map.
2. **Object clone()**: It returns a copy of all the mappings of a map and used for cloning them into another map.
3. **boolean containsKey(Object key)**: It is a boolean function which returns true or false based on whether the specified key is found in the map.
4. **boolean containsValue(Object Value)**: Similar to containsKey() method, however it looks for the specified value instead of key.
5. **Value get(Object key)**: It returns the value for the specified key.
6. **boolean isEmpty()**: It checks whether the map is empty. If there are no key-value mapping present in the map then this function returns true else false.
7. **Set keySet()**: It returns the Set of the keys fetched from the map.
8. **value put(Key k, Value v)**: Inserts key value mapping into the map. Used in the above example.
9. **int size()**: Returns the size of the map – Number of key-value mappings.
10. **Collection values()**: It returns a collection of values of map.

11. **Value remove(Object key)**: It removes the key-value pair for the specified key. Used in the above example.
12. **void putAll(Map m)**: Copies all the elements of a map to the another specified map.