How to synchronize HashMap in Java with example

BY CHAITANYA SINGH | FILED UNDER: JAVA COLLECTIONS

HashMap is a non-synchronized collection class. If we need to perform thread-safe operations on it then we must need to synchronize it explicitly. In this tutorial we will see how to synchronize `HashMap`.

Example:

In this example we have a HashMap<Integer, String> it is having integer keys and String type values. In order to synchronize it we are using Collections.synchronizedMap(hashmap) it returns a thread-safe map backed up by the specified HashMap.

Important point to note in the below example:
Iterator should be used in a synchronized block even if we have synchronized the HashMap explicitly (As we did in the below code).

Syntax:

```
Map map = Collections.synchronizedMap(new HashMap());
...
//This doesn't need to be in synchronized block
Set set = map.keySet();
// Synchronizing on map, not on set
synchronized (map) {
      // Iterator must be in synchronized block
      Iterator iterator = set.iterator();
      while (iterator.hasNext()){
          ...
      }
}
```

Complete Code:

```
package beginnersbook.com;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;
import java.util.Set;
import java.util.Iterator;
public class HashMapSyncExample {
    public static void main(String args[]) {
         HashMap<Integer, String> hmap= new HashMap<Integer, String>();
         hmap.put(2, "Anil");
         hmap.put(44, "Ajit");
         hmap.put(1, "Brad");
         hmap.put(4, "Sachin");
```

```
        hmap.put(88, "XYZ");

        Map map= Collections.synchronizedMap(hmap);
        Set set = map.entrySet();
        synchronized(map){
            Iterator i = set.iterator();
            // Display elements
            while(i.hasNext()) {
               Map.Entry me = (Map.Entry)i.next();
               System.out.print(me.getKey() + ": ");
               System.out.println(me.getValue());
            }
        }
    }
}
```

Output:

```
1: Brad
2: Anil
4: Sachin
88: XYZ
44: Ajit
```