**Difference between HashSet and HashMap**

By Chaitanya Singh | Filed Under: [Java.util package](https://beginnersbook.com/category/java-util-package/)

In this article we are gonna discuss the differences between [HashSet](https://beginnersbook.com/2013/12/hashset-class-in-java-with-example/) and [HashMap](https://beginnersbook.com/2013/12/hashmap-in-java-with-example/) classes.

**HashSet vs HashMap**

**Differences:**

|  |  |
| --- | --- |
| **HashSet** | **HashMap** |
| HashSet class implements the Set interface | HashMap class implements the Map interface |
| In HashSet we store objects(elements or values) e.g. If we have a HashSet of string elements then it could depict a set of HashSet elements: {“Hello”, “Hi”, “Bye”, “Run”} | HashMap is used for storing key & value pairs. In short it maintains the mapping of key & value (The HashMap class is roughly equivalent to Hashtable, except that it is unsynchronized and permits nulls.) This is how you could represent HashMap elements if it has integer key and value of String type: e.g. {1->”Hello”, 2->”Hi”, 3->”Bye”, 4->”Run”} |
| HashSet does not allow duplicate elements that means you can not store duplicate values in HashSet. | HashMap does not allow duplicate keys however it allows to have duplicate values. |
| HashSet permits to have a single null value. | HashMap permits single null key and any number of null values. |

**Similarities:**  
1) Both HashMap and HashSet are not synchronized which means they are not suitable for thread-safe operations unitl unless synchronized explicitly. This is how you can synchronize them explicitly:  
**HashSet:**

Set s = Collections.synchronizedSet(new HashSet(...));

**HashMap:**

Map m = Collections.synchronizedMap(new HashMap(...));

2) Both of these classes do not guarantee that the order of their elements will remain constant over time.

3) If you look at the source code of HashSet then you may find that it is backed up by a HashMap. So basically it internally uses a HashMap for all of its operations.

4) They both provide constant time performance for basic operations such as adding, removing element etc.

**HashSet example**

import java.util.HashSet;

class HashSetDemo{

public static void main(String[] args) {

// Create a HashSet

HashSet<String> hset = new HashSet<String>();

//add elements to HashSet

hset.add("AA");

hset.add("BB");

hset.add("CC");

hset.add("DD");

// Displaying HashSet elements

System.out.println("HashSet contains: ");

for(String temp : hset){

System.out.println(temp);

}

}

}

**Output:**

HashSet contains:

AA

BB

CC

DD

**HashMap example**

import java.util.HashMap;

class HashMapDemo{

public static void main(String[] args) {

// Create a HashMap

HashMap<Integer, String> hmap = new HashMap<Integer, String>();

//add elements to HashMap

hmap.put(1, "AA");

hmap.put(2, "BB");

hmap.put(3, "CC");

hmap.put(4, "DD");

// Displaying HashMap elements

System.out.println("HashMap contains: "+hmap);

}

}

**Output:**

HashMap contains: {1=AA, 2=BB, 3=CC, 4=DD}

**References:**

[HashSet javadoc](https://docs.oracle.com/javase/7/docs/api/java/util/HashSet.html)  
[HashMap javadoc](https://docs.oracle.com/javase/7/docs/api/java/util/HashMap.html)

**Enjoyed this post? Try these related posts**

1. [Java – How to Sort a HashSet?](https://beginnersbook.com/2015/09/sorting-hashset-java/)
2. [Java – LinkedList poll(), pollFirst() and pollLast() methods](https://beginnersbook.com/2014/08/java-linkedlist-poll-pollfirst-and-polllast-methods/)
3. [Iterate a LinkedList in reverse sequential order – java](https://beginnersbook.com/2014/08/iterate-a-linkedlist-in-reverse-sequential-order-java/)
4. [HashMap – Get value from key example](https://beginnersbook.com/2014/08/hashmap-get-value-from-key-example/)
5. [Clone a generic LinkedList in Java](https://beginnersbook.com/2014/08/clone-a-generic-linkedlist-in-java/)
6. [Delete all the elements from HashSet](https://beginnersbook.com/2014/08/delete-all-the-elements-from-hashset/)