HashSet vs HasMap

|  |  |
| --- | --- |
| **HashSet** | **HashMap** |
| HashSet class implements the Set Interface | HashMap class implements the Map Interface |
| In HashSet we store objects (elements or values) ex., 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, ex., {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 until 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(…));

1. Both of these classes do not guarantee that the order of their elements will remain constant over time
2. If you took 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.
3. They both provide constant time performance for basic operations such as adding, removing element etc.,