

JAVA - THE HASHMAP CLASS

The HashMap class uses a hashtable to implement the Map interface. This allows the execution time of basic operations, such as `get()` and `put()`, to remain constant even for large sets.

The HashMap class supports four constructors. The first form constructs a default hash map:

```
HashMap( )
```

The second form initializes the hash map by using the elements of `m`:

```
HashMap(Map m)
```

The third form initializes the capacity of the hash map to `capacity`:

```
HashMap(int capacity)
```

The fourth form initializes both the capacity and fill ratio of the hash map by using its arguments:

```
HashMap(int capacity, float fillRatio)
```

Apart from the methods inherited from its parent classes, HashMap defines the following methods:

SN	Methods with Description
1	void clear() Removes all mappings from this map.
2	Object clone() Returns a shallow copy of this HashMap instance: the keys and values themselves are not cloned.
3	boolean containsKey(Object key) Returns true if this map contains a mapping for the specified key.
4	boolean containsValue(Object value) Returns true if this map maps one or more keys to the specified value.
5	Set entrySet() Returns a collection view of the mappings contained in this map.
6	Object get(Object key) Returns the value to which the specified key is mapped in this identity hash map, or null if the map contains no mapping for this key.
7	boolean isEmpty() Returns true if this map contains no key-value mappings.

8 **Set keySet()**

Returns a set view of the keys contained in this map.

9 **Object put(Object key, Object value)**

Associates the specified value with the specified key in this map.

10 **putAll(Map m)**

Copies all of the mappings from the specified map to this map. These mappings will replace any mappings that this map had for any of the keys currently in the specified map.

11 **Object remove(Object key)**

Removes the mapping for this key from this map if present.

12 **int size()**

Returns the number of key-value mappings in this map.

13 **Collection values()**

Returns a collection view of the values contained in this map.

Example:

The following program illustrates several of the methods supported by this collection:

```
import java.util.*;

public class HashMapDemo {

    public static void main(String args[]) {

        // Create a hash map
        HashMap hm = new HashMap();
        // Put elements to the map
        hm.put("Zara", new Double(3434.34));
        hm.put("Mahnaz", new Double(123.22));
        hm.put("Ayan", new Double(1378.00));
        hm.put("Daisy", new Double(99.22));
        hm.put("Qadir", new Double(-19.08));

        // Get a set of the entries
        Set set = hm.entrySet();
        // Get an iterator
        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());
        }
        System.out.println();
        // Deposit 1000 into Zara's account
        double balance = ((Double)hm.get("Zara")).doubleValue();
        hm.put("Zara", new Double(balance + 1000));
        System.out.println("Zara's new balance: " +
            hm.get("Zara"));
    }
}
```

This would produce the following result:

```
Zara: 3434.34  
Mahnaz: 123.22  
Daisy: 99.22  
Ayan: 1378.0  
Qadir: -19.08  
  
Zara's new balance: 4434.34
```