Why Hashtable does not allow null keys or values?

As specified in JDK documentation, Hashtable does not allow null keys or values. HashMap allows one null key and any number of null values. Why is this?

java hashmap hashtable

edited Sep 6 '17 at 21:19

Pritam Banerjee

9,318 6 38 56

asked Aug 16 '12 at 6:34

Love Hasija

1,596 2 17 23

because key can't be duplicated in a single map. – Dangling Piyush Aug 16 '12 at 6:38

2 Maybe this answer will help you. – Vic Aug 16 '12 at 6:39

4 The Hashtable wants to enforce a contract by doing this. This contract ensures that the get(.) method applied to an hashtable will return null if and only if the key is not in the map. – Alexandre Dupriez Aug 16 '12 at 6:40

possible duplicate of Why does Hashtable not take null key? – Krease Nov 4 '14 at 17:59

6 Answers

Hashtable is the older class, and its use is generally discouraged. Perhaps they saw the need for a null key, and more importantly - null values, and added it in the HashMap implementation

Edit

From `Hashtable` JavaDoc:

> To successfully store and retrieve objects from a Hashtable, the objects used as keys must implement the hashCode method and the equals method.

Since `null` isn't an object, you can't call `.equals()` or `.hashCode()` on it, so the `Hashtable` can't compute a hash to use it as a key.

edited Jan 5 '17 at 8:16

answered Aug 16 '12 at 6:38

Jainendra
16.8k 23 101 150

16 `ConcurrentHashMap` is a newer class but also has the restriction of not allowing null keys or values. They add this restriction for performance reasons since it's a lot of extra work to support null keys and values but probably not useful the majority of the time. – DaoWen Aug 16 '12 at 6:45

From the Answer i was expecting a reason for Why Hashtable does not allows null keys or values ? Although you are suggesting newer and older. – Kumar Abhishek Jan 5 '17 at 4:59

Because in HashTable when you put an element it will take into account key and value hash. Basically you will have something like :

```
public Object put(Object key, Object value){

key.hashCode();
```

While in HashMap if it's null he will compute a key according to the value. Still, I personnally don't like to add null key entry.

answered Aug 16 '12 at 6:44

TecHunter

4,797 2 17 43

The reason is the reason on the accepted answer: Hashtable is old.

However, the use of Hashtable IS NOT discouraged in favor of HashMap in every scenario.

- **Hashtable is synchronized, so it is THREAD-SAFE. HashMap is not.**

Neither Hashtable nor ConcurrentHashMap support null keys or values. HashMap does.

If you want a drop-in replacement that doesn't require anything else than changing the class and works in every scenario, there is none. The most similar option would be ConcurrentHashMap (which is thread safe but doesn't support locking the whole table):

> This class is fully interoperable with Hashtable in programs that rely on its thread safety but not on its synchronization details.

HashMap is a better replacement for single threaded applications or any time synchronization is not a requirement, because of the performance impact synchronization introduces.

Sources:

- Hashtable
- HashMap
- ConcurrentHashMap

edited Mar 30 '16 at 13:43

answered Aug 6 '15 at 17:06

oops! Thank you for the heads up! I went for the obvious and thought the only difference between ConcurrentHashMap and HashMap was thread safety. Not the first time it has happened. What an akward naming convention! – NotGaeL Mar 30 '16 at 13:47

---

so to conclude

Because in HashTable when you put an element it will take into account key and value hash. Basically you will have something like :

```
public Object put(Object key, Object value){

    key.hashCode();

    //some code

    value.hashCode();

}
```

HashTable - Does not allow null keys This is because, in put(K key, V value) method, we have key.hashcode() which throws null pointer exception. HashTable - Does not allow null value This is because, in put(K key, V value) method we have if(value==null){throw new NullPointerException

HashMap allows null values as it doesn't have any checks like HashTable, while it allows only one null key. This is done with the help of putForNullKey method, which add the value to the 0th index of the internal Array every time the key is provided as null

edited Sep 9 '16 at 8:34

xenteros
11.2k 10 32 63

answered Sep 9 '16 at 8:25

Rehan
1

Hash table is very old class , from JDK 1.0

HashTable class is implemented on hashing mechanism, that's mean to store any key-value pair, its required hash code of key object. If key would be null, it will not able to given hash ,it will through null pointer exception and similar case for value it is throwing null if the value is null.

But later on it was realized that null key and value has its own importance that is why one null key and multiple null values are allowed in later implemented classes like HashMap class.

For hash map null keys will allow and there is a null check is there for keys if the key is null then that element will be stored in a zero location in Entry array. null key we can use for some default value..

=> Hashtable methods are synchronised it neveruses object based locking.

HashMap implements by considering it special

```
 static final int hash(Object key) {
       int h;
       return (key == null) ? 0 : (h = key.hashCode()) ^ (h >>> 16);
   }
```

Java 8 you cannot infer types for hash table.

```
private Map<String,String> hashtable = new Hashtable<>(); // Not Allowed

private Map<String,String> hashtable = new HashMap<>(); // Allowed
```

answered Jan 5 '17 at 5:18

Kumar Abhishek

874 9 17

HashTable - Does not allow null keys
This is because, in put(K key, V value) method, we have `key.hashcode()` which throws null pointer exception.
HashTable - Does not allow null value
This is because, in put(K key, V value) method we have `if(value==null){throw new NullPointerException`

answered Feb 29 '16 at 23:17

Arijit Dasgupta

157 1 10

The class name is java.util.Hashtable not "HashTable" ! – Sujal Mandal Aug 21 '16 at 3:50