Search ...

Total Posts: 123

Contribute Your Articles

YouTube 134

Categories

Follow

RSS 3

Shares

G+

in

Hashtable Vs SynchronizedMap Vs ConcurrentHashMap

September 21, 2017 Posted by Abhi Andhariya

ansfer data between two layers of our MVC framework.

Core Java, Core Java Interview Questions, Interview Questions, Java MultiThreading, Multithreading Interview Questions

ava Collection classes are heart of Java API. It is essential to use built-in java collections like HashMap, ArrayList or

inkedList for accessing, storing and processing data in java applications. For Example, we have extensively used HashMap to

nall core java interviews, you will definitely face questions on HashMap Internals like,

what is the use of hashCode() and equals() method? How put() and get() method works in HashMap?

structure of the HashMap (using put() or remove() method), it may cause an inconsistency in the state of HashMap.

or you may face question like How is HashSet implemented internally in Java? As a follow up question to above questions, interviewer may ask you: is HashMap a thread-safe class?

Well, **HashMap is not thread-safe.** If multiple threads are accessing the same HashMap object and try to modify the

To use HashMap in multithreaded environment, you must write your relevant code inside synchronized block or use any external Lock implementation. But in that case there are high chances of errors and deadlock situations, if proper care has

not been taken. In short, it is not advisable to use HashMap in multithreaded environment. Instead use any of the similar thread-safe collections like Hashtable, Collections. Synchronized Map or Concurrent Hash Map.

Though all of them are thread-safe, ConcurrentHashMap provides better performance than remaining two. lets understand them one by one.

Hashtable is a legacy class available since jdk 1.1 which uses synchronized methods to achieve thread safety. At a time only one thread can read or write into Hashtable. In other word, thread acquires lock on entire Hashtable instance. Hence

its performance is quite slow and we can not utilize the advantages of multithreaded architecture.

Another point to note about Hashtable is that it does not allow null keys or values whereas HashMap allows one null key and any number of null values.

Hashtable

Collections.SynchronizedMap SynchronizedMap is a static inner class of utility class java.util.Collections. It is quite similar to Hashtable and it also acquires lock on entire Map instance. It is not a legacy class like Hashtable and it was introduced in jdk 1.5.

Collections.synchronizedMap(Map map) static method. For Example,

import java.util.*;

public class SynchronizedMapDemo {

It provides functionality to convert any thread-unsafe Map implementation to thread-safe implementation using

public static void main(String[] args) { // create HashMap Map<String,String> map = new HashMap<String,String>();

```
// populate the map
     map.put("1","Malay");
     map.put("2","Ankit");
     map.put("3","Chintan");
     // create a synchronized map
     Map<String,String> syncmap = Collections.synchronizedMap(map);
     System.out.println("Synchronized map is : "+syncmap);
Output
  Synchronized map is : {3=Chintan, 2=Ankit, 1=Malay}
```

get() and remove() method implementation of SynchronizedMap :

//other instance variables private final Map < k ,v > m; // Backing Map

implements Map<K,V>, Serializable {

mutex;

private static class SynchronizedMap<K,V>

final Object

// Object on which to synchronize

It internally wraps all the methods of Map interface with synchronized keyword. For example, here is the internal put(),

```
SynchronizedMap(Map<K,V> m) {
             if (m==null)
                 throw new NullPointerException();
             this.m = m;
             mutex = this;
                  // Other constructors and methods
                 public V get(Object key) {
             synchronized(mutex) {return m.get(key);}
          public V put(K key, V value) {
             synchronized(mutex) {return m.put(key, value);}
          public V remove(Object key) {
             synchronized(mutex) {return m.remove(key);}
Click here to check full code of SynchronizedMap.
Similar to HashMap, Synchronized HashMap allows one null key and multiple null values as it just wraps the methods of
HashMap with synchronized keyword. Rest of the behavior remains same as original collection.
You can also check SynchronizedSortedMap which is similar data structure to convert thread-unsafe TreeMap and other
SortedMap implementation to corresponding thread-safe collection.
```

performance as at a time only one thread can access that Map instance. To overcome this issue, ConcurrentHashMap was introduced in Java 5 along with other concurrent classes like

ConcurrentHashMap

HashMap

Thread 2

CountDownLatch, CyclicBarrier, CopyOnWriteArrayList, BlockingQueue within java.util.Concurrent package. More than one threads can read and write concurrently in ConcurrentHashMap and still it provides thread-safety. Amazing, isn't it? How is it implemented internally?

Hashtable and SynchronizedMap both acquires lock on entire Map object which provides thread-safety, but not good

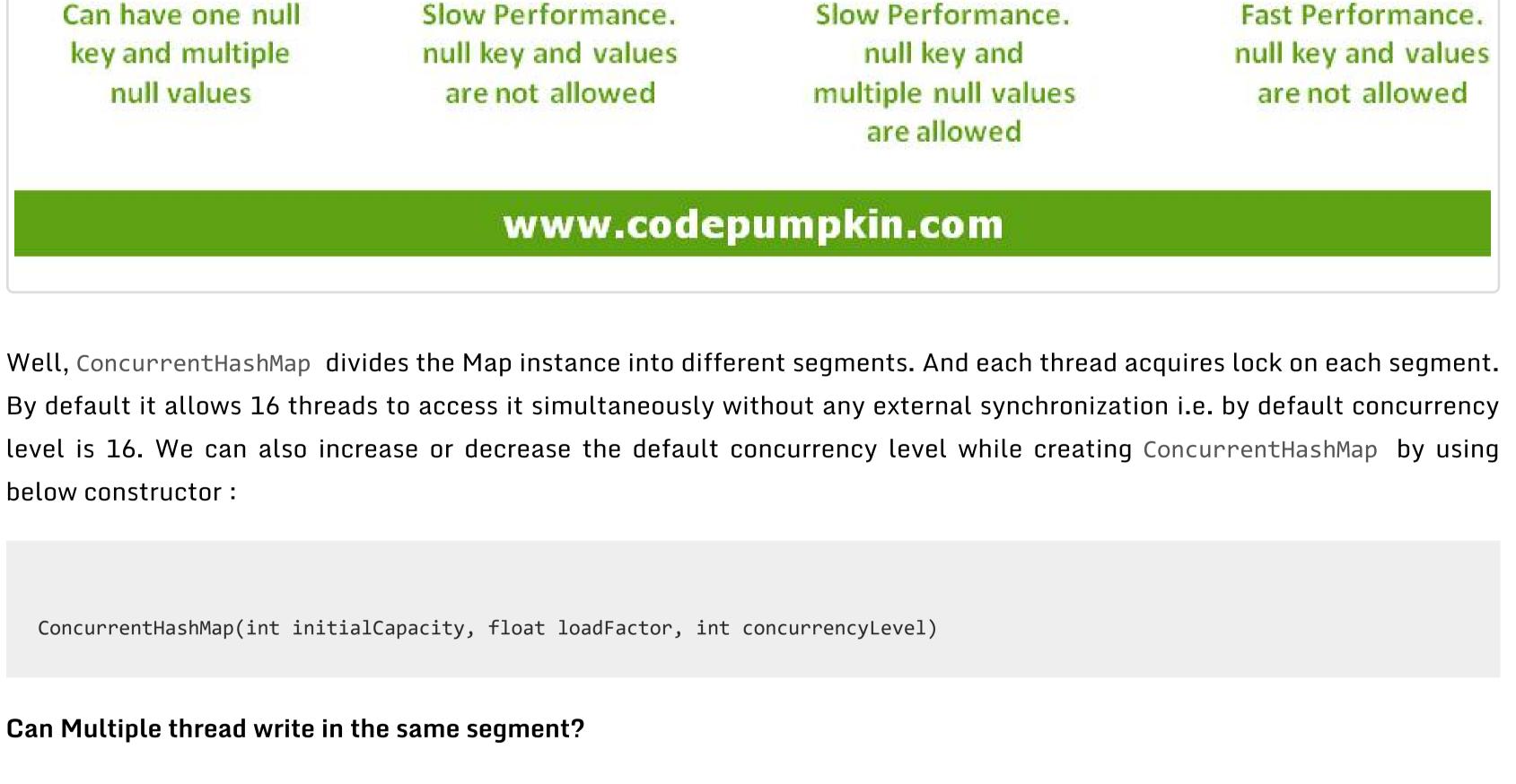
Reading waiting on the threads Thread 1 Thread 1 same segment Thread Thread 1

Thread 2

SynchronizedMap

Thread-Safe.

Hashtable



If one thread is writing in a segment, can another thread read from that segment()? Yes. but in this case last updated value will be seen by the reading thread.

ConcurrentHashMap doesn't allow null keys and null values.

4. Hashtable and ConcurrentHashMap doesn't allow null keys and null values, whereas SynchronizedMap may allow null keys and null values based on the original collection class being passed inside it.

- You can also check our articles on sorting and searching such as Bubble Sort, Selection sort, Binary Search, Fibonacci Search, Merge Sort etc.
- If you like the content on CodePumpkin and if you wish to do something for the community and the planet Earth, you can donate to our campaign for planting more trees at CodePumpkin Cauvery Calling Campaign.

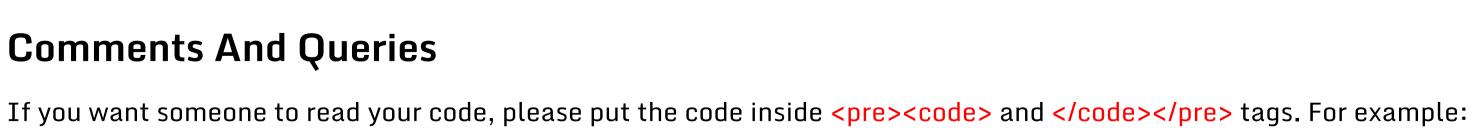
We may not get time to plant a tree, but we can definitely donate ₹42 per Tree.

Abhi Andhariya



















Surviving Java Developer, Passionate Blogger, Table Tennis Lover, Bookworm, Occasional illustrator and a



For more information on supported HTML tags in disgus comment, click here.

Share 1.9K people like this. Sign Up to see what your friends like.

Interview Questions Categories Algorithms (10) Core Java Interview Questions (19)C/C++(2) Multithreading Interview Questions **(7)** Core Java (48)

Tic Tac Toe | Java Program Implementation Program to find Unique Array Element Snakes N Ladders | Java Program

Popular Posts

No part of this blog maybe copied or reproduced or reused or republished in any way, except to share either using

the share feature of LinkedIn,

Facebook, Twitter or any other Social Media Sites or posting a direct link to

> Algorithms (10) > C / C++ (2) > Core Java (48) Java MultiThreading (11) Data Structure (10) > Design Patterns (15) > Javascript (7) > Pumpkin Practice (10) > Python (15) > SQL (2) Tag Cloud Algorithms Collection Framework

Collections Concurrency Core Java Creational Design Patterns DataStructure

Design Patterns HashMap

Java Java B Java keywords

javascript JVM internals Multithreading python Python Tutorials Synchronization Thread x vs y **Interview Questions**

Core Java Interview Questions

Questions (7) > Programming (7) > SQL Interview Questions (2)

Multithreading Interview

(19)

Popular Posts Tic Tac Toe | Java Program **Implementation** > Program to find Unique Array

How is HashSet

implemented internally in Java?

> Sudoku Solver using Recursive

Snakes N Ladders | Java Program

High Radius Technologies (1)

Sapient Global Markets (3)

Element

Backtracking

Implementation Interview Experiences > CGI (1)

Morgan Stanley (6)

> ServiceNow (1)

> Synechron (1)

Recent Posts > CGI Interview Questions - Set 1

> File I/O | Python

isha

Tree.

> Wissen Infotech (2)

for loop | Python Flow Control **Cauvery Calling**

We may not get time to plant a tree,

but we can definitely donate ₹42 per

Join us to save the planet Earth by

donating at CodePumpkin Cauvery

ACTION NOW to Save Cauvery

₹42 per tree

CauveryCalling.org

#CauveryCalling

> Exception Handling | Python

> while loop | Python Flow Control

Like Us On Facebook Code Pumpkin

Calling Campaign.

ConcurrentHashMap

Thread 2

Thread 3 is

Multiple

Thread-Safe.

Null keys and values

No. Thread acquires a lock on segment in put() operation and at a time only one thread can write in that segment.

While in ConcurrentHashMap, even if its size become very large, only portion or segment of the Map is locked which improves the performance in multithreading environment. 3. SynchronizedMap just wraps original Map object with synchronized block.

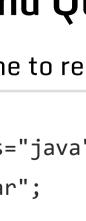
We will soon publish our article on the internal working of ConcurrentHashMap. Stay tuned 🙂

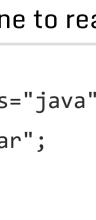
that in our post. You can also contribute your articles by creating contributor account here. Happy Learning 🙂

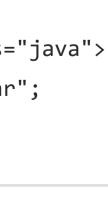
Visit Full Profile

big fan of Joey Tribbiani, The Walking Dead and Game of Thrones...!!

Tags: Collection Framework, Collections, Core Java, HashMap, Java, Synchronization, Thread, x vs y













Programming (7)

SQL Interview Questions (2)

How is HashSet implemented internally in Java? Implementation Sudoku Solver using Recursive Backtracking

this blog - An excerpt from the blog may be quoted while sharing it in the above mentioned manner. Any other form of reuse, must be only after explicit written consent of the CodePumpkin.

Privacy Policy

Disclaimer

Terms And Condtions

is waiting is waiting Not Thread-Safe. Thread-Safe. Can have one null Slow Performance.

Thread 2

Yes. Two threads are allowed to write concurrently in different segments. Can Multiple thread read from the same segment? Yes. Thread doesn't acquire a lock on segment in get() operation and any number of threads can read from the same

Can two threads write in the different segment?

Summary 1. HashMap is not thread-safe. Once the size of Hashtable and SynchronizedMap becomes considerable large because for the iteration it has to be locked for the longer duration.

segment.

That's all for this topic. If you guys have any suggestions or queries, feel free to drop a comment. We would be happy to add

About The Author

<code class="java"> String foo = "bar"; </code>

Data Structure (10)

Design Patterns (15)

Interview Experience (15) cgi (1) High Radius Technologies (1) Morgan Stanley (6) Sapient Global Markets (3) ServiceNow (1) Synechron (1) Wissen Infotech (2) Javascript (7)

Pumpkin Practice (10)

Python (15)

SQL (2)

Java MultiThreading (11)

Copyright @ 2017 Code Pumpkin All Rights Reserved.

Theme created by Themes Tune. Powered by WordPress.org