20+ Core Java Interview Questions Answers

These Java interview questions are a mix of easy, tough and tricky Java questions e.g. Why multiple inheritance is not supported in Java is one of the tricky questions. Most questions are asked on Senior and experienced level i.e. 3, 4, 5 or 6 years of Java experience e.g. How HashMap works in Java, which is most popular on experienced Java interviews.

By the way, recently I was looking at answers and comments made on Java interview questions given in this post and I found some of them quite useful to include into main post to benefit all. By the way apart from blogs and articles, you can also take advantage of some books, which are especially written for clearing any programming interviews and some focused on Java programming, two books which comes in minds are [Cracking the Coding Interview](http://www.amazon.com/dp/098478280X/?tag=javamysqlanta-20) and Programming Interviews Exposed: Secrets to Landing Your Next Job and. Both books are focused on programming in general and lot of other related topics e.g. data structures, algorithms, database, SQL, networking and behavioral questions, but also contains Java concepts.

**Question 1: What’s wrong using HashMap in the multi-threaded environment? When does the**get()**method go to an infinite loop?** ([answer](http://java67.blogspot.com/2013/06/how-get-method-of-hashmap-or-hashtable-works-internally.html))

Well, nothing is wrong, depending on how you use it. For example, if you initialize the HashMap just by one thread and then all threads are only reading from it, then it’s perfectly fine. One example of this is a Map which contains configuration properties. The real problem starts when at-least one of that thread is updating HashMap i.e. adding, changing or removing any key value pair. Since put() operation can cause re-sizing and which can further lead to infinite loop, that’s why either you should use [Hashtable](http://javarevisited.blogspot.com/2012/01/java-hashtable-example-tutorial-code.html) or [ConcurrentHashMap](http://javarevisited.blogspot.com/2013/02/concurrenthashmap-in-java-example-tutorial-working.html), later is better.

**Question 2. Does overriding the hashCode() method have any performance implication?**([answer](http://java67.blogspot.com/2013/04/example-of-overriding-equals-hashcode-compareTo-java-method.html))

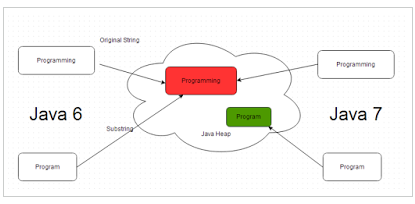
This is a good question and open to all, as per my knowledge a poor hash code function will result in the frequent collision in HashMap which eventually increases the time for adding an object into Hash Map. From Java 8 onwards though, collision will not impact performance as much as it does in earlier versions, because after a threshold the linked list will be replaced by the binary tree, which will give you O(logN) performance in the worst case, as compared to O(n) of linked list.

**Question 3: Do all properties of an Immutable Object need to be final?**([answer](http://javarevisited.blogspot.com/2013/03/how-to-create-immutable-class-object-java-example-tutorial.html))

Not necessarily, as stated above you can achieve same functionality by making the member non-final but private and not modifying them except in a constructor. Don’t provide setter methods for them and if it is a mutable object, then don’t ever leak any reference for that member. Remember making a reference variable final, only ensures that it will not be reassigned to a different value, but you can still change individual properties of object, pointed by that reference variable. This is one of the key point, Interviewer like to hear from candidates.

**Question 4: How does the substring() method inside String works?**([answer](http://javarevisited.blogspot.sg/2011/10/how-substring-in-java-works.html))

Another good Java interview question, I think the answer is not sufficient, but here it is “Substring creates a new object out of source string by taking a portion of original string”. This question was mainly asked to see if the developer is familiar with the risk of memory leaks, which a sub-string can create. Until Java 1.7, substring holds the reference of the original character array, which means even a sub-string of 5 character long, can prevent 1GB character array from garbage collection, by holding a strong reference.



This issue is fixed in Java 1.7, where the original character array is not referenced anymore, but that change also made the creation of substring a bit more costly in terms of time. Earlier it was on the range of O(1), which could be O(n) in worst case on Java 7.

**Question 5: Can you write a critical section code for the singleton?**([answer](http://javarevisited.blogspot.sg/2014/05/double-checked-locking-on-singleton-in-java.html))

This core Java question is a followup of the previous question and expecting the candidate to write Java singleton using double checked locking. Remember to use the volatile variable to make Singleton thread-safe. Here is the code for critical section of a thread-safe Singleton pattern using double checked locking idiom:

|  |  |
| --- | --- |
| 01 | public class Singleton { |
| 02 |  |

|  |  |
| --- | --- |
| 03 | private static volatile Singleton \_instance; |
| 04 |  |

|  |  |
| --- | --- |
| 05 | /\*\* |
| 06 | \* Double checked locking code on Singleton |

|  |  |
| --- | --- |
| 07 | \* @return Singelton instance |
| 08 | \*/ |

|  |  |
| --- | --- |
| 09 | public static Singleton getInstance() { |
| 10 | if (\_instance == null) { |

|  |  |
| --- | --- |
| 11 | synchronized (Singleton.class) { |
| 12 | if (\_instance == null) { |

|  |  |
| --- | --- |
| 13 | \_instance = new Singleton(); |
| 14 | } |

|  |  |
| --- | --- |
| 15 | } |
| 16 | } |

|  |  |
| --- | --- |
| 17 | return \_instance; |
| 18 | } |

|  |  |
| --- | --- |
| 19 |  |
| 20 | } |

**Question 6: How do you handle error condition while writing stored procedure or accessing stored procedure from java?**([answer](http://javarevisited.blogspot.com/2013/04/spring-framework-tutorial-call-stored-procedures-from-java.html))

This is one of the *tough Java interview questions* and its open for all, my friend didn’t know the answer so he didn’t mind telling me. My take is that stored procedure should return an error code if some operation fails but if stored procedure itself fails than catching SQLException is the only choice.

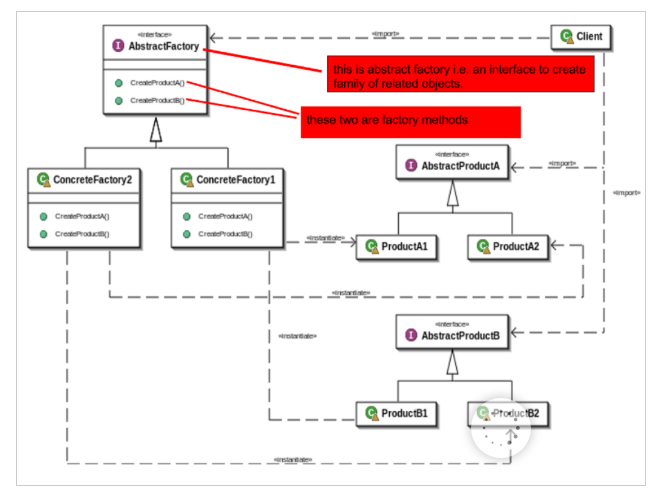
**Question 7 : What is difference between Executor.submit() and Executer.execute() methods ?**([answer](http://java67.blogspot.com/2012/08/5-thread-interview-questions-answers-in.html))

This question is from my list of Top 15 Java multi-threading question answers. It’s getting popular day by day because of huge demand of Java developers with good concurrency skills. The answer is that former returns an object of Future which can be used to find result from worker thread.

There is a difference when looking at exception handling. If your tasks throw an exception and if it was submitted with executing this exception will go to the uncaught exception handler (when you don’t have provided one explicitly, the default one will just print the stack trace to System.err). If you submitted the task with submit any thrown exception, [checked exception](http://javarevisited.blogspot.sg/2011/12/checked-vs-unchecked-exception-in-java.html) or not, is then part of the task’s return status. For a task that was submitted with submitting and that terminates with an exception, the Future.get() will re-throw this exception, wrapped in an ExecutionException.

**Question 8:  What is the difference between factory and abstract factory pattern?**([answer](http://javarevisited.blogspot.sg/2013/01/difference-between-factory-and-abstract-factory-design-pattern-java.html))

Abstract Factory provides one more level of abstraction. Consider different factories each extended from an Abstract Factory and responsible for the creation of different hierarchies of objects based on the type of factory. E.g. AbstractFactory extended by AutomobileFactory, UserFactory, RoleFactory etc. Each individual factory would be responsible for the creation of objects in that genre. Here is UML diagram of factory and abstract factory pattern:



**Question 9: What is a Singleton? Is it better to make the whole method synchronized or only critical section synchronized?**([answer](http://javarevisited.blogspot.com/2012/12/how-to-create-thread-safe-singleton-in-java-example.html))

Singleton in Java is a class with just one instance in the whole Java application, for example, java.lang.Runtime is a Singleton class. Creating Singleton was tricky prior Java 4 but once Java 5 introduced Enum its very easy.

**Question 10:  Can you write code for iterating over HashMap in Java 4 and Java 5?**([answer](http://java67.blogspot.com/2014/05/3-examples-to-loop-map-in-java-foreach.html))

Tricky one but he managed to write using while and a for loop. Actually there are four ways to iterate over any Map in Java, one involves using keySet() and iterating over key and then using get() method to retrieve values, which is bit expensive. Second method involves using entrySet() and iterating over them either by using for each loop or while with Iterator.hasNext() method. This one is a better approach because both key and value objects are available to you during Iteration and you don’t need to call the get() method for retrieving value, which could give O(n) performance in case of huge linked list at one bucket. See my post [4 ways to iterate over Map in Java](http://javarevisited.blogspot.com/2011/12/how-to-traverse-or-loop-hashmap-in-java.html) for detailed explanation and code examples.

**Question 11 : When do you override hashCode() and equals()?**([answer](http://javarevisited.blogspot.com/2013/08/10-equals-and-hashcode-interview.html))

Whenever necessary, especially if you want to do equality check based upon business logic rather than object equality, e.g. two employee objects are equal if they have the same emp\_id, despite the fact that they are two different objects, created by different part of the code. Also overriding both these methods are must if you want to use them as key in HashMap. Now as part of the equals-hashcode contract in Java, when you override equals, you must override hashcode as well, otherwise your object will not break invariant of classes e.g. Set, Map which relies on equals() method for functioning properly. You can also check my post [5 tips on equals in Java](http://javarevisited.blogspot.com/2011/02/how-to-write-equals-method-in-java.html) to understand subtle issue which can arise while dealing with these two methods.

**Question 12 :. What will be the problem if you don’t override hashCode() method ?**([answer](http://java67.blogspot.sg/2013/04/example-of-overriding-equals-hashcode-compareTo-java-method.html))

If you don’t override the equals method, then the contract between equals and hashcode will not work, according to which, two objects which are equal by equals() must have the same hashcode. In this case, an other object may return different hashCode and will be stored on that location, which breaks invariant of HashMap class, because they are not supposed to allow duplicate keys. When you add object using put() method, it iterate through all Map.Entry object present in that bucket location, and update value of previous mapping, if Map already contains that key. This will not work if hashcode is not overridden.

**Question 13 : Is it better to synchronize critical sections of getInstance() method or the whole getInstance() method?**([answer](http://javarevisited.blogspot.com/2014/05/double-checked-locking-on-singleton-in-java.html))

The answer is only the critical section, because if we lock the whole method, then every time some some one call this method, it will have to wait even though we are not creating any object. In other words, synchronization is only needed, when you create object, which happens only once. Once object has created, there is no need for any synchronization. In fact, that’s very poor coding in terms of performance, as synchronized method reduce performance up to 10 to 20 times. Here is UML diagram of Singleton pattern:

By the way, there are several ways to create a thread-safe singleton in Java, which you can also mention as part of this question or any follow-up.

**Question 14: Where does equals() and hashCode() method comes in the picture during the get() operation?**([answer](http://javarevisited.blogspot.com/2011/02/how-hashmap-works-in-java.html))

This core Java interview question is a follow-up of previous Java question and the candidate should know that once you mention hashCode, people are most likely ask, how they are used in HashMap. When you provide a key object, first it’s hashcode method is called to calculate bucket location. Since a bucket may contain more than one entry as linked list, each of those Map.Entry object is evaluated by using equals() method to see if they contain the actual key object or not.

**Questions 15: How do you avoid a deadlock in Java?**([answer](http://javarevisited.blogspot.sg/2010/10/what-is-deadlock-in-java-how-to-fix-it.html))

You can avoid deadlock by breaking the circular wait condition. In order to do that, you can make an arrangement in the code to impose the ordering on acquisition and release of locks. If lock will be acquired in a consistent order and released in just opposite order, there would not be a situation where one thread is holding a lock which is acquired by other and vice-versa. See the detailed answer for the code example and more detailed explanation.

**Question 16:  What is the difference between creating String as new() and literal?**([answer](http://javarevisited.blogspot.com/2012/10/10-java-string-interview-question-answers-top.html))

When we create the string with new() Operator, it’s created in heap and not added into string pool while String created using literal are created in String pool itself which exists in PermGen area of heap.

|  |  |
| --- | --- |
| 1 | String str = new String("Test") |

does not put the object str in String pool, we need to call String.intern() method which is used to put them into String pool explicitly. It’s only when you create String object as a String literal e.g. String s = "Test" that Java automatically puts that into the String pool. By the way there is a catch here Since we are passing arguments as “Test”, which is a String literal, it will also create another object as “Test” on string pool. This is the one point, which has gone unnoticed until knowledgeable readers of Javarevisited blog suggested it. To learn more about the difference between String literal and String object, see [this](http://java67.blogspot.sg/2014/08/difference-between-string-literal-and-new-String-object-Java.html)article.

Here is a nice image which shows this difference quite well:

**Question 17: What is an Immutable Object? Can you write an Immutable Class?**( [answer](http://javarevisited.blogspot.in/2013/03/how-to-create-immutable-class-object-java-example-tutorial.html" \t "_blank))

Immutable classes are Java classes whose objects can not be modified once created. Any modification in Immutable object results in the new object. For example, [String is immutable in Java](http://javarevisited.blogspot.sg/2010/10/why-string-is-immutable-in-java.html). Mostly Immutable classes are also final in Java, in order to prevent sub classes from overriding methods, which can compromise Immutability. You can achieve the same functionality by making member as non-final but private and not modifying them except in constructor.

Apart form obvious, you also need to make sure that, you should not expose the internals of Immutable object, especially if it contains a mutable member. Similarly, when you accept the value for the mutable member from client e.g. java.util.Date, use [clone() method](http://javarevisited.blogspot.sg/2013/09/how-clone-method-works-in-java.html) keep a separate copy for yourself, to prevent the risk of malicious client modifying mutable reference after setting it.

The Same precaution needs to be taken while returning value for a mutable member, return another separate copy to the client, never return original reference held by Immutable class. You can see my post How to create an Immutable class in Java for step by step guide and code examples.

**Question 18: Give the simplest way to find out the time a method takes for execution without using any profiling tool?**([answer](http://javarevisited.blogspot.com/2012/10/10-java-string-interview-question-answers-top.html))

Read the system time just before the method is invoked and immediately after thr method returns. Take the time difference, which will give you the time taken by a method for execution.

To put it in code…

|  |  |
| --- | --- |
| 1 | long start = System.currentTimeMillis (); |
| 2 | method (); |

|  |  |
| --- | --- |
| 3 | long end = System.currentTimeMillis (); |
| 4 |  |

|  |  |
| --- | --- |
| 5 | System.out.println (“Time taken for execution is ” + (end – start)); |

Remember that if the time taken for execution is too small, it might show that it is taking zero milliseconds for execution. Try it on a method which is big enough, in the sense the one which is doing considerable amount of processing

**Question 19: Which two methods you need to implement to use an Object as key in HashMap?**([answer](http://javarevisited.blogspot.com/2013/01/difference-between-identityhashmap-and-hashmap-java.html))

In order to use any object as Key in HashMap or Hashtable, it must implement equals and hash-code methods in Java. Read How HashMap works in Java for a detailed explanation on how equals and hash code method is used to put and get an object from HashMap.

**Question 20: How would you prevent a client from directly instantiating your concrete classes? For example, you have a Cache interface and two implementation classes MemoryCache and DiskCache, How do you ensure there is no object of this two classes is created by client using new() keyword.**

I leave this question for you to practice and think about before I give the answer. I am sure you can figure out the right way to do this, as this is one of the important decision to keep control of classes in your hand, great from a maintenance perspective.

I am also very grateful to my readers who have generously contributed several good questions from Java Interviews for both beginners and experienced developers alike. I have already answered many of these questions in this blog and you can easily find a relevant post by using the search box at the top right corner of this page.