**[10 Equals and HashCode Interview Questions in Java](http://javarevisited.blogspot.ru/2013/08/10-equals-and-hashcode-interview.html" \o "10 Equals and HashCode Interview Questions in Java)**

Equals and HashCode methods in Java are two fundamental methods from java.lang.Object class, which is used to compare equality of objects, primarily inside hash based collections such as Hashtable and HashMap. Both [equals() and hashCode()](http://javarevisited.blogspot.com/2012/07/how-to-override-equals-and-hashcode.html) are defined in java.lang.Object class and there default implementation is based upon Object information e.g. default equals() method return true, if two objects are exactly same i.e. they are pointing to same memory address, while default implementation of [hashcode method](http://javarevisited.blogspot.com/2011/10/override-hashcode-in-java-example.html) return int and implemented as native method. Similar default implementation of toString() method, returns type of class, followed by memory address in [hex String](http://javarevisited.blogspot.com/2013/03/convert-and-print-byte-array-to-hex-string-java-example-tutorial.html). It's advised to override these method, based upon logical and business rules e.g. String overrides equals to check equality of two String based upon content, we have also seen and example implementation of equals() and hashCode for custom classes. Because of there usefulness and usage, they are also very popular in various level of [Java Interviews](http://javarevisited.blogspot.sg/2011/04/top-20-core-java-interview-questions.html), and In this tutorial, I am going to shared some of the really interesting questions from equals() and hashCode() method in Java. This question not only test your concept on both method, but also gives an opportunity to explore them more.

Equals and HashCode Interview questions in Java

[equals and hashcode method interview questions in Java](http://3.bp.blogspot.com/-K6q0DQ1v-tw/TWu8owBtc2I/AAAAAAAAADA/oBoHDBiJ8ag/s1600/17.jpg)Here is my list of 10 interesting questions on both of these methods. I also suggest to read [Effective Java Items](http://www.amazon.com/dp/0321356683/?tag=javamysqlanta-20) on equals() and hashCode() to fill your gaps on knowledge of this two critical methods. I have seen, programmer struggles to write equals() and hashCode() by hands, for a rich class, which contains different data types e.g. int, float, date etc. Reading those items and trying examples will give you enough confidence to face any question on equals and hashCode methods.

**When you are writing equals() method, which other method or methods you need to override?**

hashcode, is the right answer. Since equals and hashCode has there contract, so overriding one and not other, will break contract between them. By the way this question can lead on interesting discussion, if Interviewer likes to go on deep e.g. he may ask about what are those contracts, what happens if those contracts breaks etc. I like to give an example How equals and hashcode are used in hash based collections e.g. [Hashtable](http://javarevisited.blogspot.sg/2012/01/java-hashtable-example-tutorial-code.html), that leaves positive impression more often. You can also mention about [compareTo()](http://javarevisited.blogspot.com/2011/11/how-to-override-compareto-method-in.html) here to score some additional point, this method should also needs to be consistent with equals, which is another interesting question in our list.

**Can two object which are not equal have same hashCode?**

YES, two object, which are not equal by equals() method can still return same hashCode. By the way, this is one of the confusing bit of equals and hashcode contract.

**How does get() method of HashMap works, if two keys has same hashCode?**

This is the follow-up of previous interview questions on equals and hashcode, in fact some time this leads to discussion of the earlier point. When two key return same hashcode, they end-up in same bucket. Now, in order to find the correct value, you used keys.equals() method to compare with key stored in each Entry of linked list there. Remember to point out keys.equals() method, because that's what interviewer is looking for. You can also see here for full [list of interview question on Java HashMap](http://javarevisited.blogspot.com/2011/02/how-hashmap-works-in-java.html).

**Where have you written equals() and hashCode in your project?**

This is to see, if developer has even written these methods or not. Of course almost all of Java programmer are exposed to this, you can point out value objects, Hibernate entities from your domain, where you have overridden equals and hashCode. Always gives examples from your domain and from your project, rather than a trivial example from a test program, because if Interviewer is asking this question, it means he is interested in examples form your domain.

**Suppose your Class has an Id field, should you include in equals()? Why?**

This question is asked to one of my reader as [Hibernate Interview question](http://javarevisited.blogspot.com/2013/05/10-hibernate-interview-questions-answers-java-j2ee-senior.html), well including id is not a good idea in equals() method because this method should check equality based upon content and business rules. Also including id, which is mostly a database identifier and not available to transient object until they are saved into database.

**What happens if equals() is not consistent with compareTo() method?**

This is an interesting questions, which asked along with equals() and hashCode() contract. Some java.util.Set implementation e.g. SortedSet or it's concrete implementation TreeSet uses compareTo() method for comparing objects. If compareTo() is not consistent means doesn't return zero, if equals() method returns true, the it may break Set contract, which is not to avoid any duplicates.

**What happens if you compare an object with null using equals()?**

When a null object is passed as argument to equals() method, it should return false, it must not throw NullPointerException, but if you call equals method on reference, which is null it will throw NullPointerException. That’s why it’s better to use == operator for comparing null e.g. if(object != null) object.equals(anohterObject). By the way if you comparing String literal with another String object than you better call equals() method on literal rather than known object to avoid NPE, one of those simple [tricks to avoid NullPointerException in Java](http://javarevisited.blogspot.com/2013/05/ava-tips-and-best-practices-to-avoid-nullpointerexception-program-application.html).

**What is difference in using instanceof and getClass() method for checking type inside equals?**

This question was asked multiple times, sometime by looking at your equals() and hashCode implementation. Well key difference comes from point that instanceof operator returns true, even if compared with sub class e.g. Subclass instanceof Super class is true, but with getClass() it's false. By using getClass() you ensure that your equals() implementation doesn't return true if compared with sub class object. While if you use instanceof operator, you end up breaking symmetry rule for equals which says that if a.equals(b) is true than b.equals(a) should also be true. Just replace a and b with instance of Super class and Sub class, and you will end up breaking symmetry rule for equals() method.

**How do you avoid NullPointerException, while comparing two Strings in Java?**

Since when compared to null, equals return false and doesn't throw NullPointerException, you can use this property to avoid NPE while using comparing String. Suppose you have a known String "abc" and you are comparing with an unknown String variable str, then you should call equals as "abc".equals(str), this will not throw [Exception in thread Main: java.lang.NullPointerException](http://javarevisited.blogspot.com/2012/06/common-cause-of-javalangnullpointerexce.html), even if str is null. On the other hand if you call str.equals("abc"), it will throw NPE. So be careful with this. By the way this is one of the Java coding best practices, which Java developer should follow, while using equals() method.

**What is difference between "==" and equals() method in Java?**

One of the most classical interview question on equals(). It has been asked numerous times during in past decade. I have also covered this question already. See [here](http://javarevisited.blogspot.com/2012/07/auto-boxing-and-unboxing-in-java-be.html) for detailed discussion on how it affect equality checking of String and Integer in auto boxing world.

That's all on this **list of Java interview Questions on Equals and HashCode methods in Java**. It's one of the fundamental concept of Java programming language, but yet has several subtle things, which is unknown to many Java programmers. I strongly suggest to get yourself really good on equals(), hashCode(), compareTo() and compare() method, not only to do well on Java Interviews, but also to write correct code in Java.

Read more: <http://javarevisited.blogspot.com/2013/08/10-equals-and-hashcode-interview.html#ixzz2lbQn5j44>