**Tricky Java interview questions**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

[**http://java67.blogspot.sg/2012/09/top-10-tough-core-java-interview-questions-answers.html**](http://java67.blogspot.sg/2012/09/top-10-tough-core-java-interview-questions-answers.html)

**Top 10 tough core Java interview questions answers programming**

**tough core Java interview questions and answers**

What is tough core java interview question ? Why do people look for tough Java questions before going for interview? well I don't thing I need to answer these tough questions because its pretty natural to prepare for tough questions even if you are not expecting tough questions from core Java. If you are prepare for tough and [tricky Java interview question](http://java67.blogspot.sg/2012/08/10-advanced-core-java-interview.html) than you feel more confident and answer other Java interview question with confidence. On the other hand if you are not prepare for tough and tricky core Java questions than seeing them on Java interview or written test may surprise you. But definition of [tough core Java questions](http://javarevisited.blogspot.sg/2011/07/java-multi-threading-interview.html) is not universal, same Java question which is easy for one programmer might be tough for other Java programmer. That's why it's best to prepare your own list of tough interview questions before appearing on any Java job interview. In this article I am going to share you with my **Top 10 tough core Java interview questions and answers**, which may help you in Java interview.

## 10 tough Java interview question and answer

Here is my list of *10 tough or tricky Java interview questions*. These questions are mostly from Core Java and I have not included [J2EE questions](http://javarevisited.blogspot.sg/2011/09/servlet-interview-questions-answers.html). As I said you may know answers of these tough Java question or you may not even find it tough enough to challenge your Java knowledge but once upon a time these were asked in various Java interview and many programmer including my friends and colleagues finds them tough to answer.

**Why wait and notify is declared in Object class instead of Thread ?**

Another [tough java question](http://javarevisited.blogspot.sg/2011/04/top-20-core-java-interview-questions.html), how can you answer this question if you are not designed Java programming language. anyway some common sense and deep knowledge of Java programming helps to answer such tough core java interview question. See this blog post to learn  [Why wait and notify is declared in Object class and not in Thread](http://javarevisited.blogspot.sg/2012/02/why-wait-notify-and-notifyall-is.html).

**Why multiple inheritance is not supported in Java ?**

I found this *core Java question* really tough to answer because your answer may not satisfy Interviewer, in most cases Interviewer is looking for specific points and if you can bring them, they would be happy. Key to answer this kind of tough question in Java is to prepare topic well to accommodate any follow-ups. See [Why multiple inheritance is not supported in Java](http://javarevisited.blogspot.sg/2011/07/why-multiple-inheritances-are-not.html) for answer of this tough Java question.

**Why Java does not support operator overloading ?**

One more similar category of  tough Java question. C++ supports operator overloading than why not Java? this is the argument Interviewer will give to you and some time even say that + operator is overloaded in Java for String concatenation, Don't be fooled with such arguments. See  [Why support operator overloading is not supported in Java](http://javarevisited.blogspot.sg/2011/08/why-java-does-not-support-operator.html) for detailed answer of this tricky Java question.

**Why String is immutable in Java?**

My favorite Java interview question, this is tough, tricky but same time very useful as well. Some interviewer also ask this question as Why String is final in Java. look at this post for some points which make sense on [Why String is final or immutable in Java](http://javarevisited.blogspot.sg/2010/10/why-string-is-immutable-in-java.html)

**Why char array is preferred to store password than String in Java?**

Another [tricky Java question](http://javabuddy.hubpages.com/hub/10-most-tricky-question-in-java) which is based on String and believe me there are only few Java programmer which can answer this question correctly. This is a real *tough core Java interview question* and again solid knowledge of String is required to answer this. see [Why char array is better than String for storing password in Java](http://javarevisited.blogspot.sg/2012/03/why-character-array-is-better-than.html) to find out answer of this tough Java question.

**How to create thread-safe singleton in Java using double checked locking?**

This Java question is also asked as What is [thread-safe](http://javarevisited.blogspot.sg/2012/01/how-to-write-thread-safe-code-in-java.html) singleton  and how to do you write it. Well Singleton created with double checked locking before Java 5 was broker and its possible to have multiple instance of Singleton if multiple thread try to create instance of Singleton at same time. from Java 5 its easy to [create thread safe Singleton using Enum](http://javarevisited.blogspot.gr/2012/07/why-enum-singleton-are-better-in-java.html). but if interviewer persist with double checked locking then you have to write that code for them. remember to use volatile variable.  See [10 Java singleton interview question](http://javarevisited.blogspot.sg/2011/03/10-interview-questions-on-singleton.html) for more details on this topic.

**Write Java program to create deadlock in Java and fix it ?**

One of the classical but t*ough core Java interview question* and you are likely to fail if you have not involved in coding of multi-threaded concurrent Java application. See  [how to create and prevent deadlock in Java](http://javarevisited.blogspot.sg/2010/10/what-is-deadlock-in-java-how-to-fix-it.html) for complete answer of  this tough core Java interview question

**What happens if your Serializable class contains a member which is not  serializable? How do you fix it?**

Any attempt to Serialize that class will fail with NotSerializableException, but this can be easily solved by making that variable [transient](http://javarevisited.blogspot.sg/2011/09/transient-keyword-variable-in-java.html) for static in Java. See [Top 10 Serialization interview question answers in Java](http://javarevisited.blogspot.sg/2011/04/top-10-java-serialization-interview.html) for more details.

**Why wait and notify  called from synchronized method in Java?**

Another tough core Java question for wait and notify. They are called from [synchronized method or synchronized block](http://javarevisited.blogspot.sg/2011/04/synchronization-in-java-synchronized.html) because wait and modify need monitor on Object on which wait or notify get called. See  [Why wait and notify require synchronized context](http://javarevisited.blogspot.sg/2011/05/wait-notify-and-notifyall-in-java.html) for complete answer of this *tough and tricky Java multi-threading question*.

**Can you override static method in Java? if I create same method in subclass is it compile time error?**

No you can not override [static method in Java](http://javarevisited.blogspot.sg/2011/11/static-keyword-method-variable-java.html) but its not a compile time error to declare exactly same method in sub class, That is called method hiding in Java. See  [Can you override static method in Java](http://java67.blogspot.sg/2012/08/can-we-override-static-method-in-java.html) for complete answer of this tough Java interview question.

These were my list of **tough core Java interview question and answers**. Some of the Java question doesn't look that tough for experience programmer but they are really *tricky* to answer for intermediate and beginners in Java. by the way if you have faced any tough Java question in interview then please share with us.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<http://java67.blogspot.sg/2012/09/top-10-tricky-java-interview-questions-answers.html>

### Top 10 Tricky Java interview questions and Answers

What is a tricky question? Well, tricky Java interview questions are those questions which has some surprise element on it. If you try to answer a tricky question with common sense, you will most likely fail because they require some specific knowledge. Most of the *tricky Java questions* comes from confusing concepts like [function overloading and overriding](http://java67.blogspot.sg/2012/09/difference-between-overloading-vs-overriding-in-java.html),  Multi-threading which is really tricky to master, character encoding, [checked vs unchecked exceptions](http://java67.blogspot.sg/2012/12/difference-between-runtimeexception-and-checked-exception.html) and subtle Java programming details like **Integer overflow**. Most important thing to answer a tricky Java question is attitude and *analytical thinking* , which helps even if you don't know the answer. Anyway in this Java article we will see 10 Java questions which is real tricky and requires more than average knowledge of Java programming language to answer them correctly. As per my experience there are always one or two tricky or [tough Java interview question](http://java67.blogspot.sg/2012/09/top-10-tough-core-java-interview-questions-answers.html) on any core Java or [J2EE interviews](http://javarevisited.blogspot.sg/2011/09/servlet-interview-questions-answers.html), so its good to prepare tricky questions from Java in advance. If I take interview, I purposefully put these kind of question to gauge depth of candidate's understanding in Java. Another advantage of asking such question is surprise element, which is key factor to put candidate on some pressure during interviews. Since these questions are less common, there is good chance that many Java developer doesn't know about it.

## 10 Tricky Java interview question - Answered

Here is my list of 10 tricky Java interview questions, Though I have prepared and shared lot of difficult core Java interview question and answers, But I have chosen them as Top 10 tricky questions because you can not guess answers of this tricky Java questions easily, you need some subtle details of Java programming language to answer these questions.

**Question: What does the following Java program print?**

public class Test {

public static void main(String[] args) {

System.out.println(Math.min(Double.MIN\_VALUE, 0.0d));

}

}

Answer: This questions is tricky because unlike the [Integer](http://java67.blogspot.sg/2013/03/how-to-convert-java-string-to-int-or.html), where MIN\_VALUE is negative, both the MAX\_VALUE and MIN\_VALUE of the Double class are positive numbers. The Double.MIN\_VALUE is 2^(-1074), a double constant whose magnitude is the least among all double values. So unlike the obvious answer, this program will print 0.0 because Double.MIN\_VALUE is greater than 0. I have asked this question to Java developer having experience up to 3 to 5 years and surprisingly almost 70% candidate got it wrong.

**Question : What will happen if you put return statement or System.exit () on try or catch block ? Will finally block execute?**

This is a very *popular tricky Java question* and its tricky because many programmer think that no matter what, but finally block will always execute. This question challenge that misconcept by putting return statement in try or catch block or calling System.exit from try or catch block. Answer of this tricky question in Java is that finally block will execute even if you put return statement in try block or catch block but finally block won't run if you call System.exit form try or catch.

**Can you override private or static method in Java ?**

Another popular Java tricky question, As I said method overriding is a good topic to ask trick questions in Java. Anyway, [you can not override private or static method in Java](http://java67.blogspot.sg/2012/08/can-we-override-static-method-in-java.html), if you create similar method with same return type and same method arguments in child class then it will hide the super class method, this is known as method hiding. Similarly you cannot override private method in sub class because it's not accessible there, what you do is create another private method with same name in child class. See [Can you override private method in Java](http://java67.blogspot.sg/2012/08/can-we-override-private-method-in-java.html) or more details.

**Question: What does the the expression 1.0 / 0.0 will return? will it throw Exception? any compile time error?**

Answer : This is another tricky question from Double class. Though Java developer knows about double primitive type and Double class, while doing floating point arithmetic they don't pay enough attention to Double.INFINITY, NaN, and -0.0 and other rules that govern the arithmetic calculations involving them. Simple answer to this question is that it will not throw ArithmeticExcpetion and return Double.INFINITY. Also note that the comparison x == Double.NaN always evaluates to false, even if x itself is a NaN. To test if x is a NaN, one should use the method call Double.isNaN(x) to check if given number is NaN or not. If you know SQL, this is very close to NULL there.

**Does Java support multiple inheritance ?**

This is the trickiest question in Java, if C++ can support direct multiple inheritance than why not Java is the argument Interviewer often give. Answer of this question is much more subtle then it looks like, because Java does support multiple inheritance of Type by allowing interface to extend other interfaces, what Java doesn't support is multiple inheritance of implementation. This distinction also get blur because of default method of Java 8, which now provides Java, multiple inheritance of behaviour as well. See [Why multiple inheritance is not supported in Java](http://javarevisited.blogspot.sg/2011/07/why-multiple-inheritances-are-not.html) to answer this tricky Java question.

**What will happen if we put a key object in a HashMap which is already there ?**

This tricky Java questions is part of another frequently asked question, How HashMap works in Java. HashMap is also a popular topic to create confusing and tricky question in Java. Answer of this question is, if you put the same key again than it will replace the old mapping because HashMap doesn't allow duplicate keys. Same key will result in same hashcode and will end up at same position in bucket. Each bucket contains a linked list of Map.Entry object, which contains both Key and Value. Now Java will take Key object form each entry and compare with this new key using equals() method, if that return true then value object in that entry will be replaced by new value. See [How HashMap works in Java](http://java67.blogspot.sg/2013/06/how-get-method-of-hashmap-or-hashtable-works-internally.html) for more tricky Java questions from HashMap.

**Question : What does the following Java program print?**

public class Test {

public static void main(String[] args) throws Exception {

char[] chars = new char[] {'\u0097'};

String str = new String(chars);

byte[] bytes = str.getBytes();

System.out.println(Arrays.toString(bytes));

}

}

Answer: The trikyness of this question lies on character encoding and how String to byte array conversion works. In this program, we are first creating a String from a character array, which just has one character '\u0097', after than we are getting byte array from that String and printing that byte. Since \u0097 is within the 8-bit range of byte primitive type, it is reasonable to guess that the str.getBytes() call will return a byte array that contains one element with a value of -105 ((byte) 0x97). However, that's not what the program prints and that's why this question is tricky. As a matter of fact, the output of the program is operating system and locale dependent. On a Windows XP with the US locale, the above program prints [63], if you run this program on Linux or Solaris, you will get different values.

To answer this question correctly, you need to know about how Unicode characters are represented in Java char values and in Java strings, and what role character encoding plays in String.getBytes(). In simple word, t[o convert a string to a byte array](http://javarevisited.blogspot.sg/2014/08/2-examples-to-convert-byte-array-to-String-in-Java.html), Java iterate through all the characters that the string represents and turn each one into a number of bytes and finally put the bytes together. The rule that maps each Unicode character into a byte array is called a character encoding. So It's possible that if same character encoding is not used during both encoding and decoding then retrieved value may not be correct. When we call str.getBytes() without specifying a character encoding scheme, the JVM uses the default character encoding of platform to do the job. The default encoding scheme is operating system and locale dependent. On Linux, it is UTF-8 and on Windows with a US locale, the default encoding is Cp1252. This explains the output we get from runing this program on Windows machines with a US locale. No matter which character encoding scheme is used, Java will always translate Unicode characters not recognized by the encoding to 63, which represents the character U+003F (the question mark, ?) in all encodings.

**If a method throws NullPointerException in super class, can we override it with a method which throws RuntimeException?**

One more tricky Java questions from overloading and overriding concept. Answer is you can very well throw super class of RuntimeException in overridden method but you can not do same if its checked Exception. See [Rules of method overriding in Java](http://javarevisited.blogspot.sg/2011/12/method-overloading-vs-method-overriding.html) for more details.

**What is the issue with following implementation of compareTo() method in Java**

public int compareTo(Object o){

Employee emp = (Employee) emp;

return this.id - o.id;

}

**where id is an integer number ?**

Well three is nothing wrong in this Java question until you guarantee that id is always positive. This Java question becomes tricky when you can't guaranteed that id is positive or negative. tricky part is, If id becomes negative than **subtraction may overflow** and produce incorrect result. See [How to override compareTo method in Java](http://javarevisited.blogspot.sg/2011/11/how-to-override-compareto-method-in.html) for complete answer of this Java tricky question for experienced programmer.

**How do you ensure that N thread can access N resources without deadlock**

If you are not well versed in writing multi-threading code then this is real tricky question for you. This Java question can be tricky even for experienced and senior programmer, who are not really exposed to deadlock and race conditions. Key point here is order, if you acquire resources in a particular order and release resources in reverse order you can prevent deadlock. See [how to avoid deadlock in Java](http://javarevisited.blogspot.sg/2010/10/what-is-deadlock-in-java-how-to-fix-it.html) for a sample code example.

**Question : Consider the following Java code snippet, which is initializing two variables and both are not volatile, and two threads T1 and T2 are modifying these values as following, both are not synchronized**

int x = 0;

boolean bExit = false;

Thread 1 (not synchronized)

x = 1;

bExit = true;

Thread 2 (not synchronized)

if (bExit == true)

System.out.println("x=" + x);

**Now tell us, is it possible for Thread 2 to print “x=0”?**

Answer: It's impossible for a list of tricky Java questions to not contain anythign from multi-threading. This is the simplest one I can get. Answer of this question is Yes, It's possible that thread T2 may print x=0.Why? because without any instruction to compiler e.g. synchronized or volatile, bExit=true might come before x=1 in compiler reordering. Also x=1 might not become visible in Thread 2, so Thread 2 will load x=0. Now, how do you fix it? When I asked this question to couple of programmers they answer differently, one suggest to make both thread synchronized on a common mutex, another one said make both variable volatile. Both are correct, as it will prevent reordering and guarantee visibility. But best answer is you just need to make bExit as volatile, then Thread 2 can only print “x=1”. x does not need to be volatile because x cannot be reordered to come after bExit=true when bExit is volatile.

**What is difference between CyclicBarrier and CountDownLatch in Java**

Relatively newer Java tricky question, only been introduced form Java 5. Main difference between both of them is that you can reuse CyclicBarrier even if Barrier is broken but you can not reuse CountDownLatch in Java. See [CyclicBarrier vs CountDownLatch in Java](http://java67.blogspot.sg/2012/08/difference-between-countdownlatch-and-cyclicbarrier-java.html) for more differences.

**What is difference between StringBuffer and StringBuilder in Java ?**

Classic Java questions which some people think tricky and some consider very easy. StringBuilder in Java was introduced in JDK 1.5 and only difference between both of them is that StringBuffer methods e.g. length(), capacity() or append() are [synchronized](http://javarevisited.blogspot.sg/2011/04/synchronization-in-java-synchronized.html) while corresponding methods in StringBuilder are not-synchronized. Because of this fundamental difference, concatenation of String using StringBuilder is faster than StringBuffer. Actually its considered bad practice to use StringBuffer any more, because in almost 99% scenario, you perform string concatenation on same thread. See [StringBuilder vs StringBuffer](http://javarevisited.blogspot.sg/2011/07/string-vs-stringbuffer-vs-stringbuilder.html) for more differences.

**Can you access non static variable in static context?**

Another tricky Java question from Java fundamentals. No you can not access non-static variable from static context in Java. If you try, it will give compile time error. This is actually a common problem beginners in Java face, when they try to access instance variable inside main method. Because main is static in Java, and instance variables are non-static, you can not access instance variable inside main. Read [why you can not access non-static variable from static method](http://javarevisited.blogspot.sg/2012/02/why-non-static-variable-cannot-be.html) to learn more about this tricky Java questions.

Now, its practice time, here are some questions for you guys to answer, these are contributed by readers of this blog, big thanks to them.

1. When Singleton doesn't remain Singleton in Java?
2. is it possible to load a class by two ClassLoader?
3. is it possible for equals() to return false, even if contents of two Objects are same?
4. Why compareTo() should be consistent to equals() method in Java?
5. When do Double and BigDecimal give different answers for equals() and compareTo() == 0.
6. How does "has before" apply to volatile work?
7. Why is 0.1 \* 3 != 0.3,
8. Why is (Integer) 1 == (Integer) 1 but (Integer) 222 != (Integer) 222 and which command arguments change this.
9. What happens when exception is thrown by a Thread?
10. Difference between notify() and notifyAll() call?
11. Difference between System.exit() and System.halt() method?
12. Does following code legal in Java? is it example of method overloading or overriding?

public String getDescription(Object obj){

return obj.toString;

}

public String getDescription(String obj){

return obj;

}

and

public void getDescription(String obj){

return obj;

}

This was my list of **Some of the most common tricky question in Java** . It's not a bad idea to prepare tricky Java question before appearing for any core Java or J2EE interview. One or two open ended or tricky question is quite common in Java interviews.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<http://java67.blogspot.sg/2012/08/10-java-coding-interview-questions-and.html>

Top 10 Java Programming Coding Interview Questions Answers for programmers

**Java coding interview questions answers**

Java coding interview questions are mostly some [programming exercises](http://javarevisited.blogspot.sg/2011/06/top-programming-interview-questions.html) which is asked to be solved in java programming language in limited time during any core Java or J2EE interview. No matter whether you have 2 years of experience or 4 years of experience, There is always some c*oding interview question in Java* programming job you are applying. Actually questions related to Java Coding are increasingly getting popular on Java interviews for two reasons, first its difficult to crack Java coding interview questions than answering fact based questions like [Why String is immutable in Java](http://javarevisited.blogspot.sg/2010/10/why-string-is-immutable-in-java.html) or [Why main is static in Java](http://javarevisited.blogspot.sg/2011/12/main-public-static-java-void-method-why.html); Second reason of popularity of Coding question in Java interviews is read need of good developers who are required to do lot of coding in projects rather than doing some maintenance works. Since Java is front line language for any server side application, and as complexity of business process and need of performance is increasing,  its obvious that amount of coding skill required in Java programmers are getting increased with every passing days, which effectively mean Java coding questions are top of any list of [core Java interview question](http://javarevisited.blogspot.sg/2011/04/top-20-core-java-interview-questions.html). These *Coding interview questions* are collected from various Java programming interviews, from friends and colleagues and can be a good starting point to refresh your coding skills before appearing on any Java interviews. These basics Java Programs and logical questions can also be a good resources for learning programming and to improve your problem solving skills in Java.

### 10 Programming and Coding Interview questions answers in Java

Here are my list of 10 Java coding interview questions and answers, which is good to prepare before appearing on any Java interviews. As I said Java coding questions are mostly based on programming, logical analysis and problem solving skill and are on  top of any list of [tough Java interview questions](http://java67.blogspot.sg/2012/09/top-10-tough-core-java-interview-questions-answers.html), so better to get it right in first place. Any way you may be able to solve and find answers of these Java coding questions by yourself, but if you stuck do a google, and you can get many alternative ways to solve these problem. Some times knowing more than one way to solve any programming question or coding problem in Java also helps to impress interviewer. This list mainly contains basic programs asked on Interviews.

**Write a Java program to replace certain characters from String like**

public String replace(String str, char ch)

This is a [tricky Java coding interview question](http://javabuddy.hubpages.com/hub/10-most-tricky-question-in-java) is asked in one of the written test my friend had appeared recently. This Java coding question can be solved in multiple way e.g. by using charAt() or subString() method,  but any approach throws couple of follow-up question e.g. you may be asked to write two version to solve this coding exercise, one by using recursion and other by using Iteration. They may also ask you to write [JUnit test](http://javarevisited.blogspot.sg/2012/08/best-practices-to-write-junit-test.html) for this function which means handling null, empty string etc. By the way this programming question is quite common on technical interviews not just Java but also C, C++ or Scala, but knowing API definitely helps to produce better solution quickly.

**Write a Java program to print Fibonacci series upto 100?**

This is one of the most *popular coding interview question asked in Java programming language*. Even though, Writing program for Fibonacci series is one of the basic Java program, not every Java developer get it right in interview. Again interview can ask to solve this programming interview question, by using recursion or Iteration. This Java programming question also test your problem solving skills and if you come up with an original solution, that may even help. See here for complete [code example of Fibonacci series in Java](http://java67.blogspot.sg/2012/07/java-program-fibonacci-series-with.html)

**FizzBuzz problem :** Write a Java program that prints the numbers from 1 to 50. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz"

This is also one of the classical programming questions, which is asked on any Java programming or technical interviews. This questions is very basic but can be very trick for programmers, who can't code, that's why it is used to differentiate programmers who can do coding and who can't. Here is a sample Java program to solve FizzBuzz problem :

**public** **class** FizzBuzzTest{

**public** **static** **void** main(String args[]){

**for**(**int** i = **1**; i <= **50**; i++) {

**if**(i % (**3**\***5**) == **0**) System.out.println("FizzBuzz");

**else** **if**(i % **5** == **0**) System.out.println("Buzz");

**else** **if**(i % **3** == **0**) System.out.println("Fizz");

**else** System.out.println(i);

}

}

}

**Write a Comparator in Java to compare two employees based upon there name, departments and age?**

This is pure Java based Coding exercise. In order to solve this Java coding or programming interview question you need to know [What is comparator in Java](http://javarevisited.blogspot.sg/2011/06/comparator-and-comparable-in-java.html) and How to use compare method in Java for sorting Object. Sorting is one of the most logical and practical question on technical interview and ability to sort Java object is must to code in Java.  This article help you to solve this Java coding question by explaining [how to sort object in Java using Comparable and Comparator](http://java67.blogspot.com/2012/10/how-to-sort-object-in-java-comparator-comparable-example.html). Just remember that Comparable has compareTo() method and use to sort object based upon there natural order e.g. numeric order for number, and alphabetic order for String, while Comparator can define any arbitrary sorting. A good followup question can also be difference between Comparator and Comparable in Java, so be ready for that.

**Design vending machine in Java which vends Item based upon four denomination of coins and return coin if there is no Item.**

This kind of Java coding interview question appear in written test and I believe if you get it right, you are almost through the Interview. These kind of problem solving questions in Java are not easy, you need to design , developer and write JUnit test within 2 to 3 hours and only good Java developers, with practical coding experience can solve this kind of Java programming question. What helps you is to keep practicing your coding skill even before interview. See this [programming exercise in Java](http://java67.blogspot.com/2013/01/10-programming-questions-and-exercises.html) to get yourself going. I personally like to ask programming questions, which test your object oriented design skills e.g. designing ATM machine, designing parking lot or implementing logic for Traffic Signal controller.

**Write a Java program to check if a number is Armstrong or not ?**

Another popular *logical coding interview questions* in Java, which is based on programming logic. In order to answer this programming question, you need to know what is Armstrong number, but that is not a problem because question may specify that and even provide sample input and output. Key thing to demonstrate is logic to check if a number is Armstrong or not. In most cases, you can not use utility methods defined by logic and you need to produce logic by yourself by using basic operators and methods. By the way this is also one of the basic programming questions and I have already provided a solution for this. I suggest to see this [Java program to find Armstrong Number in Java](http://java67.blogspot.sg/2012/07/java-program-to-find-armstrong-numbers.html)  to answer this coding question

**Write a Java program to prevent deadlock in Java ?**

Some of the programming or coding interview question is always based on fundamental feature of Java programming language e.g. multi-threading, synchronization etc. Since writing deadlock proof code is important for a Java developer, programming questions which requires knowledge of concurrency constructs becomes popular coding question asked in Java Interviews. Deadlock happens if four condition is true e.g. mutual exclusion, no waiting, circular wait and no preemption. If you can break any of this condition than you can create Java programs,which are deadlock proof. One easy way to avoid deadlock is by imposing an ordering on acquisition and release of locks. You can further check [How to fix deadlock in Java](http://javarevisited.blogspot.sg/2010/10/what-is-deadlock-in-java-how-to-fix-it.html)  to answer this Java programming questions with coding in Java

**Write Java program to reverse String in Java without using API functions ?**

Another classic *Java programming or coding exercis*e mostly asked on 2 to 5 years experienced Java interviews. Despite being simple answering this coding question is not easy, specially if you are not coding frequently. Its best to prepare this programming question in advance to avoid any embarrassment during interviews. I suggest to see this post which shows [How to reverse String using recursion in Java](http://javarevisited.blogspot.sg/2012/01/how-to-reverse-string-in-java-using.html)

**Write a Java program to find if a number is prime number or not**

One more basic Java program, which made it's way to Interviews. One of the simplest coding question and also a very popular Java programming exercise. Beauty of these kinds of logical questions is that, they can really test basic programming skills or a coder, programmer or developer. Not just problem solving, you can also check there coding style and thought process. By the way. you can  check this [article](http://javarevisited.blogspot.sg/2012/04/java-program-to-print-prime-numbers-in.html) for answer of this Java coding interview question.

**How to Swap two numbers without using third variable in Java?**

This Java program might require just four lines to code, but it's worth preparing. Most of the programmers make same kind of mistakes, while writing solution for this program e.g. Integer overflow, they tend to forget that integer can overflow if it's limit exceeded, which is not very big. Sure shot way to answer this programming questions is to use [XOR trick to swap numbers](http://javarevisited.blogspot.com/2013/02/swap-two-numbers-without-third-temp-variable-java-program-example-tutorial.html), as mentioned in that blog post.

**Create a Java program to find middle node of linked list in Java in one pass?**

Any list of programming questions is incomplete without any questions from linked list, arrays and string, these three forms bulk of coding questions asked on Java interviews. Trick to solve this problem is to remember that last node of linked list points to null and you can trade memory with speed. Sometime your approach to come to two pointer solution really matters, by taking rational steps as mentioned above, you can sound more intelligent, problem solver and genuine. Quick solution of this programming question can be found [here](http://javarevisited.blogspot.com/2012/12/how-to-find-middle-element-of-linked-list-one-pass.html).

**How to find if a linked list contains cycle or not in Java?**

Another programming question based on linked list. By the way this coding question is bit tricky than previous one, but this can also be solved using two pointer approach. If linked list has cycle, than fast pointer will either catch slow pointer or point to null. See [Java program to check if linked list contains loop in Java](http://javarevisited.blogspot.com/2013/05/find-if-linked-list-contains-loops-cycle-cyclic-circular-check.html) for complete solution of this coding interview question.

**Implement Producer Consumer design Pattern in Java using wait, notify and notifyAll method in Java?**

Similar to deadlock related programming interview question, this is also used to test programmers ability to write bug free concurrent programs in Java. This coding questions can be difficult if you haven't used wait and notify before, you can confuse yourself as hell on different aspect e.g. which condition to check, on which lock you should synchronized etc. I suggest following [here](http://java67.blogspot.com/2012/12/producer-consumer-problem-with-wait-and-notify-example.html) to answer this multithreading based programming interview question.

**Write a Java program to calculate Factorial of a number in Java?**

This Java coding interview questions is also based on list of basic Java programs for beginners. As usual, you better remember how to calculate factorial and how to code solution using loop and recursive method calls. For complete code solution of this programming question, see [Java program to calculate factorial](http://javarevisited.blogspot.com/2012/04/java-program-to-find-factorial-of.html)

These are some of the **Java coding interview questions** **and answers**, which appears frequently on Java Programming interviews. I have included links, with some of my blog posts, which discusses answers of these Java coding question, but you can also find answers by doing google yourself. Please share  what kind of Programming, logical, Problem solving or coding related questions, asked to you in Java interviews?

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<http://java67.blogspot.sg/2012/09/java-collection-interview-questions.html>

Java Collection interview Questions Answers

Java Collection Framework interview questions and answers contains questions from popular Java collection classes e.g. HashMap, [ArrayList](http://javarevisited.blogspot.sg/2011/05/example-of-arraylist-in-java-tutorial.html), HashSet, [ConcurrentHashMap](http://javarevisited.blogspot.sg/2011/04/difference-between-concurrenthashmap.html) and legacy collection classes like Vector and [Hashtable](http://javarevisited.blogspot.sg/2012/01/java-hashtable-example-tutorial-code.html). Interview questions from Java Collection framework is one of the most asked concept on any [Core Java interview](http://javarevisited.blogspot.sg/2011/04/top-20-core-java-interview-questions.html) or [J2EE interview](http://javarevisited.blogspot.sg/2011/09/servlet-interview-questions-answers.html) at both beginners, experienced and intermediate level e.g. 2 to 4 years experienced Java programmer. In order to answer interview questions from Java collection framework, You need a good knowledge of various Collection interfaces e.g. Map, List and Set and familiarities with there frequently used implementation e.g. HashMap, Hashtable, [TreeMap](http://javarevisited.blogspot.sg/2011/12/treemap-java-tutorial-example-program.html)  from Map, ArrayList, LinkedList and Vector for List interface and [HashSet](http://javarevisited.blogspot.com/2012/06/hashset-in-java-10-examples-programs.html), TreeSet of Set interface. Questions related to Iterating collection , modifying Collection and thread-safety of Collection is also popular area from where **Java collection interview Question**s asked during interview. In this List of *Java collection interview questions and answers*, we will see questions from all above area. I have also included some questions from new Concurrent collection classes introduced in Java 5 e.g. ConcurrentHashMap and CopyOnWriteArrayList.

## Java Collection interview Questions Answers

Here is list of my favorite*,* [*frequently asked Questions from Java collection framework*](http://javarevisited.blogspot.sg/2011/11/collection-interview-questions-answers.html). Almost all of these questions have appeared in Java interview at various level ranging from Junior to Senior software engineer level at different Companies e.g. Capegemini, Tech Mahindra, TCS , Satyam and CTS.

**What is Difference between Hashtable and HashMap in Java?**

This Java collection interview questions is I guess most popular one. Most of Java programmer who has at least 2 years of experience has seen this question on core Java or J2EE interview. Well there are many difference between them but most important is [thread-safety](http://javarevisited.blogspot.sg/2012/01/how-to-write-thread-safe-code-in-java.html), HashMap is not thread-safe while Hashtable is thread-safe collection. See [Hashtable vs HashMap in Java](http://java67.blogspot.sg/2012/08/5-difference-between-hashtable-hashmap-Java-collection.html) for more differences between HashMap and Hashtable in Java.

**What is difference between Hashtable and ConcurrentHashMap in Java?**

Another frequently asked Java collection interview question post Java 5 world which introduced Concurrent Collection classes like ConcurrentHashMap and CopyOnWriteArrayList along with Concurrency utilities e.g. [CyclicBarrier](http://javarevisited.blogspot.sg/2012/07/cyclicbarrier-example-java-5-concurrency-tutorial.html) and [CountDownLatch](http://javarevisited.blogspot.sg/2012/07/countdownlatch-example-in-java.html). Well both Hashtable and ConcurrentHashMap are thread-safe here but later provides more scalability than former. See [Difference between ConcurrentHashMap and Hashtable in Java](http://javarevisited.blogspot.sg/2011/04/difference-between-concurrenthashmap.html) for answer of this Java collection interview question.

**What is Difference between Iterator and Enumeration in Java?**

One of the classic interview Questions asked on Java collection framework, This is pretty old and programmer who has been working in Java for 4 to 6 years must have seen this question before. Well [Iterator and ListIterator in Java](http://javarevisited.blogspot.sg/2011/10/java-iterator-tutorial-example-list.html) is a new way to iterator collection in Java and provides ability to remove object while traversing while Enumeration doesn't allow you to remove object. See [Iterator vs Enumeration in Java](http://javarevisited.blogspot.sg/2010/10/what-is-difference-between-enumeration.html) for more differences between both of them.

**What is Difference between fail-safe and fail-fast Iterator in Java?**

This is relatively new Java collection interview question because concept of fail-safe iterator is come along with ConcurrentHashMap and CopyOnWriteArrayList. See [Difference between fail-safe and fail-fast Iterator in Java](http://javarevisited.blogspot.sg/2012/02/fail-safe-vs-fail-fast-iterator-in-java.html) for answer of this Java collection question.

**How HashMap works internally in Java?**

One of the most frequently asked Java interview question to experience Java programmer of 4 to 5 years of experience. I have seen this question on big companies like Morgan Stanley, JP Morgan, Nomura and other banks e.g. Barclays capital. See [How HashMap works internally in Java](http://javarevisited.blogspot.com/2011/02/how-hashmap-works-in-java.html) for detailed answer of this Java collection interview question.

**Can you write code to traverse Map in Java on 4 ways?**

Another Java collection question which appear as part of [Java Coding interview question](http://java67.blogspot.sg/2012/08/10-java-coding-interview-questions-and.html) and appeared in many interviews. As you know there are multiple ways to traverse or iterate Map in Java e.g. for loop, while loop using Iterator etc. [4 ways to iterator Map in Java](http://javarevisited.blogspot.sg/2011/12/how-to-traverse-or-loop-hashmap-in-java.html) has detailed explanation and sample code which is sufficient to answer this Java collection framework interview question.

**What is difference between Vector and ArrayList in Java?**

Along with [Difference between HashMap and hashtable](http://javarevisited.blogspot.sg/2010/10/difference-between-hashmap-and.html), this Java collection interview question is probably second in the list of frequently asked question on Java collection framework. Both ArrayList and Vector implements List interface from Java 4 but they have differences including synchronization, See [difference between Vector and ArrayList in Java](http://javarevisited.blogspot.sg/2011/09/difference-vector-vs-arraylist-in-java.html) for complete answer of this collection interview question in Java.

**What is difference between ArrayList and LinkedList in Java?**

A follow-up question which is asked in response to previous Java collection interview question. Here also both LinkedList and ArrayList are List implementation but there internal data-structure is different, one is derived from Array while other is derived from LinkedList. See [LinkedList vs ArrayList in Java](http://javarevisited.blogspot.sg/2012/02/difference-between-linkedlist-vs.html) to answer this Java Collection interview question.

**What is difference between List and Set in Java ?**

List vs Set is one of the most important concept to understand in Java Collection framework and this *Java collection interview question* focus on that. Most important difference between them is that List allows duplicates and maintain insertion order while Set doesn't allow duplicates and doesn't maintain any order. See [Difference between Set and List in Java](http://javarevisited.blogspot.sg/2012/04/difference-between-list-and-set-in-java.html) to see more differences between them

**How do you find if ArrayList contains duplicates or not ?**

Since List allows duplicates this becomes a followup question of earlier Java collection framework interview question. See [How to check if ArrayList contains duplicates or not](http://javarevisited.blogspot.sg/2012/02/how-to-check-or-detect-duplicate.html) for answer of this Java collection question.

These were some of the **frequently asked Java collection framework interview question** you can also call them Java collection FAQ. Collection and Threads are most important part of Java programming language and considered as fundamentals of Java, So always prepare them well before appearing for any Java or J2EE interview. If you have any interesting Java collection interview question or are you looking answer for any Java collection question then please post here.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

http://java67.blogspot.sg/2012/09/10-linux-and-unix-interview-questions-answers-wipro-tcs-capegemini.html

10 Linux and UNIX Interview Questions and Answers asked in Wipro TCS Capegemini

**UNIX and Linux Interview Questions and Answers**

Questions from various UNIX operating systems e.g. Solaris, Linux, IBM AIX or any other UNIX operating system is asked on different support and [programming interviews](http://javarevisited.blogspot.sg/2011/06/top-programming-interview-questions.html). I have always seen few interview questions from Linux and UNIX along with SQL in almost every Java programming interviews. You just can not afford not to prepare questions from UNIX and Linux until your Job absolutely doesn't require any work in UNIX operating system. I have collected many [UNIX command interview questions](http://javarevisited.blogspot.sg/2011/05/unix-command-interview-questions.html) and already shared them but I found that except system admin jobs, many programming job interviews only focus on general UNIX commands e.g. [chmod](http://javarevisited.blogspot.sg/2012/03/10-example-of-chmod-command-in-unix.html), [find](http://javarevisited.blogspot.sg/2011/03/10-find-command-in-unix-examples-basic.html) or [grep](http://javarevisited.blogspot.sg/2011/06/10-examples-of-grep-command-in-unix-and.html) and fundamentals like finding files and directories, managing file space, [networking commands](http://javarevisited.blogspot.sg/2010/10/basic-networking-commands-in-linuxunix.html), checking process status and managing file permissions. In this article we will see such kind of frequently asked interview questions and answers from UNIX and Linux operating System. Questions are very fundamental in nature and not limited to Linux only and equally applicable to other UNIX operating systems e.g. Solaris, IBM AIX etc. Many of these UNIX questions are asked during various interviews on companies like TCS, Infosys, Citibank , Wipro, Capegemini and Tech Mahindra. But as I said they are very fundamental and can be asked in any company.

## UNIX and Linux Interview questions answers

Here is my list of frequently asked *UNIX and Linux interview questions and answers*. All these questions are based upon fundamental commands and concepts which is must for working in any UNIX operating system e.g. Solaris.

**1) How to find all the links in a folder in UNIX or Linux ?**

This is a *tricky UNIX question* as there is no specific command to find all symbolic links. Though you have [ln command for creating and updating soft links](http://javarevisited.blogspot.sg/2011/04/symbolic-link-or-symlink-in-unix-linux.html) but nothing which gives you all the links in a directory. You need to use ls command which list everything in directory and then you need to list all the links, as they starts with "l" as first characters, as shown in above article .

here is the actual UNIX command to find all links in a directory :

linux@nyj872:~ ls -lrt

total 2.0K

-rw-r--r-- 1 Linux Domain Users 0 Dec 6 2011 a

drwxr-xr-x+ 1 Linux Domain Users 0 Sep 19 12:30 java/

lrwxrwxrwx 1 Linux Domain Users 4 Sep 19 12:31 version\_1.0 -> java/

linux@nyj872:~ ls -lrt | grep '^l'

lrwxrwxrwx 1 Linux Domain Users 4 Sep 19 12:31 version\_1.0 -> java/

**2) How to find a process and kill that ?**

Another classic UNIX interview questions. Answer of this question is simple if you are familiar with ps, grep and kill command. by using "ps -ef" you can get list of all process and then use grep to find your process and get the PID of that process. Once you got PID you can use kill command to kill that process as shown in this [example of kill command in UNIX](http://javarevisited.blogspot.sg/2011/12/kill-command-unix-linux-example.html).

**3) How to run a program in background in UNIX or Linux ?**

an easy UNIX or Linux interview question, only when you know. You can use &amp; to run any process in background and than you can use jobs to find the job id for that process and can use fg and bg command to bring that process into foreground and background.

**4) How to sort output of a command in reverse order in Linux or UNIX ?**

One more Linux command interview question which checks knowledge of frequently used command. you can use sort command in UNIX to sort output of any command by using PIPE. By using -r option with sort command you can sort output of any command in reverse order. See these [sort command examples](http://javarevisited.blogspot.sg/2011/08/unix-sort-command-example-tutorial.html) for more details.

**5) How to create archive file in UNIX or Linux Operating System ?**

Another interview question based on knowledge of UNIX or Linux command. you can use [tar command](http://www.blogger.com/javarevisited.blogspot.sg/2011/11/tar-command-in-unix-linux-example.html) to great archives in UNIX or Linux. you can even combine tar and gzip to create a compressed archive in UNIX.

**6) What is meaning of a file has 644 permission ?**

To answer this UNIX or Linux interview question, you must know basics of files and directories in UNIX. 644 represents permission 110 for owner, permission 100 for group and 100 for others which means read + write for owner who create that file and read only permission for group and others. See this [tutorial on UNIX file permission](http://javarevisited.blogspot.sg/2011/11/file-permissions-in-unix-linux-example.html) for more details.

**7) How will you remove empty files or directories from /tmp ?**

See [how to delete empty directory and files in UNIX](http://javarevisited.blogspot.sg/2012/08/delete-empty-files-directories-unix.html) to answer this UNIX command interview questions.

**8) I have read permission on a directory but I am not able to enter it why ?**

One more tricky UNIX questions. In order to get into a directory you need execute permission. if your directory does not have execute permission than you can not go into that directory by using cd command. read [UNIX files and directory permissions](http://javarevisited.blogspot.sg/2011/11/file-permissions-in-unix-linux-example.html) for more information.

**9) How do you find all files which are modified 10 minutes before ?**

This is another the Linux interview questions from frequently used command e.g. find and grep. you can use -mtime option of find command to list all the files which are modified 10 or m minutes before. see these [find command examples](http://javarevisited.blogspot.sg/2011/03/10-find-command-in-unix-examples-basic.html) for more details.

**10) How to do you find size of directory in UNIX or Linux ?**

This is another tricky and bit tough Linux interview question as popular ls command doesn't show complete size of directories in UNIX. you need to use du command to get full size of directories including all sub directories in UNIX. See [How to find directory size in UNIX](http://javarevisited.blogspot.sg/2012/08/delete-empty-files-directories-unix.html) for exact command and detailed explanation.

These were some of the *frequently asked UNIX and Linux command interview questions* and answers which appear in many IT Job interview which requires knowledge of UNIX operating system, Including programming job interviews e.g. [core Java and J2EE interviews](http://javarevisited.blogspot.sg/2011/04/top-20-core-java-interview-questions.html).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<http://javarevisited.blogspot.sg/2013/09/how-clone-method-works-in-java.html>

How clone method works in Java?

clone is a tricky method from java.lang.Object class, which is used to create copy of an Object in Java. Intention of clone() method is simple, to provide a cloning mechanism, but some how it's implementation became tricky and has been widely criticized from long time. Anyway, we will not go to classic debate of clone in Java, at-least for now; instead, we will try to learn *how clone method works in Java*. To be fair, understating cloning mechanism in Java is not easy and even experienced Java programmer fail to explain how cloning of mutable object works, or [difference between deep copy and shallow copy in Java](http://java67.blogspot.sg/2013/05/difference-between-deep-copy-vs-shallow-cloning-java.html). In this three part article, we will first see working of clone method in Java, and in second part we will learn **how to override clone method in Java**, and finally we will discuss *deep copy vs shallow copy* mechanism. The reason I chose to make this a three part article, is to keep focus on one thing at a time. Since clone() itself is confusing enough, it's best to understand concept one by one. In this post, we will learn what is clone method, what it does and How clone method works in Java. By the way, clone() is one of the few fundamental methods defined by objects, others being equals, [hashcode()](http://javarevisited.blogspot.sg/2011/10/override-hashcode-in-java-example.html), [toString()](http://javarevisited.blogspot.com/2012/09/override-tostring-method-java-tips-example-code.html) along with [wait and notify methods](http://javarevisited.blogspot.com/2012/02/why-wait-notify-and-notifyall-is.html).

## What is clone of object in Java?

An object which is returned by clone() method is known as clone of original instance. A clone object should follow basic characteristics e.g. a.clone() != a, which means original and clone are two separate object in Java heap, a.clone().getClass() == a.getClass() and clone.equals(a), which means clone is exact copy of original object. These characteristic is followed by a well behaved, correctly overridden clone() method in Java, but it's not enforced by cloning mechanism. Which means, an object returned by clone() method may violate any of these rules. By following convention of returning object by calling super.clone(), when overriding clone() method, you can ensure that it follows first two characteristics. In order to follow third characteristic, you must [override equals method](http://javarevisited.blogspot.com/2011/02/how-to-write-equals-method-in-java.html) to enforce logical comparison, instead of physical comparison exists in java.lang.Object. For example, clone() method of Rectangle class in this method return object, which has these characteristics, but if you run same program by commenting equals(), you will see that third invariant i.e. clone.equals(a) will return false. By the way there are couple of good items on [Effective Java](http://www.amazon.com/dp/0321356683/?tag=javamysqlanta-20) regarding effective use of clone method, I highly recommend to read those items after going through this article.

## How Clone method works in Java

java.lang.Object provides default implementation of clone() method in Java. It's declared as [protected](http://javarevisited.blogspot.sg/2012/10/difference-between-private-protected-public-package-access-java.html) and native in Object class, so implemented in native code. Since it's convention to return clone() of object by calling super.clone() method, any cloning process eventually reaches to java.lang.Object clone() method. This method, first checks if corresponding object implements Cloneable interface, which is a marker interface. If that instance doesn't implements Cloneable then it throws CloneNotSupported in Java, a checked exception, which is always required to be handled while cloning an object. If object pass this check, than java.lang.Object's clone() method creates a shallow copy of object and returned it to the caller. Since Object class' clone() method creates copy by creating new instance, and then copying field-by-field, similar to assignment operator, it's fine for primitives and Immutable object, but not suited if your class contains some mutable data-structure e.g. Collection classes like ArrayList or [arrays](http://javarevisited.blogspot.sg/2011/06/converting-array-to-arraylist-in-java.html). In that case, both original object and copy of object will point to the same object in heap. You can prevent this by using technique known as deep cloning, on which each mutable field is cloned separately. In short, here is how clone method works in Java :

1) Any class calls clone() method on instance, which implements Cloneable and overrides protected clone() method from Object class, to create a copy.

Rectangle rec = **new** Rectangle(**30**, **60**);

logger.info(rec);

**try** {

logger.info("Creating Copy of this object using Clone method");

Rectangle copy = rec.clone();

logger.info("Copy " + copy);

} **catch** (CloneNotSupportedException ex) {

logger.debug("Cloning is not supported for this object");

}

2) Call to clone() method on Rectangle is delegated to super.clone(), which can be a custom [super class](http://java67.blogspot.sg/2013/06/difference-between-this-and-super-keyword-java.html) or by default java.lang.Object

@Override

**protected** Rectangle **clone**() **throws** CloneNotSupportedException {

**return** (Rectangle) **super**.clone();

}

3) Eventually call reaches to java.lang.Object's clone() method, which verify if corresponding instance implements Cloneable interface, if not then it throws CloneNotSupportedException, otherwise it creates a field-by-field copy of instance of that class and returned to caller.

So in order for clone() method to work properly, two things need to happen, a Class should implement Cloneable interface and should override clone() method of Object class. By the way this was this was the simplest example of overriding clone method and how it works, things gets more complicated with real object, which contains mutable fields, arrays, collections, [Immutable object](http://javarevisited.blogspot.com/2013/03/how-to-create-immutable-class-object-java-example-tutorial.html) and primitives, which we will see in second part of this **Java Cloning tutorial** series.

## Java clone() method Example

In this article, we have not seen complexity of overriding clone method in Java, as our Rectangle class is very simple and only contains primitive fields, which means shallow cloning provided by Object's clone() method is enough. But, this example is important to understand process of Object cloning in Java, and How clone method works. Here is complete code of this clone() method overriding example :

**import** **org.apache.log4j.Logger**;

/\*\*

\* Simple example of overriding clone() method in Java to understand How Cloning of

\* Object works in Java.

\*

\* @author

\*/

**public** **class** **JavaCloneTest** {

**private** **static** **final** Logger logger = Logger.getLogger(JavaCloneTest.class);

**public** **static** **void** **main**(String args[]) {

Rectangle rec = **new** Rectangle(**30**, **60**);

logger.info(rec);

Rectangle copy = **null**;

**try** {

logger.info("Creating Copy of this object using Clone method");

copy = rec.clone();

logger.info("Copy " + copy);

} **catch** (CloneNotSupportedException ex) {

logger.debug("Cloning is not supported for this object");

}

//testing properties of object returned by clone method in Java

logger.info("copy != rec : " + (copy != rec));

logger.info("copy.getClass() == rec.getClass() : " + (copy.getClass() == rec.getClass()));

logger.info("copy.equals(rec) : " + copy.equals(rec));

//Updating fields in original object

rec.setHeight(**100**);

rec.setWidth(**45**);

logger.info("Original object :" + rec);

logger.info("Clonned object :" + copy);

}

}

**public** **class** **Rectangle** **implements** Cloneable{

**private** **int** width;

**private** **int** height;

**public** **Rectangle**(**int** w, **int** h){

width = w;

height = h;

}

**public** **void** **setHeight**(**int** height) {

**this**.height = height;

}

**public** **void** **setWidth**(**int** width) {

**this**.width = width;

}

**public** **int** **area**(){

**return** widthheight;

}

@Override

**public** String **toString**(){

**return** String.format("Rectangle [width: %d, height: %d, area: %d]", width, height, area());

}

@Override

**protected** Rectangle **clone**() **throws** CloneNotSupportedException {

**return** (Rectangle) **super**.clone();

}

@Override

**public** **boolean** **equals**(Object obj) {

**if** (obj == **null**) {

**return** **false**;

}

**if** (getClass() != obj.getClass()) {

**return** **false**;

}

**final** Rectangle other = (Rectangle) obj;

**if** (**this**.width != other.width) {

**return** **false**;

}

**if** (**this**.height != other.height) {

**return** **false**;

}

**return** **true**;

}

@Override

**public** **int** **hashCode**() {

**int** hash = **7**;

hash = **47** hash + **this**.width;

hash = **47** hash + **this**.height;

**return** hash;

}

}

*Output:*

**2013**-**05**-**20** **23**:**46**:**58**,**882** **0** [main] INFO JavaCloneTest - Rectangle [*width:* **30**, *height:* **60**, *area:* **1800**]

**2013**-**05**-**20** **23**:**46**:**58**,**882** **0** [main] INFO JavaCloneTest - Creating Copy of **this** object using Clone method

**2013**-**05**-**20** **23**:**46**:**58**,**882** **0** [main] INFO JavaCloneTest - Copy Rectangle [*width:* **30**, *height:* **60**, *area:* **1800**]

**2013**-**05**-**20** **23**:**46**:**58**,**882** **0** [main] INFO JavaCloneTest - copy != rec : **true**

**2013**-**05**-**20** **23**:**46**:**58**,**882** **0** [main] INFO JavaCloneTest - copy.getClass() == rec.getClass() : **true**

**2013**-**05**-**20** **23**:**46**:**58**,**882** **0** [main] INFO JavaCloneTest - copy.equals(rec) : **true**

**2013**-**05**-**20** **23**:**46**:**58**,**882** **0** [main] INFO JavaCloneTest - Original object :Rectangle [*width:* **45**, *height:* **100**, *area:* **4500**]

**2013**-**05**-**20** **23**:**46**:**58**,**882** **0** [main] INFO JavaCloneTest - Cloned object :Rectangle [*width:* **30**, *height:* **60**, *area:* **1800**]

From output, you can clearly see that cloned object has same attribute as original object in Java. Also changing attribute of original object is not affecting state of copy object, because they only contains primitive fields, had then contain any mutable object, it would have affected both of them. You can also see that it follow standard properties of cloned object i.e. clone != original, clone.getClass() == original.getClass() and clone.equals(original).

### Things to Remember - Clone method in Java

1) Clone method is used to create a copy of object in Java. In order to use clone() method, class must implement java.lang.Cloneable [interface](http://javarevisited.blogspot.com/2012/04/10-points-on-interface-in-java-with.html) and override protected clone() method from java.lang.Object. A call to clone() method will result in CloneNotSupportedException, if that class doesn't implement Cloneable interface.

2) No constructor is called during cloning of Object in Java.

3) Default implementation of clone() method in Java provides "*shallow copy"* of object, because it creates copy of Object by creating new instance and then copying content by assignment, which means if your Class contains a mutable field, then both original object and clone will refer to same internal object. This can be dangerous, because any change made on that mutable field will reflect in both original and copy object. In order to avoid this, override clone() method to provide deep copy of object.

4) By convention, clone of an instance should be obtained by calling super.clone() method, this will help to preserve invariant of object created by clone() method i.e. clone != original and clone.getClass() == original.getClass(). Though these are not absolute requirement as mentioned in Javadoc.

5) Shallow copy of an instance is find, until it only contains primitives and Immutable objects, otherwise, you need to modify one or more mutable fields of object returned by super.clone, before returning it to caller.

That's all on **How clone method works in Java**. Now we know, what is clone and what is Cloneable interface, couple of things about clone method and what does default implementation of clone method do in Java. This information is enough to move ahead and read second part of this Java cloning tutorial, on which we will learn, *how to override clone() method in Java*, for classes composed with primitives, Mutable and Immutable objects in Java.

**Recommended Book**

Like most of important topics in Java, Joshua Bloch has shared some words of wisdom on object cloning and clone method in Java. I highly suggest going through those items on his evergreen [Effective Java](http://www.amazon.com/dp/0321356683/?tag=javamysqlanta-20) book.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |
| --- |
| Invoking clone() with Generic Types in Java |
| (Want to skip ahead to the code? [Click here.](http://community.wvu.edu/%7Enwhiteha/projects/misc/cloning_generics_in_java/#the_code))  If you have done any work with generic types in Java, you have probably encountered an unexpected problem: Java refuses to let you clone generic objects, even if they implement the Cloneable interface. Consider the following simple class with a copy constructor:  public class GenericContainerExample<E extends Cloneable>{    private E data;    GenericContainerExample(){  this.data=null;  }    GenericContainerExample<E>(E data){  this.data=data;  }    GenericContainerExample<E>(GenericContainerExample<E> c){  this.data=c.data.clone();  }    public void setData(E data){  this.data=data;  }    public E getData(){  return this.data;  }    public String toString(){  return this.data.toString();  }  }  This seems reasonable enough. We have the foresight to try to create a deep copy of the internal data when creating a copy of an instance of the class, lest multiple instances of the class should contain references to the same data. (It should be noted, however, that the Java specification is somewhat murky regarding the behaviour of clone(). Typically, however, the developer of a class will use clone to return a deep copy of the object, which is our assumption for this example.) We have forced the type to be Cloneable for this reason. If you try to compile this code, however, you will get an error similar to the following:  Error: cannot find symbol     Symbol: method clone()     location java.lang.Cloneable  Java is telling us that "Cloneable" objects don't necessarily have the method clone() available. This might seem strange, but the Java specification is such that the Cloneable interface only requires that it be legal for the object to be cloned if it does have a clone() method. (See this page: <http://docs.oracle.com/javase/6/docs/api/java/lang/Cloneable.html>)  The only "easy" way to invoke clone() on an object of a generic type in Java is by first casting it to a specific type which does implement clone() at compile time. But that completely defeats the purpose of using generic types, so it's not of much use. (If you want to experiment with this, try casting c.data to something like java.util.ArrayList and you'll see that the compiler has no problem letting you invoke clone().)  Another potential solution might be to cast the data to type Object, which does have a clone() method, and then invoke clone(). Unfortunately, even though this is a bad idea for other reasons, it gives us a different error:  Error: clone() has protected access in java.lang.Object  No matter how you try to cast the data, there doesn't seem to be any way to reliably clone it... unless you turn to reflection.  Reflection is a powerful tool which allows us to peer into and manipulate objects at runtime in "non-traditional" ways. (Reflection is actually a fairly old concept, so it is only "non-traditional" in the sense that it is a more-advanced and somewhat more-obscure technique.) There are many ways to use reflection, and not all of them are good ideas, but we are only interested in one: invoking methods.  Java allows us to enumerate a class's methods and then invoke any of those methods on an object. We can use this power to easily overcome the strange limitations of the Cloneable interface, but as you will see from the number of exceptions we will need to handle, there is a lot that can go wrong. (For more information about reflection in Java, including some of the caveats, see this page: <http://docs.oracle.com/javase/tutorial/reflect/>)  Here is a method which will take an instance of our cloneable generic class, E, and return a true clone of it.  import java.lang.reflect.\*;  // Returns a clone of d iff it implements Cloneable and possesses a "clone()" method.  // Returns d itself otherwise.  private E E\_clone(E d) throws IllegalAccessException, InvocationTargetException  {    for (Class i: d.getClass().getInterfaces())      if (i.getName().equals("java.lang.Cloneable")) // Be sure E implements Cloneable.      for (Method m : d.getClass().getMethods())        if (m.getName().equals("clone") && // Be sure E has an appropriate clone(),            m.getParameterTypes().length==0 && // takes no parameters, and returns            m.getReturnType().getName().equals("java.lang.Object") // type Object.              return (E)m.invoke(d); // Invoke d.clone(), typecast, and return it.    return d;  // If one of the above conditions was not met, then E is not cloneable,  }            // so return a shallow copy instead.  The code is fairly self-explanatory, even without comments, but in case it isn't clear enough, I will explain further.  First, we invoke getInterfaces() on E, which returns an array of type Class containing each of the interfaces implemented by E. We search this array to see if E implements the Cloneable interface. If so, then we invoke getMethods(), which returns an array of type Method containing (you guessed it) all of the methods provided by E. We search this array for a method called "clone" with no parameters and returning type Object. If such a method exists for E, we invoke that method for d (the data we want to clone). The result is then typecast to type E and returned.  If one of the conditions we set is not met, we instead return the original data, d. We could throw an exception instead, but here I make the assumption that E is an immutable type, so it is probably safe to use a shallow copy. That means that we no longer have to impose Cloneable on our generic type (unless we want to). Removing the exceptions, our class becomes:  public class GenericContainerExample<E extends Cloneable>{    private E data;    GenericContainerExample(){ this.data=null; }    GenericContainerExample(E data){ this.data=data; }    GenericContainerExample(GenericContainerExample<E> c)    { this.data=E\_clone(c.data); }    public void setData(E data){ this.data=data; }    public E getData(){ return this.data; }    public String toString(){ return this.data.toString(); }    private E E\_clone(E d){      try{        for (Class i: d.getClass().getInterfaces())          if (i.getName().equals("java.lang.Cloneable"))            for (java.lang.reflect.Method m : d.getClass().getMethods())              if (m.getName().equals("clone") &&                  m.getParameterTypes().length==0 &&                  m.getReturnType().getName().equals("java.lang.Object"))                return (E)m.invoke(d);}      catch(IllegalAccessException ex){} // This should never happen.      catch(java.lang.InvocationTargetException ex){} // This should never happen.      return d;    }  } |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<http://java67.blogspot.sg/2012/08/difference-between-countdownlatch-and-cyclicbarrier-java.html>

Difference between CountDownLatch and CyclicBarrier in Java

**Difference between CountDownLatch and CyclicBarrier in Java**

Both CyclicBarrier and CountDownLatch are used to implement a scenario where one Thread waits for one or more Thread to complete there job before starts processing but there is one Difference between CountDownLatch and CyclicBarrier in Java which separates them apart and that is, you can not reuse same CountDownLatch instance once count reaches to zero and latch is open, on the other hand CyclicBarrier can be reused by resetting Barrier, Once barrier is broken.

A useful property of a CountDownLatch is that it doesn't require that threads calling countDown wait for the count to reach zero before proceeding, it simply prevents any thread from proceeding past an await until all threads could pass.

A CyclicBarrier supports an optional Runnable command that is run once per barrier point, after the last thread in the party arrives, but before any threads are released. This *barrier action* is useful for updating shared-state before any of the parties continue.

The CyclicBarrier uses a fast-fail all-or-none breakage model for failed synchronization attempts: If a thread leaves a barrier point prematurely because of interruption, failure, or timeout, all other threads, even those that have not yet resumed from a previous await(), will also leave abnormally via BrokenBarrierException (or InterruptedException if they too were interrupted at about the same time).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<http://javarevisited.blogspot.com/2012/01/google-interview-questions-answers-top.html>

Top 10 Google Interview Questions for Software Engineer - Books, Resources

These **Google interview questions** are some of my favorites collected from different sources. Every Programmer know that Google is one of the best technology company and its dream for many software developer to work for google, but at same time interview process at google is very tough and only few genuine intelligent programmers get through there interview process. **Google interview questions** are always been a good topic of discussion when few young software developer gathered around, I can still remember when one of my friend got call from google for interview then how whole bunch was got excited. We have searched a lot on internet on *google interview questions* *and answers* for him and us and then make a note of some of the best questions for preparation. I am listing down some google interview questions from that list. Apart from the popular questions asked in various Google Interview for software engineers or developers, books on algorithm and data structure plays a lot more important roles. Books like [Algorithms for Interviews](http://www.amazon.com/dp/1453792996/?tag=javamysqlanta-20) and [Introduction to Algorithms](http://www.amazon.com/dp/0072970545/?tag=javamysqlanta-20) are must read for any Programmer, who is serious about converting Google interview or similar companies like Facebook, Amazon and Microsoft. Once you start preparing for Google, rest of them become very easy.

## Google Interview questions answers for Software Engineers

Answers of these **Google interview questions** can be found  by doing google :) , Also each question in itself demand a blog post and I will try to cover those one by one as and when time allows. I have divided all interview questions on different category like data structure, puzzles, operating system , Software Design and Coding etc for better organization.  In the last section, I have shared some top class, must read books for Google Interview preparation. It contains books like Are You Smart Enough to Work at Google?, which is not only a good read but tells you a lot about Google interview, it's process. Before starting preparation, a must read to get yourself motivated.

### Google Interview Questions on Data Structure and Algorithm

Here are few questions from data structures appeared on various google interviews, data-structure is a complex topic and can get very complex at google and we have purposefully not collected very tough data-structure questions at that time because of time limit and we wanted to gain some confidence to solve moderate questions, though this may not be a good strategy for many people but it depends how much knowledge you have and how confident you are on data-structure.

1. Find out the fastest way to locate the largest element in a circular sorted array ?
2. You have a binary search tree and integer n, find out the most efficient way to locate two nodes of the three whose summation is equals to "n" ?
3. How do you convert a max heap to min heap ?
4. You have two list L1 and L2 write an algorithm to see if both the lists are equivalent or not?
5. How do you figure out time and space complexity of recursive function ?
6. how will you implement three stacks with one array.

See [here](http://javarevisited.blogspot.com/2013/03/top-15-data-structures-algorithm-interview-questions-answers-java-programming.html) for few more questions on data structure and algorithms

### Google Interview Questions based on Networking and Operating Systems

Here are some of my favorite Google interview questions from various operating systems like Unix and Linux. Good knowledge of how operating system works always helps to excel any programming interview and Google is not an exception.

1. If you have given a chance to rewrite Linux, how will you do that ?
2. Explain how congestion control works in the TCP protocol ?
3. How would you optimize TCP Protocol without going on UDP ways ?

See here for some more [socket based networking questions](http://javarevisited.blogspot.sg/2014/08/socket-programming-networking-interview-questions-answers-Java.html) form Programming interviews

### Google Interview Questions on Software design

Here are some of software design questions from various Google interviews for software engineer position. Software design is critical piece to get though Google interview and people has almost always faced one of two questions from this category.

1. You have provided a set of points across globe as latitudes and longitudes ? How will you determine points which are within a particular mile ?
2. Can you design data package transfer form London to Tokyo ?
3. Design a distributed system for storing a static set of (key, value) pairs which will behave like a hash table, user will provide you key and system will return value.
4. Design Algorithm for Lift to get minimum waiting time on each floor ?

For more questions on Object oriented design and software design see [here](http://javarevisited.blogspot.com/2012/06/20-design-pattern-and-software-design.html)

### Google interview questions on Coding

There will always be some questions on coding in Google interview, especially if it is for software engineer or developer role. They want to see you writing code. There is no language barrier and you can choose whatever programming language you are most comfortable with.**Google interview questions on coding** mostly requires good knowledge of recursion, data structure and good familiarity of programming language.

1. Can you write code in Java or C++ to find the power set of a given set. For example if S={a,b} the power set is P={{},{a},{b},{a,b}} ( you can also choose any of your favorite programming language)
2. Write code in Java to find out whether a binary tree is a mirror image of itself or not. Code should be thread-safe?
3. Can you write code to implement your own hashtable in C++ or Java?
4. Write code to find out number of occurrence of a number in a sorted array ?
5. Can you implement a Generic LRU Cache in Java ?
6. Write a program to solve a Rubik's cube in any programming language.

For more coding questions, you can  also see my list of [Top 30 programming interview questions](http://javarevisited.blogspot.com/2011/06/top-programming-interview-questions.html).

### Miscellaneous Google Interview Questions

In this category of *google interview question* you will see mostly behavioral questions, some time before throwing data-structure questions they ask light question and some time after exhaustive screening they ask based on your profile and luck. Apart from below they also questions based on puzzles and riddles.

1. How Google Search work ? If asked to you how will you design it to scale and fast at same time ?
2. Why do you want to work with Google ?
3. Which Google product you like most and why ?
4. Do you like coding or designing application ?
5. Questions about your previous project and work experience.

Puzzles are another popular topic on various google interview and there are lot of puzzles available in internet as well make sure you prepare on puzzles before appearing to any google interview. Some time they even ask questions on lateral thinking and tricks which doesn't require complex logic but sharp mind to guess. See this list of the most [popular puzzles asked in programming interviews](http://javarevisited.blogspot.com/2013/04/top-10-puzzles-riddles-logical-programming-question-lateral-thinking-interview.html).

Books to Prepare Google Interview Questions

There are lot of books available in market, which covers lot of different topics asked in Google or Amazon interviews, but most of them lack quality. In order to succeed in Google, most important thing is data structure and algorithm topic along with problem solving skills, if you have that, half of battle is won. Following are some of the popular books, which programmers prefer during preparation of Google interviews:

1. [Are You Smart Enough to Work at Google?  By William Poundstone](http://www.amazon.com/dp/0316099988/?tag=javamysqlanta-20)

Google is known for asking insanely difficult puzzles, this books takes you on both. It's readable, fun and same time introduce hard realities of Google interviews. You will be glued to read this book, after reading the first interview experience :)

2. **Algorithms For Interviews By Adnan Aziz and Amit Prakash**

This book is especially written for Google interviews. Authors has put decent effort to not only show how to use algorithm but also focus on problem solving skill. A must read for any Google aspirants.

3. **Introduction to Algorithms (Includes CD-Rom) By Thomas Cormen,** Charles Leiserson, Ronald Rivest, Clifford Stein

Another top class book on topic of Algorithm. As I said, data structure and algorithms are extremely important to survive and excel google interviews. Read this book at least one time, if not two. Another must read for Google aspirants.

4. **Top 10 coding interview problems asked in Google with solutions: Algorithmic Approach**

One more great book to prepare for Google interviews. This book present 10 top quality code questions asked on Google, with full solution including pseudo code and C++. This book fills lot of gap on how to solve coding questions on Google, something freshers and software engineers lacks.

That's all on **Google interview questions for Software Engineers and Programmers** .What is most important is that most of the questions asked in google is original and you don't see them again. practice can give you guidance on how to tackle those questions.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

http://java67.blogspot.sg/2013/05/difference-between-deep-copy-vs-shallow-cloning-java.html

Difference between Deep and Shallow Copy in Java Object Cloning

Shallow copy and deep copy is related with cloning process so before go into the deep of shallow and deep copy we need to understand what is clone in java. Clone is nothing but the process of copying one object to produce the exact object, which is not guaranteed. We all know in Java object is referred by reference we can not copy one object directly to another object. So we have cloning process to achieve this objective. Now one question arise in mind why we need this process so the answer is whenever we need local copy of the object to modify the object in some method but not in method caller.  So we can define Cloning as “**create a copy of object** “ .I think now we are some how clear about the cloning but there is more on it depending upon how we are doing this copy, we can divide cloning in two types.

* Shallow Copy
* Deep Copy

Before going into the deep of shallow and deep copy we need to understand how we achieve cloning in java.

### How to Clone in java?

In Java everything is achieved through class, object and interface .By default no Java class support cloning but Java provide one interface called Cloneable, which is a [marker interface](http://javarevisited.blogspot.com/2012/01/what-is-marker-interfaces-in-java-and.html) andbyimplementingthis interface we can make duplicate copy of our object by calling clone() method of java.lang.Object class. This Method is protected inside the object class and Cloneable interface is a marker interface and this method also throw  **CloneNotSupportedException** if we have not implement thisinterface and try to call clone() method of Object class. By default any clone() method gives **shallow copy** of the object i.e. if we invoke super. clone() then it’s a shallow copy but if we want to **deep copy** we have to override the clone() method and make it public and give own definition of making copy of object. Now we let’s see what is shallow and deep copy of object in Java programming language.

### Shallow Copy

Whenever we use default implementation of clone method we get shallow copy of object means it create new instance and copy all the field of object to that new instance and return it as **object type** we need to explicitly cast it back to our original object. This is shallow copy of the object. clone() method of the object class support shallow copy of the object. If the object contains primitive as well as non primitive or reference type variable In shallow copy, the cloned object also refers to the same object to which the original object refers as only the object references gets copied and not the referred objects themselves. That's why the name shallow copy or shallow cloning in Java. If only primitive type fields or [Immutable objects](http://javarevisited.blogspot.com/2013/03/how-to-create-immutable-class-object-java-example-tutorial.html) are there then there is no difference between shallow and deep copy in Java.

### Deep Copy

Whenever we need own meaning of copy not to use default implementation we call it as deep copy, whenever we need deep copy of the object we need to implement according to our need. So for deep copy we need to ensure all the member class also implement the Cloneable interface and override the clone() method of the object class. After that we override the clone() method in all those classes even in the classes where we have only primitive type members otherwise we would not be able to call the protected clone() method of Object class on the instances of those classes inside some other class. It’s typical restriction of the protected access.

### Difference between Shallow and Deep Copy in Java

I think now we know what is deep and shallow copy of object in Java, let see some difference between them so that we can get some more clarity on them.

* When we call Object.clone(), this method performs a shallow copy of object, by copying data field by field, and if we override this method and by convention first call super.clone(), and then modify some fields to "deep" copy, then we get deep copy of object. This modification is done to ensure that original and cloned object are independent to each other.
* In shallow copy main or parent object is copied, but they share same fields or children if fields are modified in one parent object other parent fields have automatic same changes occur,but in deep copy this is not the case.
* If our parent object contains only primitive value then shallow copy is good for making clone of any object because in new object value is copied but if parent object contains any other object then only reference value is copied in new parent object and both will point to same object so in that case according to our need we can go for deep copy.
* Deep copy is expensive as compare to shallow copy in terms of object creation, because it involves recursive copying of data from other mutable objects, which is part of original object.

This is all about deep copy and shallow copy of objects in Java. Now the question comes when we use shallow copy and when go for deep copy , so answer would be simple that if the object has only primitive fields or Immutable objects, then obviously we will go for shallow copy, but if the object has references to other mutable objects, then based on the requirement, shallow copy or deep copy can be chosen. Means if the references are not modified anytime, then there is no point in going for deep copy, We can go for shallow copy. But if the references are modified often, then you need to go for deep copy. Again there is no hard and fast rule, it all depends on the requirement.

Hope this article will help to make clear about deep and shallow copy of cloning process.

Related **Java Interview Questions articles** from Java67 Blog

[Difference between wait and sleep in Java](http://java67.blogspot.com/2012/08/what-are-difference-between-wait-and.html)

[Difference between String and StringBuffer in Java](http://java67.blogspot.com/2012/08/difference-between-string-and-stringbuffer-in-java.html)

[Difference between Runnable and Thread in Java](http://java67.blogspot.com/2012/08/what-is-thread-and-runnable-in-java.html)

[Difference between TreeSet and TreeMap in Java](http://java67.blogspot.com/2012/08/difference-between-treemap-and-treeset-java.html)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<http://java67.blogspot.sg/2013/06/difference-between-this-and-super-keyword-java.html>

### Difference between this and super keywords in Java

this and super are two special keywords in Java, which is used to represent current instance of a class and it's super class. Java Programmers often confused between them and not very familiar with there special properties, which is asked at various [core Java interviews](http://java67.blogspot.com/2012/09/top-10-tough-core-java-interview-questions-answers.html). Couple of questions, which I remember about this and super keyword is that, **Can we reassign this in Java?** and ***difference between this and super keyword in Java***. Do you want to try that? Ok, I am not giving answer now, rather I will let you know answer at the end of this post. As I said in first line, main difference between this and super in Java is that, this represent current instance of a class, while super represent current instance of parent class. Now where does this and super variables used, well you might have seen examples of calling one constructor from other i.e. [constructor chaining](http://java67.blogspot.com/2012/12/how-constructor-chaining-works-in-java.html), that's achieved by using this and super keyword. You can use this() to call no argument constructor of same class, while super() to call no argument or default constructor of parent class. By the way, call is not limited to only no argument constructor, you can call any constructor by passing appropriate parameters. We will see example of using this and super in a while. Another use of this and super in Java is for *accessing instance variables of a class and it's parent*. By the way, you can also access them even without prefixing super and this, if they are not ambiguous in current block e.g. if there is no [local variable](http://javarevisited.blogspot.com/2012/02/difference-between-instance-class-and.html) of same name exists in that block, but in case of ambiguity they provide explicit reference and are more readable. Classic example of using this is inside a [constructor](http://javarevisited.blogspot.com/2012/12/what-is-constructor-in-java-example-chainning-overloading.html), which accepts a parameter of same name as an instance variable. In this post, we will learn about more differences between super and this in Java and take a look at some of the use cases.

## Similarities between this and super

Before seeing *difference between this and super keywords in Java*, let's see some similarities between them :

1) Both this and super are non static and can not be used in static context, which means you can not use this and super keyword inside [main method in Java](http://java67.blogspot.com/2012/08/what-is-main-method-in-java-why-main-is.html). Failing to do so will result in compiler error "non static variable this can not be referenced from static context". Same is true for using super keyword inside main method.

public static void main(String args[]) {

// compiler error - non static variable can not be referenced from static context

System.out.println(this.name);

}

2) Both this and super can be used in constructor chaining to call another constructor e.g. this() and super() respectively calls **no argument constructor** of child and parent class.

As shown in this example, we are first forwarding call from no argument constructor of B, to a constructor which accepts one String argument, which further call to super(""), a call to super class, one argument constructor.

class *A*{

*A*(){

System.out.**println**("A'*s* *no* *argument* *constructor*");

}

A(String args){

System.out.**println**("A'*s* *one* *argument* *constructor*");

}

}

class *B* extends *A*{

*B*(){

**this**(""); // calling one arg constructor of class B

System.out.**println**("B'*s* *no* *argument* *constructor*");

}

B(String args){

**super**(""); // calling one argument constructor of class A

System.out.**println**("B'*s* *one* *argument* *constructor*");

}

}

// Test Class and Output

public class *Test* {

public static void **main**(String *args*[]) {

B b **=** **new** B();

}

}

Output**:**

A's one argument constructor

B's one argument constructor

B's no argument constructor

3) If used them inside constructor than this and super must be first statement, otherwise compiler will complain. Which means you can not call this() and super() from same constructor.

### Super vs This in Java

Now we know how to use super and this keyword in Java, and comfortable with there intended use. One use of this keyword, which I didn't shown here is that you can also use them inside [Inner classes](http://javarevisited.blogspot.sg/2012/12/inner-class-and-nested-static-class-in-java-difference.html), they are quite handy to get reference of outer class in Java, Outer.this and Outer.super can be used to get current instance of Outer class and it's parent in Java. Don't forget to replace Outer, with the name of enclosing class. Now In short, here are *main differences between this and super keyword in Java*

1) this is used in context of the class you are working on, while super is used to refer current instance of parent class.

2) Every constructor by default calls super(), which is a call to no argument constructor of parent class, but you can still call another constructor either explicitly by using this() or super().

That's all on **difference between this and super keyword in Java** and How to use them in a Java program. We have seen calling another constructor from this and super keyword, and access member variables from current and parent class using them. Remember they are very special variables, and now answer of my question, which I asked in first paragraph. No, you can not assign new value to this variable, because it's final. You can try doing that in an IDE, you will get compiler error *"can not assign a new value to final variable this"*.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*