Top 18 Java Design Pattern Interview Questions Answers for Experienced

**Design pattern interview question in Java**  
design pattern interview question are integral part of any good list of core Java interview questions. Java is a popular Object oriented programming language and have lots of design pattern and design principles, contributed by many developers and open source framework. As a Java programmer its expected from you to know OOPS concept like [Abstraction](http://javarevisited.blogspot.sg/2010/10/abstraction-in-java.html), [Encapsulation](http://javarevisited.blogspot.sg/2012/03/what-is-encapsulation-in-java-and-oops.html)and [polymorphism](http://javarevisited.blogspot.sg/2011/08/what-is-polymorphism-in-java-example.html), What is design pattern in Java, Some popular Java design pattern and most importantly when to use those design pattern in Java application. purpose of asking design pattern interview question in Java is to check whether Java programmer are familiar to those essential design patterns or not. Design patterns in Java interviews are as important as[multi-threading](http://javarevisited.blogspot.sg/2011/07/java-multi-threading-interview.html), [collection](http://javarevisited.blogspot.sg/2011/11/collection-interview-questions-answers.html)and [programming questions](http://javarevisited.blogspot.sg/2011/06/top-programming-interview-questions.html). If you are senior or experienced Java programmer than expect more complex and tough design pattern in Java interview e.g. Chain of responsibility design pattern and solving real time [software design questions](http://javarevisited.blogspot.sg/2012/06/20-design-pattern-and-software-design.html).

Top Java design pattern questions and answers

Here is my list of t*op 10 design pattern interview question in Java*. I have also provided answer of those Java design pattern question as link. no matter which level of Java interview are you going e.g. programmer, software engineer, senior software engineer in Java, you can expect few question from Java design pattern.

**1. When to use Strategy Design Pattern in Java?**  
[Java design pattern interview question and answers for senior and experience programmer](http://3.bp.blogspot.com/-1lzFJzIgaHk/UF2Ci6kY5pI/AAAAAAAAAes/OYiM7r-DHzc/s1600/17.jpg)Strategy pattern in quite useful for implementing set of related algorithms e.g. compression algorithms, filtering strategies etc. Strategy design pattern allows you to create Context classes, which uses Strategy implementation classes for applying business rules. This pattern follow open closed design principle and quite useful in Java. One example of Strategy pattern from JDK itself is a Collections.sort() method and[Comparator interface](http://java67.blogspot.sg/2012/10/how-to-sort-object-in-java-comparator-comparable-example.html), which is a strategy interface and defines strategy for comparing objects. Because of this pattern, we don't need to modify sort() method (closed for modification) to compare any object, at same time we can implement Comparator interface to define new comparing strategy (open for extension).  
  
**2. What is Observer design pattern in Java? When do you use Observer pattern in Java?**  
This is one of the most common Java design pattern interview question. Observer pattern is based upon notification, there are two kinds of object Subject and Observer. Whenever there is change on subject's state observer will receive notification. See [What is Observer design pattern in Java with real life example](http://javarevisited.blogspot.sg/2011/12/observer-design-pattern-java-example.html) for more details.  
  
**3. Difference between Strategy and State design Pattern in Java?**  
This is an interesting Java design pattern interview questions as both Strategy and State pattern has same structure. If you look at UML class diagram for both pattern they look exactly same, but there intent is totally different. State design pattern is used to define and mange state of object, while Strategy pattern is used to define a set of interchangeable algorithm and let's client to choose one of them. So Strategy pattern is a client driven patternwhile Object can manage there state itself.  
  
**4. What is decorator pattern in Java? Can you give an example of Decorator pattern?**  
Decorator pattern is another popular java design pattern question which is common because of its heavy usage in java.io package. BufferedReader and BufferedWriter are good example of decorator pattern in Java. See [How to use Decorator pattern in Java](http://javarevisited.blogspot.com/2011/11/decorator-design-pattern-java-example.html) fore more details.  
  
**5. When to use Composite design Pattern in Java? Have you used previously in your project?**  
This design pattern question is asked on Java interview not just to check familiarity with Composite pattern but also, whether candidate has real life experience or not. *Composite pattern* is also a core Java design pattern, which allows you to treat both whole and part object to treat in similar way. Client code, which deals with Composite or individual object doesn't differentiate on them, it is possible because Composite class also implement same interface as there individual part. One of the good example of Composite pattern from JDK is JPanel class, which is both Component and Container.  When paint() method is called on JPanel, it internally called paint() method of individual components and let them draw themselves. On second part of this design pattern interview question, be truthful, if you have used then say yes, otherwise say that you are familiar with concept and used it by your own. By the way always remember, giving an example from your project creates better impression.  
  
**6. What is Singleton pattern in Java?**  
Singleton pattern in Java is a pattern which allows only one instance of Singleton class available in whole application. java.lang.Runtime is good example of Singleton pattern in Java. There are lot's of follow up questions on Singleton pattern see [10 Java singleton interview question answers](http://javarevisited.blogspot.com/2011/03/10-interview-questions-on-singleton.html) for those followups   
  
**7. Can you write thread-safe Singleton in Java?**  
There are multiple ways to write thread-safe singleton in Java e.g by writing singleton using double checked locking, by using static Singleton instance initialized during [class loading.](http://javarevisited.blogspot.sg/2012/07/when-class-loading-initialization-java-example.html) By the way using Java enum to create thread-safe singleton is most simple way. See [Why Enum singleton is better in Java](http://javarevisited.blogspot.gr/2012/07/why-enum-singleton-are-better-in-java.html) for more details.  
  
**8. When to use Template method design Pattern in Java?**Template pattern is another popular core Java design pattern interview question. I have seen it appear many times in real life project itself. Template pattern outlines an algorithm in form of template method and let subclass implement individual steps. Key point to mention, while answering this question is that template method should be final, so that subclass can not override and change steps of algorithm, but same time individual step should be abstract, so that child classes can implement them.  
  
**9. What is Factory pattern in Java? What is advantage of using static factory method to create object?**  
Factory pattern in Java is a creation Java design pattern and favorite on many Java interviews.Factory pattern used to create object by providing static factory methods. There are many advantage of providing factory methods e.g. caching immutable objects, easy to introduce new objects etc. See [What is Factory pattern in Java and benefits](http://javarevisited.blogspot.sg/2011/12/factory-design-pattern-java-example.html) for more details.  
  
**10. Difference between Decorator and Proxy pattern in Java?**Another tricky Java design pattern question and trick here is that both Decorator and Proxy implements interface of the object they decorate or encapsulate. As I said, many Java design pattern can have similar or exactly same structure but they differ in there intent. Decorator pattern is used to implement functionality on already created object, while Proxy pattern is used for controlling access to object. One more difference between Decorator and Proxy design pattern is that, Decorator doesn't create object, instead it get object in it's constructor, while Proxy actually creates objects.  
  
**11. When to use Setter and Constructor Injection in Dependency Injection pattern?**  
Use Setter injection to provide optional dependencies of an object, while use Constructor injection to provide mandatory dependency of an object, without which it can not work. This question is related to [Dependency Injection design pattern](http://javarevisited.blogspot.com/2012/12/inversion-of-control-dependency-injection-design-pattern-spring-example-tutorial.html) and mostly asked in context of Spring framework, which is now become an standard for developing Java application. Since Spring provides IOC container, it also gives you way to specify dependencies either by using setter methods or constructors. You can also take a look my [previous post](http://javarevisited.blogspot.com/2012/11/difference-between-setter-injection-vs-constructor-injection-spring-framework.html) on same topic.  
  
  
**12. What is difference between Factory and Abstract factory in Java**  
see  [here](http://javarevisited.blogspot.sg/2011/04/top-20-core-java-interview-questions.html) to answer this Java design pattern interview question.   
  
  
**13. When to use Adapter pattern in Java? Have you used it before in your project?**  
Use Adapter pattern when you need to make two class work with incompatible interfaces. Adapter pattern can also be used to encapsulate third party code, so that your application only depends upon Adapter, which can adapt itself when third party code changes or you moved to a different third party library. By the way this Java design pattern question can also be asked by providing actual scenario.  
 **14. Can you write code to implement producer consumer design pattern in Java?**  
Producer consumer design pattern is a concurrency design pattern in Java which can be implemented using multiple way. if you are working in Java 5 then its better to use Concurrency util to implement producer consumer pattern instead of plain old [wait and notify in Java](http://javarevisited.blogspot.sg/2011/05/wait-notify-and-notifyall-in-java.html).  Here is a good example of implementing [producer consumer problem using BlockingQueue in Java](http://javarevisited.blogspot.sg/2012/02/producer-consumer-design-pattern-with.html).  
  
**15. What is Open closed design principle in Java?**  
Open closed design principle is one of the SOLID principle defined by Robert C. Martin, popularly known as Uncle Bob. This principle advices that a code should be open for extension but close for modification. At first this may look conflicting but once you explore power of polymorphism, you will start finding patterns which can provide stability and flexibility of this principle. One of the key example of this is State and Strategy design pattern, where Context class is closed for modification and new functionality is provided by writing new code by implementing new state of strategy. See [this](http://javarevisited.blogspot.com/2011/11/great-example-of-open-closed-design.html)article to know more about Open closed principle.  
  
**16. What is Builder design pattern in Java? When do you use Builder pattern ?**  
Builder pattern in Java is another creational design pattern in Java and often asked in Java interviews because of its specific use when you need to build an object which requires multiple properties some optional and some mandatory. See [When to use Builder pattern in Java](http://javarevisited.blogspot.sg/2012/06/builder-design-pattern-in-java-example.html) for more details    
  
**17. Can you give an example of  SOLID design principles in Java?**  
There are lots of SOLID design pattern which forms acronym SOLID, read this [list of SOLID design principles for Java programmer](http://javarevisited.blogspot.sg/2012/03/10-object-oriented-design-principles.html)  to answer this Java interview question.  
 **18. What is difference between Abstraction and Encapsulation in Java?**  
I have already covered answer of this Java interview question in my previous post as [Difference between encapsulation and abstraction in Java](http://java67.blogspot.sg/2012/08/difference-between-abstraction-and-encapsulation-java-oops.html). See there to answer this question.   
  
  
This was my list of **10 popular design pattern interview question in Jav**a. I have not included MVC (Model View Controller) design pattern because that is more specific to J2EE and [Servlet JSP interview](http://javarevisited.blogspot.sg/2011/09/servlet-interview-questions-answers.html), but if you are going for any Java interview which demands experience in J2EE than you must prepare MVC design pattern. That's all on Java design pattern interview question and answers. Please let us know if you have any other interesting question on Java design pattern.  
  
Other **Java interview questions** post:

[How to reverse String in Java without using API](http://javarevisited.blogspot.sg/2012/01/how-to-reverse-string-in-java-using.html)

[Top 10 Java coding interview question](http://java67.blogspot.sg/2012/08/10-java-coding-interview-questions-and.html)

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

[What is Serialization in Java](http://javarevisited.blogspot.sg/2011/04/top-10-java-serialization-interview.html)

[What is Synchronization in Java with Example](http://javarevisited.blogspot.sg/2011/04/synchronization-in-java-synchronized.html)

[Can we override private method in Java](http://java67.blogspot.sg/2012/08/can-we-override-private-method-in-java.html)

Read more: <http://www.java67.com/2012/09/top-10-java-design-pattern-interview-question-answer.html#ixzz4DQTQrSil>