### Java/J2EE Interview Questions

### <http://www.javagyan.com/preparing-for-an-interview/java-j2ee-interview-questions>

This page is for Java/J2EE job seekers and interviewers and for people interested in self-assessments on Java/J2EE technologies. There are numerous books and articles on the market covering specific topics like Java, J2EE, EJB, Design Patterns, ANT, Maven, CVS, Multi-Threading, Servlets, JSP, emerging technologies like AOP (Aspect Oriented Programming), Test Driven Development (TDD), Dependency Injection Dl (aka IoC – Inversion of Control) etc. But from an interview perspective it is not possible to brush up on all these books where each book usually has from 300 pages to 600 pages. The basic purpose of this page is to cover question (and only questions)  on all the core concepts and key areas, which all Java/J2EE developers, designers and architects should be conversant with to perform well in their current jobs and to launch a successful career by doing well at interviews. The interviewer can also use this book to make sure that they hire the right candidate depending on their requirements.

**Multi-Threading**

* What language features are available to allow shared access to data in a multi-threading environment? (Hint: Synchronized block,Synchronized method,wait, notify)
* What is the difference between synchronized method and synchronized block? (Hint:Block on subset of data. Smaller code segment).
* What Java language features would you use to implement a producer (one thread) and a consumer (another thread) passing data via a stack? (Hint: wait, notify)

**Java language**

* What Java classes are provided for date manipulation? (Hint:Calendar, Date)
* What is the difference between String and StringBuffer? (Hint: mutable, efficient)
* How do you ensure a class is Serializable? (Hint:Implement Serializable)
* What is the difference between static and instance field of a class? (Hint:Per class vs. Per Object)
* What methods do you need to implement to store a class in Hashtable or HashMap? (Hint: hashCode(), equals()) .
* How do you exclude a field of a class from serialization? (Hint: transient)

**Inheritance**

* What is the difference between an Interface and an abstract base class? (Hint: interface inheritance, implementation inheritance.)
* What does overriding a method mean? (Hint: Inheritance)
* What about overloading? (Hint: different signature)

**Memory**

* What is the Java heap, and what is the stack? (Hint: dynamic, program thread execution.)
* Why does garbage collection occur and when can it occur? (Hint: To recover memory, as heap gets full.)
* If I have a circular reference of objects, but I no longer reference any of them from any executing thread, will these cause garbage collection problems? (Hint: no)

**Exceptions**

* What is the problem or benefits of catching or throwing type “java.lang.Exception”? (Hint: Hides all subsequent exceptions.)
* What is the difference between a runtime exception and a checked exception? (Hint: Must catch or throw checked exceptions.)

**JSP**

* What is the best practice regarding the use of scriptlets in JSP pages? Why? (Hint: Avoid)How can you avoid scriptlet code? (Hint:custom tags, Java beans)
* What do you understand by the term JSP compilation? (Hint: compiles to servlet code)

**Servlets**

* What does Servlet API provide to store user data between requests? (Hint: HttpSession)
* What is the difference between forwarding a request and redirecting? (Hint: redirect return to browser )
* What object do you use to forward a request? (Hint: RequestDispatcher)
* What do you need to be concerned about with storing data in a servlet instance fields? (Hint: Multi-threaded.)
* What’s the requirement on data stored in HttpSession in a clustered (distributable) environment? (Hint: Serializable)
* If I store an object in session, then change its state, is the state replicated to distributed Session? (Hint: No, only on setAttribute() call.)
* How does URL-pattern for servlet work in the web.xml? (Hint: /ddd/\* or \*.jsp)
* What is a filter, and how does it work? (Hint: Before/after request, chain.)

**JDBC**

* What form of statement would you use to include user-supplied values? (Hint: PreparedStatement)
* Why might a preparedStatement be more efficient than a statement? (Hint: Execution plan cache.)
* How would you prevent an SQL injection attack in JDBC? (Hint: PreparsedStatement )
* What is the performance impact of testing against NULL in WHERE clause on Oracle? (Hint: Full table scan. )
* List advantages and disadvantages in using stored procedures? (Hint: Pro: integration with existing dbase, reduced network trafficCon: not portable, mutliple language knowledge required )
* What is the difference between sql.Date, sql.Time, and sql.Timestamp? (Hint: Date only, time only, date and time )
* If you had a missing int value how do you indicate this to PreparedStatement? (Hint: setNull(pos, TYPE))
* How can I perform multiple inserts in one database interaction? (Hint: executeBatch)Given this problem: Program reads 100,000 rows, converts to Java class in list, then converts list to XML file using reflection. Runs out of program memory. How would you fix? (Hint: Read one row at time, limit select, allocate more heap (result set = cursor) )
* How might you model object inheritance in database tables? (Hint: Table per hierarchy, table per class, table per concrete class)

**JNDI**

* What are typical uses for the JNDI API within an enterprise application? (Hint: Resource management, LDAP access)
* Explain the difference between a lookup of these “java:comp/env/ejb/MyBean” and “ejb/MyBean”? (Hint: logical mapping performed for java:comp/env )
* What is the difference between new InitialContext() from servlet or from an EJB? (Hint: Different JNDI environments initialized EJB controller by ejb-jar.xml, servlet by web.xml.)
* What is an LDAP server used for in an enterprise environment? (Hint: authentication, authorization)
* What is authentication, and authorization? (Hint: Confirming identity, confirming access rights )

**EJB**

* What is the difference between Stateless and Stateful session beans (used?) (Hint: Stateful holds per client state )
* What is the difference between Session bean and Entity bean (when used?) (Hint: Entity used for persistence )
* With Stateless Session bean pooling, when would a container typically take a instance from the pool and when would it return it? (Hint: for each business method )
* What is the difference between “Required”, “Supports”, “RequiresNew” “NotSupported”, “Mandatory”, “Never”? (Hint: Needs transaction, existing OK but doesn’t need, must start new one, suspends transaction, must already be started, error if transaction)
* What is “pass-by-reference” and “pass-by-value”, and how does it affect J2EE applications? (Hint: Reference to actual object versus copy of object. RMI pass by value.)
* What EJB patterns, best practices are you aware of? Describe at least two? (Hint: Façade, delegate, value list, DAO, value object).
* Describe some issues/concerns you have with the J2EE specification? (Hint: Get their general opinion of J2EE)
* What do you understand by the term “offline optimistic locking” or long-lived business transaction? How might you implement this using EJB? (Hint: version number, date, field comparisons.)
* Explain performance difference between getting a list of summary information (e.g. customer list) via finder using a BMP entity vs. Session using DAO? (Hint: BMP: n+1 database reads, n RMI calls.)
* What is meant by a coarse-grained and a fine-grained interface? (Hint: Amount of data transferred per method call)

**XML/XSLT**

* What is the difference between a DOM parser and a SAX parser? (Hint: DOM: reads entire model, SAX: event published during parsing.)
* What is the difference between DTD and XML Schema? (Hint: level of detail, Schema is in XML.)
* What does the JAXP API do for you? (Hint: Parser independence. )What is XSLT and how can it be used? (Hint: XML transformation. )
* What would be the XPath to select any element called table with the class attribute of info? (Hint: Table[@class=’info’])

**JMS**

* How can asynchronous events be managed in J2EE? (Hint: JMS)
* How do transactions affect the onMessage() handling of a MDB? (Hint: Taking off queue. )
* If you send a JMS message from an EJB, and transaction rollback, will message be sent? (Hint: yes)
* How do you indicate what topic or queue MDB should react to? (Hint: deployment descriptor )
* What is the difference between a topic and a queue? (Hint: broadcast, single)

**SOAP**

* What is a Web service, and how does it relate to SOAP? (Hint: SOAP is the protocol.)
* What is a common transport for SOAP messages? (Hint: HTTP )
* What is WSDL? How would you use a WSDL file? (Hint: XML description of Web Service: interface and how to bind to it. )
* With new J2EE SOAP support what is: JAXR, JAX-RPC, and SAAJ? (Hint: registry, rpc, attachments)

**Security**

* Where can container level security be applied in J2EE application? (Hint: Web Uri’s, EJB methods)
* How can the current user be obtained in a J2EE application (Web and Enterprise)? (Hint: getUserPrincipal, getCallerPrincipal )
* How can you perform role checks in a J2EE application (Web and enterprise)? (Hint: IsUserInRole(), IsCallerInRole() )

**Design**

* Name some types of UML diagrams? (Hint: class, sequence, activity, use case)
* Describe some types of relationships can you show on class diagrams? (Hint: generalization, aggregation, uses)
* What is the difference between association, aggregation, and generalization? (Hint: Relationship, ownership, inheritance)
* What is a sequence diagram used to display? ( Hint: Object instance interactions via operations/signals)What design patterns do you use. Describe one you have used (not singleton)? (Hint: e.g. Builder, Factory, Visitor, Chain of Command )
* Describe the observer pattern and an example of how it would be used (Hint: e.g. event notification when model changes to view )
* What are Use Cases? (Hint: Define interaction between actors and the system )What is your understanding of encapsulation? (Hint: Encapsulate data and behavior within class )
* What is your understanding of polymorphism? (Hint: Class hierarchy, runtime determine instance )

**Process**

* Have you heard of or used test-driven development? (Hint: e.g. XP process )
* What development processes have you followed in the past? (Hint: Rational, XP, waterfall )
* How do you approach capturing client requirements? (Hint: Numbered requirements, use cases )
* What process steps would you include between the capture of requirements and when coding begins? (Hint: Architecture, Design, UML modeling, etc )
* How would you go about solving performance issue in an application? (Hint: Set goals, establish bench, profile application, make changes one at a time )
* What developer based testing are you familiar with (before system testing?) (Hint: Unit test discussion )
* How might you test a business system exposed via a Web interface? (Hint: Automated script emulating browser)
* What is your experience with iterative development? (Hint: Multiple iteration before release)

**Distributed Application**

* Explain a typical architecture of a business system exposed via Web interface? (Hint: Explain tiers (presentation, enterprise, resource) Java technology used in each tiers, hardware distribution of Web servers, application server, database server )
* Describe what tiers you might use in a typical large scale (> 200 concurrent users) application and the responsibilities of each tier (where would validation, presentation, business logic, persistence occur). (Hint: Another way of asking same question as above if their answer wasn’t specific enough)
* Describe what you understand by being able to “scale” an application? How does a J2EE environment aid scaling? (Hint: Vertical and Horizontal scaling. Thread management, clustering, split tiers )
* What are some security issues in Internet based applications? (Hint: authentication, authorization, data encryption, denial service, xss attacks, SQL injection attacks )

**General**

* What configuration management are you familiar with? (Hint: e.g. CVS, ClearCase )
* What issue/tracking process have you followed? (Hint: Want details on bug recording and resolution process).
* What are some key factors to working well within a team? (Hint: Gets a view on how you would work within interviewer’s environment.)
* What attributes do you assess when considering a new job? (what makes it a good one)? (Hint: Insight into what motivates you.)
* What was the last computing magazine you read? Last computing book?
* What is a regular online magazine/reference you use? (Hint: Understand how up to date you keep yourself.)

------------------------------------------------------------------------------------------------------------------------------

### General and general terms questions

### <http://www.jobinterviewplus.com/tests/faq_architect_java.html>

Architect interview is slightly different from all other interview types. Interviewer is looking for ability of the candidate to think independently on top of pure technical knowledge. Most of the questions are open-ended, prompting the interviewee to discussion about different aspects of Java development. Other side of the interview is general questions about position of the architect within the organization. Some questions do not have clear, direct or single answer and require discussion with the interviewer. On top of questions mentioned here you may be asked generic OO questions (what is class, what is polymorphism etc.)

* What distinguishes "good architecture" from "bad architecture"?

This is an open-ended question. There are few aspects of "good" architecture:

* 1. Shall address functional product requirements
  2. Shall address non-functional product requirements, such as performance, scalability, reliability, fault tolerance, availability, maintainability, extensibility
  3. Shall be simple and comprehendible (to support maintainability and extensibility)
  4. Shall be well structured (support multiple tiers, parallel development etc.)
  5. Shall be detailed enough to share with different levels of organizational structure (marketing, sales, development, management)

"Bad" architecture is basically opposite to "good" architecture.

* How much experience do you have with Enterprise applications? Another variant of this questions is: "Tell me about projects where you worked with J2EE?" Yet another version: "What, when and how made using EJB?"

Interviewer is looking for your experience with designing J2EE applications and your experience with J2EE technologies and general terms. This is often start of the discussion and bridge to the technical section of the questions.

* What is scalability?
* What is high-availability? How is it different from scalability?
* What is the fault tolerance?
* What resources are used to keep up to date with J2EE technology?

You may mention design pattern books, such as "Core EJB Patterns" and web sites, such as <http://www.theserverside.com>

### Specific technical questions

* What modeling tools you are familiar with? What version of TogetherJ (Rational Rose etc.) have you used?
* If stateless session bean more scalable than stateful session beans?

This is very popular questions that leads to some confusion. According to the second edition of "Core J2EE Patterns" and contrary to popular belief, stateful session beans are not less scalable than stateless session bean. The reason for that is life cycle of either type is controlled by Application Server and control of life cycle is what defines the scalability of the application

* What's the difference between EJB 1.1 and EJB 2.0?

There are many differences. Some key points you want to mention are:

* 1. New CMP model
  2. EJB Query Language
  3. Local interfaces
  4. EJBHome methods
  5. Message Driven Beans (MDB) support
* What transaction isolation levels do you know?

none, repeatable read, read committed, read uncommitted, serializable

* What transaction attributes do you know?

requires new, required, supports, not supported, mandatory, never

* What is the difference between optimistic lock and pessimistic lock?

Optimistic lock is an implicit lock that tries to make best assumption about locking strategy and minimize time spent in lock of resource. Optimistic lock is usually implemented with some kind of timestamp strategy. Pessimistic lock is an explicit lock that set by client.

* What are entity beans. Are there any issues with them?

Typical reaction to this question is very expressive answer that entity beans should not be used. There are many performancy implications with entity beans if used incorrectly. One of the famous problems are "n+1 call problem" Inter-entity bean call is very expensive operation and should be avoided.

* What core design patterns do you know?

Architect must know at least some basic design patters used in J2EE development, e.g. Business Delegate, Session Facade, VO, List Handler, DTO, Composite Entity, etc.

* Where business logic should reside?

Typical answer is "in business tier" This usually opens series of questions like: What is business logic, how to determine business logic, how business logic is different from persistent logic etc.

* What is JDO?

JDO is Java Data Object - persistent framework that is alternative to idea of entity beans

* What is the difference between JSP and servlet? When to use what?

JSP is compiled into servlet. JSP are better suit to view of information, while servlets are better for controlling stuff.

* Does the J2EE platform support nested transactions?

No.

* Can you use synchronization primitives in my enterprise beans?

No.

* Why all major application server vendors provide custom class loaders in addition to system jvm class loader?

System one does not support hot deployment.

### Performance questions

* What are performance problems in J2EE and how to solve them?
* What are Entity beans performance pitfalls?
* What performance pattern do you know?

### Design Pattern questions

* Can you use singleton in EJB?

Yes, but should not (explain why)

* What is MVC pattern and why M, V and C need to be separated?
* Describe Business Delegate pattern (or any other pattern)
* How to prevent double submission of the form from JSP page? (or describe Synchronizer Token pattern)

<http://javajeetechnology.weebly.com/architect-interview-questions.html>

**1. How did you achieved high and good performance at different layers? Briefly explain couple of instances from your experience.  
  
2. Did you involve in architecture or technology selection? If yes, how is your approach?  
  
3. Did you had any rich web application development experience? If yes, what is your experience and lessons learned?  
  
4. What kind of problems you have faced and handled effectively in your experience? Explain briefly any two or three from your experience.  
  
5.Explain the Clustering and Load Balancing with respect to any Application Server**   
  
**6.Java security questions:**  
    **a)Where can container level security be applied in J2EE application? (**Hint: Web Uri’s, EJB methods**)**   
**b)How can the current user be obtained in a J2EE application (Web and Enterprise)? (**Hint: getUserPrincipal, getCallerPrincipal)   
**c)How can you perform role checks in a J2EE application (Web and enterprise)?** (Hint: IsUserInRole(), IsCallerInRole() )   
  
**7.Development process:**  
**a)What development processes have you followed in the past?** (Hint: Rational, XP, waterfall, Agile) **b)How would you go about solving performance issue in an application?** (Hint: Set goals, establish bench, profile application, make changes one at a time )  **c)What is your experience with iterative development?** (Hint: Multiple iteration before release) **d)What process steps would you include between the capture of requirements and when coding begins?**(Hint: Architecture, Design, UML modeling, etc )  
  
**8.Distributed application:**  
**a) Describe what tiers you might use in a typical large scale (> 200 concurrent users) application and the responsibilities of each tier (where would validation, presentation, business logic, persistence occur).**Hint: Explain tiers (presentation, enterprise, resource) Java technology used in each tiers, hardware distribution of Web servers, application server, database server  
   **b) Describe what you understand by being able to “scale” an application? How does a J2EE environment aid scaling? (**Hint: Vertical and Horizontal scaling. Thread management, clustering, split tiers **)   
  c) What are some security issues in Internet based applications? (**Hint: authentication, authorization, data encryption, denial service, XSS attacks, SQL injection attacks **)**  
  
**9.Differentiate between the terms 'Design Patterns', 'Framework' and 'Architecture'.**  
**Design Pattern:** The various solutions arrived at for the known problem. This helps to avoid re-inventing the wheel. The risk-free solution can be easily used by others. For example, singleton is the design pattern that you can use instantly to enfore one object per class. You do not need to think of this on your own.  
**Framework:** A framework is a structure or set of rules, used to solve or address complex issues. It is basically a reusable designf for the J2EE applications. For example, Struts, JSF,etc., are the frameworks. You can use these frameworks based on the requirements of your application and each has its own set of advantages.  
**Architecture:** It is a design that describes how the various components in the application fit together. For example, MVC is an architecture which is helpful to define how the components interact within the application  
  
**10.JDBC Vs Hibernate**  
                    **Why is Hibernate better than JDBC**  
1) Relational Persistence for JAVA  
Working with both Object-Oriented software and Relational Database is complicated task with JDBC because there is mismatch between how data is represented in objects versus relational database. So with JDBC, developer has to write code to map an object model's data representation to a relational data model and its corresponding database schema. Hibernate is flexible and powerful ORM solution to map Java classes to database tables. Hibernate itself takes care of this mapping using XML files so developer does not need to write code for this.  
2) Transparent Persistence  
The automatic mapping of Java objects with database tables and vice versa is called Transparent Persistence. Hibernate provides transparent persistence and developer does not need to write code explicitly to map database tables tuples to application objects during interaction with RDBMS. With JDBC this conversion is to be taken care of by the developer manually with lines of code.  
3) Support for Query Language  
JDBC supports only native Structured Query Language (SQL). Developer has to find out the efficient way to access database, i.e to select effective query from a number of queries to perform same task. Hibernate provides a powerful query language Hibernate Query Language (independent from type of database) that is expressed in a familiar SQL like syntax and includes full support for polymorphic queries. Hibernate also supports native SQL statements. It also selects an effective way to perform a database manipulation task for an application.  
4) Database Dependent Code  
Application using JDBC to handle persistent data (database tables) having database specific code in large amount. The code written to map table data to application objects and vice versa is actually to map table fields to object properties. As table changed or database changed then it’s essential to change object structure as well as to change code written to map table-to-object/object-to-table. Hibernate provides this mapping itself. The actual mapping between tables and application objects is done in XML files. If there is change in Database or in any table then the only need to change XML file properties.  
5) Maintenance Cost  
With JDBC, it is developer’s responsibility to handle JDBC result set and convert it to Java objects through code to use this persistent data in application. So with JDBC, mapping between Java objects and database tables is done manually. Hibernate reduces lines of code by maintaining object-table mapping itself and returns result to application in form of Java objects. It relieves programmer from manual handling of persistent data, hence reducing the development time and maintenance cost.  
6) Optimize Performance  
Caching is retention of data, usually in application to reduce disk access. Hibernate, with Transparent Persistence, cache is set to application work space. Relational tuples are moved to this cache as a result of query. It improves performance if client application reads same data many times for same write. Automatic Transparent Persistence allows the developer to concentrate more on business logic rather than this application code. With JDBC, caching is maintained by hand-coding.  
7) Automatic Versioning and Time Stamping  
By database versioning one can be assured that the changes done by one person is not being roll backed by another one unintentionally. Hibernate enables developer to define version type field to application, due to this defined field Hibernate updates version field of database table every time relational tuple is updated in form of Java class object to that table. So if two users retrieve same tuple and then modify it and one user save this modified tuple to database, version is automatically updated for this tuple by Hibernate. When other user tries to save updated tuple to database then it does not allow to save it because this user does not has updated data. In JDBC there is no check that always every user has updated data. This check has to be added by the developer.  
8) Open-Source, Zero-Cost Product License  
Hibernate is an open source and free to use for both development and production deployments.  
9) Enterprise-Class Reliability and Scalability  
Hibernate scales well in any environment, no matter if use it in-house Intranet that serves hundreds of users or for mission-critical applications that serve hundreds of thousands. JDBC can not be scaled easily.  
  
**Disadvantages of Hibernate**  
1) Steep learning curve.  
2) Use of Hibernate is an overhead for the applications which are :  
• simple and use one database that never change  
• need to put data to database tables, no further SQL queries  
• there are no objects which are mapped to two different tables  
Hibernate increases extra layers and complexity. So for these types of applications JDBC is the best choice.  
3) Support for Hibernate on Internet is not sufficient.  
4) Anybody wanting to maintain application using Hibernate will need to know Hibernate.  
5) For complex data, mapping from Object-to-tables and vise versa reduces performance and increases time of conversion.  
6) Hibernate does not allow some type of queries which are supported by JDBC. For example It does not allow to insert multiple objects (persistent data) to same table using single query. Developer has to write separate query to insert each object.  
  
**11.Design and describe a technical solution for high end performance system to get 2ms web reponse**

* + **Requirement is for trading application to display market update**

**12.How to determine heap size requirement for JMS application**  
  
**13.Whether Websphere MQ breaks and regroup the large payload messages**  
  
**14.What are parameters for performance tuning in Websphere Application sever 7.0**  
  
**15.What tools you have used for memory dump analysis**  
  
**16.Have you used profiling tools and what attributes do you check**  
  
**17.How to identify memory leaks in the application**  
  
**18.Is websphere provides mechanism for alarming the java heap grown large enough**  
  
**19.How to use singleton object in cluster environment**  
  
**20.What is approach to use signleton whithout declaring the explicit synchronisation**  
  
**21.What do you mean by memory inconsistency and how do you avoid the same**  
  
**22.What is authentication and authorization**  
  
**23.What is implementation methods details for handling security in webservices.**  
  
**24.What is transactional level attributes and any difference in ejb 3.0 and ORM**  
  
**25.What's new in EJB 3.0 over EJB 2.0**  
  
**26.Which core design pattern heavily used in spring framework.**  
  
**27.What design pattern exception handler uses. -COR**  
  
**28.What design pattern database connection uses**  
  
**29.How command pattern is different from chain of responsibility**  
  
**30.What is significance of factory design pattern over compile time if and else block for resolving references**   
  
**31.What would be your approach to address the solution for a given requirement from client? What would be your start point?**  
**Ans:**  
For technical team, start point is Business Requirement document /functional Specification.  
a.       Review BRD and create an initial architect document and get it approved.   
b.       In agility environment, propose module/component and create high level technical design document and get it reviewed by core architect team and approved by Business user.  
c.        Create an application prototype model (real working code) and get business user nod.   
d.        In parallel to (c) Create low level design document and get it reviewed by core architect team.  
e.        Assign team for project implementation based on technical design   
f.         Follow iteration development process  
g.       Perform integration testing  
h.       Perform system testing   
i.         User Acceptance Testing by business user   
j.         Deployment to Production System   
k.        SLA/Maintenance of application   
  
**32.      What are contents of initial architecture document or high level design document?**  
  
**33.      What are contents of low level design document?**   
  
**34.      What is difference between logical design and technical design?**   
  
**35.      What is agile development model?**   
  
**36.      What do you mean by non functional requirement?**  
  
**37.      What are differences between aspects oriented programming and OOP?**   
**Ans:**  
Aspect-Oriented Programming (AOP) :  
complements Object-Oriented Programming (OOP) and allows users to implement custom aspects for cross cutting concerns such as Transaction management, logging, security and exception handling. In spring framework it provides declarative enterprise services, especially as a replacement for EJB declarative services. Actions are taken at a particular join point such as "around," "before" and "after" advice to implement custom aspects.   
Object-oriented programming (OOP) is a programming paradigm using "objects" – data structures consisting of data fields and methods together with their interactions – to design applications using Abstraction, Encapsulation, inheritance and polymorphism.   
  
**38.      What is Inversion of control and dependency ingestion?**   
  
**39.      Explain application design considerations for deploying application in cluster environment and how to sync the resources.**   
  
**40.  Explain Business delegate pattern**   
  
**41.  Explain Command pattern**   
  
**42.  What is Session facade design pattern?**   
  
**43.  What is Decorator design pattern?**   
  
**44.  What is Observer design pattern?**   
  
**45.  How do you implement common validator in struts 2.x?**   
  
**46.  What are new features of EJB 3.0 over EJB 2.0**   
  
**47.  Why do we need to use EJB in application?**   
  
**48.  What is use of MDB bean and how it is used and implemented?**   
  
**49.  What is dirty read?**   
  
**50.  What is difference between XA Data source and non XA data source**   
  
**51.  Explain Transaction management in database and EJB.**   
  
**52.  What is cluster and non cluster index?**   
  
**53.  What is de-normalization and why do you need to make database de-normalized form?**   
  
**54.  How do you manage multiple indexes on production database table which has multiple read/write operation?**   
  
**55.  What is materialized view in oracle**   
  
**56.  How do you perform SQL query tunings and optimization?**   
  
**57.  What is AWR Report?**   
**Ans:**The AWR(Automatic Workload Repository) introduced in Oracle 10g is used to collect performance statistics including: ·      
Wait events used to identify performance problems. ·          
                    Time model statistics indicating the amount of DB time associated with a process from the V$SESS\_TIME\_MODEL and V$SYS\_TIME\_MODEL views. ·    
Active Session History (ASH) statistics from the V$ACTIVE\_SESSION\_HISTORY view. ·           
Some system and session statistics from the V$SYSSTAT and V$SESSTAT views. ·          
Object usage statistics. ·        
Resource intensive SQL statements.   
  
**58.  What are application performance considerations in each tier?**  
  
**59.  If user reported that the site is slow, how would you approach to solve and identify the root cause? ( you do not have access to prod system or data to debug)**  
  
**60.  What are project risk and how to mitigate the risk**   
  
**61.  What is software project costing? Explain one which used in your project.**   
  
**62.  What is difference between incidence based and problem based project support management.**  
  
**63.  Making MQ messages transactional in synchronous communication environment. Let's say I have an application which need to send out messages to two databases oracle and sybase, part of record is written in oracle remaining part of same message goes in sybase, so let's say if sybase transaction failed then how would you roll back whole transaction so that data in oracle  is not inconsistency with**  
**sybase database.**  
  
**64.  What are standard architectural questions:-**  
**Ans:**

* Characteristics of a good architect design/ Consideration for designing a good application architecture
  + Should address functional requirements
  + Should address non functional requirements
  + Should address cross cutting concerns
* Application Architecture layer - [Key principles of Software Architecture](http://msdn.microsoft.com/en-us/library/ee658124)
  + Layered Architecture: A decomposition of services such that most interactions occur only between neighboring layers.
  + Enterprise Process-Centric: A business process manages the interactions between multiple intra-enterprise applications, services, sub-processes and users.
  + Component-Based Architecture -Decomposes application design into reusable functional or logical components that expose well-defined communication interfaces.
  + Service-Oriented Architecture (SOA) -applications that expose and consume functionality as a service using contracts and messages.
* [Non functional requirements](http://www.javagyan.com/tutorials/nonfunctionalrequirements) of the application design- at-least highlight important ones
  + Scalability and capacity planning - considered for high user load
  + High Availability - downtime and uptime of system
  + Reliability and Usability -bug free system and easy to use
  + Maintainability and Extensibility - quality code, standard routines  and upgradable, new releases
  + Performance - Response, processing and tun around time of application
* Cross cutting concerns of the application -some key concerns addressing
  + Transaction processing
  + Internationalization and localization which includes Language locale
  + Information security -authentication and authorization
  + Caching -local,server, network, database caching
  + Logging
  + Exception Management
* UML Diagram and tools- key diagrams used by technical team
  + Diagram
    - Class diagram
    - Sequence diagram
    - Deployment diagram
    - Use case diagram
  + Tools
    - Rational Architect
    - Argo UML (freebees)
    - Visual Paradigm
    - Star UML etcs
  + Reverse Engineering and code generation
* [J2EE Core Design Pattern -as per application layer](http://www.corej2eepatterns.com/Patterns2ndEd/PatternRelationships.htm)
  + Presentation Tier
    - Intercepting Filter -intercepts incoming requests and outgoing responses and applies a filter.
    - Context Object-encapsulates state in a protocol-independent way to be shared throughout your application
    - Front Controller-a container to hold the common processing logic that occurs within the presentation tier
    - Application Controller -centralizes control, retrieval, and invocation of view and command processing
    - View Helper -encourages the separation of formatting-related code from other business logic
    - Composite View -suggests composing a View from numerous atomic pieces
    - Dispatcher View -defers business processing until view processing has been performed.
    - Service To Worker
  + Business Tier
    - Business Delegate -reduces coupling between remote tiers and provides an entry point for accessing remote services in the business tier.
    - Service Locator -encapsulates the implementation mechanisms for looking up business service components.
    - Session Facade -provides coarse-grained services to the clients by hiding the complexities of the business service interactions.
    - Application Service -centralizes and aggregates behavior to provide a uniform service layer to the business tier services.
    - Business Object -implements your conceptual domain model using an object model.
    - Composite Entity -implements a [Business Object](http://www.corej2eepatterns.com/Patterns2ndEd/BusinessObject.htm) using local entity beans and POJOs.
    - Transfer Object -provides the best techniques and strategies to exchange data across tiers
    - T O Assembler -constructs a composite [Transfer Object](http://www.corej2eepatterns.com/Patterns2ndEd/TransferObject.htm) from various sources
    - Value List Handler -uses the GoF iterator pattern to provide query execution and processing services.
  + Integration Tier
    - Data Access Object -enables loose coupling between the business and resource tiers
    - Service Activator -enables asynchronous processing in your enterprise applications using JMS.
    - Domain Store -provides a powerful mechanism to implement transparent persistence for your object model.
    - Web Service Broker -exposes and brokers one or more services in your application to external clients as a web service using XML and standard web protocols
* [J2EE Application Design Pattern](http://www.allapplabs.com/java_design_patterns/java_design_patterns.htm)
  + Creational Patterns
    - Factory Pattern
    - Abstract Factory Pattern
    - Singleton Pattern
    - Builder Pattern
    - Prototype Pattern
  + Structural Patterns
    - Adapter Pattern
    - Bridge Pattern
    - Composite Pattern
    - Decorator Pattern
    - Facade Pattern
    - Flyweight Pattern
    - Proxy Pattern
  + Behavioral Patterns
    - Chain of Responsibility Pattern
    - Command Pattern
    - Interpreter Pattern
    - Iterator Pattern
    - Mediator Pattern
    - Momento Pattern
    - Observer Pattern
    - State Pattern
    - Strategy Pattern
    - Template Pattern
    - Visitor Pattern
* J2EE Application Development Framework
  + Spring 3.0
  + Jakarta Struts 2.0 -Servlet life cycle
  + EJB 3.0
  + JSF
  + JPA -Hibernate/ibatis
  + AJAX
  + GWT
  + JSON
  + FLEX -actionscript
* J2EE Application Server - deployment and administration
  + Apache Tomcat
  + Websphere Application Server
  + JBOSS
  + WebLogic
* Database Systems - SQL- complex queries -PL/SQL
  + Oracle 9i/10g/11g
  + SQL Server 2000/2005/2008
  + MySQL 5.5
* Application build management
  + ANT - Build tools -build.xml/properties  MAVEN -pom.xml
  + HUDSON - continuous integration build server
* Source Version controls - resolving conflict, branching,tagging
  + SVN
  + CVS
  + VSS
  + GIT
* SOA -implementation and approach
  + [Webservices](http://www.javaperformancetuning.com/tips/webservice.shtml) -AXIS 2 framework, webservice annotation  RPC style and [Restful services](http://soa.dzone.com/articles/restful-web-services-best)
* [JMS](http://java.sun.com/developer/technicalArticles/Ecommerce/jms/) - MQ - MDB
  + Point-to-Point (Queue destination): In this model, a message is delivered from a producer to one consumer.
  + Publish/Subscribe (Topic destination): In this model, a message is delivered from a producer to any number of consumers.

Some generic knowledge test questions

* Java collection frameworks
  + List, Set,Tree -
    - synchronized collections
    - sorted collections
    - collections methods
  + use case for selecting one above other like arraylist over linkedlist (vector deprecated)
  + Order of processing time -add, remove and search
  + sorting and searching techniques - Heap merge sort, b-tree search
* Exception Handling in JAVA
  + compile time exception
  + run time exception
* Questions related to OOPS
  + Abstraction
    - abstract class, method
  + Encapsulation
    - public,private,protected
  + Inheritance
    - extend
    - implements
  + Polymorphism
    - Overriding - method
    - Overloading -method,class
* Questions related to multi-threading - synchronization, deadlock handling, runnable interface and thread class
* stateless entity bean and stateful entity bean scalability?
* Transaction isolation levels
  + none,repeated,read, read committed, read uncommitted, serializable
* Transaction attributes
  + new, required,support,mandatory,never,not supported
* Optimistic and pessimistic locking requirements
  + Optimistic is implicit time-stamp based
  + Pessimistic - explicit and client control
* Use of custom classloader in application server- for hot deployment
* design consideration and best practices for web services
* Tools for monitoring code quality (PMD, find bug, checkstyle, -sonar)

**65.   What is the difference between optimistic lock and pessimistic lock?**  
**Ans:**Optimistic lock is an implicit lock that tries to make best assumption about locking strategy and minimize time spent in lock of resource. Optimistic lock is usually implemented with some kind of timestamp strategy. Pessimistic lock is an explicit lock that set by client.  
  
**66.   What modeling tools you are familiar with? What version of TogetherJ (Rational Rose etc.) have you used?**  
  
**67.   If stateless session bean more scalable than stateful session beans?**   
**Ans:**This is very popular questions that leads to some confusion. According to the second edition of "Core J2EE Patterns" and contrary to popular belief, stateful session beans are not less scalable than stateless session bean. The reason for that is life cycle of either type is controlled by Application Server and control of life cycle is what defines the scalability of the application  
  
**68.   What's the difference between EJB 1.1 and EJB 2.0?**   
**Ans:**There are many differences. Some key points you want to mention are:

* 1. New CMP model
  2. EJB Query Language
  3. Local interfaces
  4. EJBHome methods
  5. Message Driven Beans (MDB) support

**69.  What distinguishes "good architecture" from "bad architecture"?**  
**Ans:**This is an open-ended question. There are few aspects of "good" architecture:

* 1. Shall address functional product requirements
  2. Shall address non-functional product requirements, such as performance, scalability, reliability, fault tolerance, availability, maintainability, extensibility
  3. Shall be simple and comprehendible (to support maintainability and extensibility)
  4. Shall be well structured (support multiple tiers, parallel development etc.)
  5. Shall be detailed enough to share with different levels of organizational structure (marketing, sales, development, management)

"Bad" architecture is basically opposite to "good" architecture.