Java-Success.com

Industrial strength Java/JEE Career Companion to open more doors

Register | Login | Logout | Contact Us

search here … Go

Home | Java FAQs | 600+ Java Q&As | Career | Tutorials | Member | Why?

Can u Debug? | Java 8 ready? | Top X | Productivity Tools | Judging Experience?

Home › Interview › JEE Interview Q&A › JDBC › JDBC Overview Interview Questions and Answers

JDBC Overview Interview Questions and Answers

Posted on December 12, 2014 by Arulkumaran Kumaraswamipillai — No Comments ↓

Q. What is JDBC? How do you connect to a database?
A. JDBC stands for **J**ava **D**ata**B**ase **C**onnectivity. It is an API which provides easy connection to a wide range of databases. To connect to a database we need to load the appropriate driver and then request for a connection object. The *Class.forName(….)* will load the driver and register it with the *DriverManager*.

```
Class.forName(“oracle.jdbc.driver.OracleDriver”);
String url = jdbc:oracle:thin:@hostname:1526:myDB
Connection myConnection =  DriverManager.getConne

```

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

open all | close all
⊞ Ice Breaker Interview
⊞ Core Java Interview (
⊟ JEE Interview Q&A (3
  ⊞ JEE Overview (2)
  ⊞ Web basics (8)
  ⊞ WebService (11)
  ⊞ JPA (2)
  ⊞ JTA (1)
  ⊟ JDBC (4)
    ♦ 12 FAQ JDBC
    JDBC Overview
    NamedParamete
    Spring, JavaCon
  ⊞ JMS (5)
  ⊞ JMX (3)
  ⊞ JNDI and LDAP (1)
⊞ Pressed for time? Jav
⊞ SQL, XML, UML, JSC
⊞ Hadoop & BigData Int
⊞ Java Architecture Inte
⊞ Scala Interview Q&As

The **driver jar file** (e.g. ojdbc14.jar) needs to be in the classpath.

Q. What is a **datasource**?
A. The **DataSource** interface provides an alternative to the ***DriverManager*** for making a connection. *DataSource* makes the code more portable than the *DriverManager* because it works with JNDI and it is created, deployed and managed separately from the application that uses it. If the *DataSource* location changes, then there is no need to change the code, but change the configuration properties in the server. This makes your application code easier to maintain. *DataSource* allows the use of connection pooling and support for distributed transactions. A *DataSource* is not only a database but also can be a file or a spreadsheet. A *DataSource* object can be bound to JNDI and an application can retrieve and use it to make a connection to a database. JEE application servers provide tools to define your *DataSource* with a JNDI name. When the server starts it loads all the *DataSources* into the application server's JNDI service.

DataSource configuration properties are shown below:

JNDI Name = jdbc/myDataSource
URL = jdbc:oracle:thin:@hostname:1526:myDB
UserName, Password
Implementation class name =
oracle.jdbc.pool.OracleConnectionPoolDataSource
Jar file in the Classpath = ojdbc14.jar
Connection pooling settings like = minimum pool size, maximum pool size, connection timeout, statement cache size etc.

Once the DataSource has been set up, then you can get the connection object as follows:

```
Context ctx = new InitialContext(); //JNDI contex
DataSource ds = (DataSource)ctx.lookup("jdbc/myDa
Connection myConnection = ds.getConnection(“usern

```

Spring, Hibernate, & I
Testing & Profiling/Sa
Other Interview Q&A 1
▶Free Java Interviev

16 Technical Key Areas

open all | close all
- Best Practice (6)
- Coding (26)
- Concurrency (6)
- Design Concepts (7)
- Design Patterns (11)
- Exception Handling (3
- Java Debugging (21)
- Judging Experience Ir
- Low Latency (7)
- Memory Managemen
- Performance (13)
- QoS (8)
- Scalability (4)
- SDLC (6)
- Security (13)
- Transaction Managen

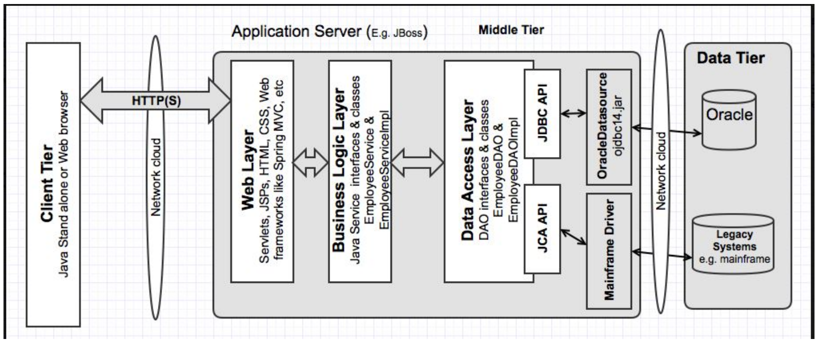
80+ step by step Java Tutorials

open all | close all
- Setting up Tutorial (6)
- Tutorial - Diagnosis (2
- Akka Tutorial (9)
- Core Java Tutorials (2
- Hadoop & Spark Tuto
- JEE Tutorials (19)
- Scala Tutorials (1)
- Spring & HIbernate Tu

Q. Why should you prefer using a DataSource?
A. **Best practice**: In a very basic application a Connection obtained from a DataSource and a DriverManager are identical. But, the JEE best practice is to use DataSource because of its portability, better performance due to pooling of valuable resources, and the JEE standard requires that applications use the container's resource management facilities to obtain connections to resources. Every major web application container provides pooled database connection management as part of its resource management framework.

Design Pattern: JDBC architecture decouples an abstraction from its implementation so that the implementation can vary independent of the abstraction. This is an example of the bridge design pattern. The JDBC API provides the abstraction and the JDBC drivers provide the implementation. New drivers can be plugged-in to the JDBC API without changing the client code.

Q. Have you used a Data Access Object (DAO) pattern? Why is it a best practice to use a DAO pattern
A. A DAO class provides access to a particular data resource in the **resource tier** or **data tier(**e.g. relational database, XML, mainframe, etc) without coupling the resource's API to the business logic in the middle tier. A tier is a physical machine, whereas a layer is logical.



jdbc overview

For example, you may have a *EmployeeServiceImpl* in the

⊞ Tools Tutorials (19)
⊞ Other Tutorials (45)

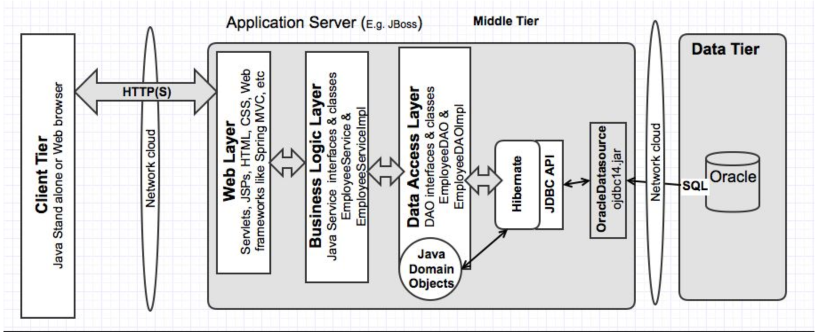
100+ Java pre-interview coding tests

open all | close all
⊞ Can you write code? (
⊞ ♦ Complete the given
⊞ Converting from A to I
⊞ Designing your classe
⊞ Java Data Structures
⊞ Passing the unit tests
⊞ What is wrong with th
⊞ Writing Code Home A
⊞ Written Test Core Jav
⊞ Written Test JEE (1)

How good are your .....?

open all | close all
⊞ Career Making Know-
⊞ Job Hunting & Resum

business logic layer with business logic, and uses *EmployeeDAO* in the data access layer for accessing data in the data tier. *EmployeeServiceImpl* uses the interface *EmployeeDAO* as opposed to the implementation. This is the best practice of "coding to interface not implementation"*.* If your data resource change from a database to a *Mainframe* system, then reimplementing *EmployeeDAO* for a different data access mechanism (to use a mainframe Connector) would have little or no impact on any classes like *EmployeeServiceImpl* that uses *EmployeeDAO*. *EmployeeDAOImpl* can even decide to use hibernate framework instead of using the JDBC directly. The *EmployeeServiceImpl* will not be impacted as long as the contract in *EmployeeDAO* is met.


Application Server (E.g. JBoss)
Middle Tier
Client Tier
Java Stand alone or Web browser
HTTP(S)
Network cloud
Web Layer
Servlets, JSPs, HTML, CSS, Web frameworks like Spring MVC, etc
Business Logic Layer
Java Service interfaces & classes EmployeeService & EmployeeServiceImpl
Data Access Layer
DAO interfaces & classes EmployeeDAO & EmployeeDAOImpl
Java Domain Objects
Hibernate
JDBC API
OracleDatasource
ojdbc14.jar
Network cloud
SQL
Oracle
Data Tier

Hibernate with jdbc overview

Inversion of control (IoC) frameworks like Spring framework promotes the design principle of "code to interface not implementation".

Q. What are the best practices relating to exception handling to make your DAOs more robust and maintainable?
A.

- If you catch an exception in your DAO code, never ignore it or swallow it because ignored exceptions are hard to troubleshoot. DAO class methods should throw checked exceptions only if the caller can reasonably recover from the exception or reasonably handle it (e.g. retry operations in optimistic concurrency control). If the caller cannot handle the exception in a meaningful way, consider throwing a runtime (i.e. unchecked) exception. For example, Hibernate 3 exceptions are all runtime exceptions.

- DAO methods should not throw low level JDBC exceptions like *java.sql.SQLException*. A DAO should encapsulate JDBC rather than expose it to rest of the application. Use chained exceptions to translate low-level exceptions into high-level checked exceptions or runtime exceptions. DAO methods should not throw *java.lang.Exception* because it is too generic and does not convey any underlying problem.
- Log your exceptions, configuration information, query parameters, etc.

Q. What are JDBC Statements? What are different types of statements? How can you create them?
A. A **statement** object is responsible for sending the SQL statements to the Database. Statement objects are created from the connection object and then executed.

```
Statement stmt = myConnection.createStatement();
ResultSet rs = stmt.executeQuery(“SELECT id, name

stmt.executeUpdate(“INSERT INTO (field1,field2) v

```

The types of statements are:

- Statement (regular statement as shown above) .
- **PreparedStatement** (more efficient than statement due to pre-compilation of SQL and prevents SQL injection attack. Always use this over a statement) .
- **CallableStatement** (to call stored procedures on the database).

To use prepared statement:

```
PreparedStatement prepStmt =
                  myConnection.prepareStatement("S
prepStmt.setInt(1, 1245);
```

Callable statements are used for calling stored procedures.

```
CallableStatement calStmt = myConnection.prepareC
ResultSet rs = cs.executeQuery();

```

Popular Posts

♦ 11 Spring boot interview questions & answers

825 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

766 views

18 Java scenarios based interview Questions and Answers

400 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

388 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

295 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

285 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

279 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

239 views

001B: ♦ Java architecture & design concepts interview questions & answers

201 views

Bio **Latest Posts**

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ Spring JMS with Websphere MQ Listener (Receiver or Subscriber)

♦ Why do Proxy, Decorator, Adapter, Bridge, and Facade design patterns look very similar? What are the differences? ›

Posted in JDBC, member-paid

Leave a Reply

Logged in as geethika. Log out?

Comment

Post Comment

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from “Blah blah” to “Wow”? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.

↑