JMS Interview Questions

1) What is JMS?

Java Message Service is the new standard for interclient communication. It allows J2EE application components to create, send, receive, and read messages. It enables distributed communication that is loosely coupled, reliable, and asynchronous.

2) What type messaging is provided by JMS

Both synchronous and asynchronous

3) How may messaging models do JMS provide for and what are they?

JMS provides for two messaging models, publish-and-subscribe and point-to-point queuing.

4) What are the types of messaging?

There are two kinds of Messaging:

- Synchronous messaging involves a client that waits for the server to respond to a message.
- Asynchronous messaging involves a client that does not wait for a message from the server. An event is used to trigger a message from a server.

5) What is publish/subscribe messaging?

With publish/subscribe message passing the sending application/client establishes a named topic in the JMS broker/server and publishes messages to this queue. The receiving clients register (specifically, subscribe) via the broker to messages by topic; every subscriber to a topic receives each message published to that topic. There is a one-to-many relationship between the publishing client and the subscribing clients.

6) Why doesn't the JMS API provide end-to-end synchronous message delivery and notification of delivery?

Some messaging systems provide synchronous delivery to destinations as a mechanism for implementing reliable applications. Some systems provide clients with various forms of delivery notification so that the clients can detect dropped or ignored messages. This is not the model defined by the JMS API. JMS API messaging provides guaranteed delivery via the once-and-only-once delivery semantics of PERSISTENT messages. In addition, message consumers can insure reliable processing of messages by using either CLIENT_ACKNOWLEDGE mode or transacted sessions. This achieves reliable delivery with minimum synchronization and is the enterprise messaging model most vendors and developers prefer. The JMS API does not define a schema of systems messages (such as

delivery notifications). If an application requires acknowledgment of message receipt, it can define an application-level acknowledgment message.

7) What are the core JMS-related objects required for each JMS-enabled application?

Each JMS-enabled client must establish the following:

- A connection object provided by the JMS server (the message broker)
- Within a connection, one or more sessions, which provide a context for message sending and receiving
- Within a session, either a queue or topic object representing the destination (the message staging area) within the message broker
- Within a session, the appropriate sender or publisher or receiver or subscriber object (depending on whether the client is a message producer or consumer and uses a point-to-point or publish/subscribe strategy, respectively). Within a session, a message object (to send or to receive)

8) What is the Role of the JMS Provider?

The JMS provider handles security of the messages, data conversion and the client triggering. The JMS provider specifies the level of encryption and the security level of the message, the best data type for the non-JMS client.

9) How does a typical client perform the communication?

1. Use JNDI to locate administrative objects.
2. Locate a single ConnectionFactory object.
3. Locate one or more Destination objects.
4. Use the ConnectionFactory to create a JMS Connection.
5. Use the Connection to create one or more Session(s).
6. Use a Session and the Destinations to create the MessageProducers and MessageConsumers needed.
7. Perform your communication.

10) Give an example of using the point-to-point model.

The point-to-point model is used when the information is specific to a single client. For example, a client can send a message for a print out, and the server can send information back to this client after completion of the print job.

11) Does Tomcat support JMS (Java Messaging Service)?

Tomcat is just a servlet container, not an EJB container nor an application server, so it does not contains any JMS basic support. However, there's nothing stopping you from using another JMS provider.

12) Is it possible to send email messages using JMS?

JMS has no inherent support for email operations.

13)How do I communicate between two clients that are on different machines on a network using JMS? I want to use a standalone application for communicating between the machine and I want to pass the message using JMS.

You can make two JMS client applications, say AppA and AppB. Make AppA listen to topic 'forA'. Make AppB listen to topic 'forB'.
If AppA sends a message to topic 'forB', AppB will receive it. If AppB sends a message to topic 'forA', AppA will receive it.
For sample code etc, try downloading SonicMQ (as a JMS server) and go through the samples.

14)Is there any relationship between javax.jms.Message and javax.mail.Message?

There is no direct relationship between javax.mail.Message and javax.jms.Message. If your requirement is to map (correlate) them, here is what you can do:

1. From JMS domain to JavaMail domain (a javax.jms.Message is received):
    1. A JMS topic/queue can be associated with one or many e-mail id(s).
    2. The JMS Message Header can be mapped to 'custom' JavaMail Message Header.
    3. The JMS Message Body can be associated with the JavaMail Message body.
    4. A JavaMail client application should be able to process these 'custom' headers and the content of the message body.
2. From JavaMail domain to JMS domain (a javax.mail.Message is received):
    1. An e-mail id can be associated with one or more JMS topics/queues.
    2. The JavaMail Message Header can be mapped to 'custom' JMS Message Header.
    3. The JavaMail Message Body can be associated with the JMS Message body.
    4. The JMS client application should be able to process these 'custom' headers and the content of the message body.

In a simple application that I tried, I removed the 'custom' header scenario and just forwarded the contents of the message (text message), which worked without any problems.Try using SonicMQ bridges, which already has something like that.

15) Is it possible to acknowledge individual messages on a queue without affecting previously received, but as yet unacknowledged, messages?

If you acknowledge a message, all previously received messages will also be acknowledged. From the javax.jms.Message Javadoc, the acknowledge method will "Acknowledge this and all previous messages received."
So the answer to your question is no, if what you meant by "affecting" is not-yet acknowledged.

I suggest an alternative. You should look at javax.jms.QueueBrowser to review queued messages. QueueBrowser has getEnumeration, which "Gets an enumeration for browsing the current queue messages in the order they would be received".

16) What encryption options are there for sending messages through JMS?

Encryption is not handled by the JMS specification. It is left to the JMS provider to implement and provide encryption and decryption of messages. These days, Progress Software's SonicMQ is a leading JMS provider and they have a robust encryption mechanism called Quality of Protection. They also provide an SSL-related feature, which also has build in encryption.

17) How does the Application server handle the JMS Connection?

1. Application server creates the server session and stores them in a pool.
2. Connection consumer uses the server session to put messages in the session of the JMS.
3. Server session is the one that spawns the JMS session.
4. Applications written by Application programmers creates the message listener.