Change History 11

11.1 Version 1.0.1

11.1.1 JMS Exceptions

A new JMS Exception chapter was added and it contains the following new information:

- Two fields were added to *JMSException* - a vendor error code and an Exception reference.
- In version 1.0, *JMSException* was the only JMS exception specified. Version 1.0.1 adds a list of standard exceptions derived from JMSException and describes when each should be thrown by JMS providers.

11.2 Version 1.0.2

The objective of JMS 1.0.2 is to correct errata in the JMS 1.0.1 specification and code that have been uncovered by implementors and users. It also contains some clarifications that resolve ambiguities found in the previous versions.

In a few edge cases, JMS implementors found that the JMS 1.0.1 specification caused problems that required small adjustments in semantics. JMS 1.0.2 includes these changes so that it correctly reflects the consensus reached by JMS partners on these issues.

11.2.1 Connection and Session Close

JMS 1.01 did not fully specify the sequence for closing a connection and its sessions. This sequence is now fully specified.

JMS 1.01 implied that calls to connection and session close returned immediately. This made it difficult for a client to properly sequence the close process. Connection and session close now explicitly state that they block until close completes.

11.2.2 Creating a Session on an Active Connection

More information about creating a session on an active (as opposed to stopped) connection has been added.

11.2.3 A Multiple Topic Subscriber Special Case

JMS 1.0.1 specified that in the special case of two topic subscribers on a session with overlapping subscriptions, a message that was selected by both would only be delivered to one. Implementation experience revealed that this case was better handled in the same way that overlapping subscriptions from different sessions are treated, so this special case has been deleted.

11.2.4 Delivery Mode and Message Retention

The effect delivery mode has on message retention has been clarified.

11.2.5 The ‘single thread’ Use of Sessions

Sessions are designed to minimize the need to write for multi-threaded code in order to support the asynchronous consumption of messages. Clarification on the benefits and the programming model of this design have been added.

11.2.6 Clearing a Message’s Properties and Body

A clarification that explicitly notes that clearing a message’s properties and clearing its body are independent.

11.2.7 Message Selector Comparison of Exact and Inexact Numeric Values

JMS 1.0.1 specified that message selectors did not support the comparison of exact and inexact numeric values. This conflicted with the requirement to support numeric promotion. This has been changed to support exact and inexact comparison.

11.2.8 Message Selector Numeric Literal Syntax

A note has been added that states that the numeric literal syntax is that specified by the Java language.

11.2.9 Comparison of Boolean Values in Message Selectors

A note has been added that only equality and inequality comparisons are supported.

11.2.10 Order of Messages Read from a Queue

A note has been added that explains that a client can read messages from a destination in an order different from the order they have been sent by using a selector that matches a later message and then using a selector that matches an earlier message.

11.2.11 Null Values in Messages

A note has been added that message values are allowed to be null.

11.2.12 JMS Source Errata

Two methods required by the spec were left out of the source, the getJMSXPropertyNames method of ConnectionMetaData and the getExceptionListener method of Connection. These have been added.

The type of the time-to-live parameter of setTimeToLive and getTimeToLive methods of MessageProducer and the type of the default time-to-live constant were int and have been changed to long.

The close sequence of TopicRequestor and QueueRequestor did not agree with the order specified in the specification and this has been corrected.

The type of the parameter of the createTextMessage method that takes an input value was changed from StringBuffer to String.

11.2.13 Changing a Durable Subscription

If a durable subscription is created with the same name as an existing one this is equivalent to unsubscribing (deleting) the old one and creating the new one.

11.2.14 Closing Constituents of Closed Connections and Sessions

There was some ambiguity about whether or not close needed to be called on all JMS objects. A note has been added that states that there is no need to close the sessions of a closed connection; and there is no need to close the producers and consumers of a closed session.

11.2.15 The Termination of a Pending Receive on Close

JMS 1.0.1 did not describe how a pending message receive is terminated if its session or connection is closed. It is now specified that in this case receive returns a null message.

11.2.16 Incorrect Entry in Stream and Map Message Conversion Table

This table erroneously included a required conversion between char and String. This has been removed.

11.2.17 Inactive Durable Subscription

A note explaining that an inactive durable subscription is one that exists but does not at the time have a TopicSubscriber created for it.

11.2.18 JMS Source JavaDoc Errata

The correct end-of-message indicator for the readBytes method of BytesMessage is a return value of -1.

The setPriority method of MessageProducer should have a parameter named ‘priority’ not ‘deliveryMode’.

11.2.19 JMS Source JavaDoc Clarifications

Note that byte values are returned as byte[] not Byte[] by the readObject method of StreamMessage and the getObject method of MapMessage.

Note that the acknowledge method of Message acknowledges all messages received on that message's session.

Note that the InvalidClientIDException is used for any client id value that a JMS provider considers invalid. Since client id value is JMS provider specific the criteria for determining a valid value is provider specific.

A note has been added to the readBytes method of StreamMessage and BytesMessage to describe how values that overflow the size of the input buffer are handled.

A note has been added that clarifies when setClientID method of Connection should be used.

Note that calling the setMessageListener method of MessageConsumer with a null value is equivalent to calling unsetMessageListener.

Note that the unsubscribe method of TopicSession should not be called to delete a durable subscription if there is a TopicConsumer currently consuming it.

Note that result of calling the setMessageListener method of MessageConsumer while messages are being consumed by an existing listener or the consumer is being used to synchronously consume messages is undefined.

Note the createTopic method of TopicSession and the createQueue method of QueueSession are used for converting a JMS provider specific name into a Topic or Queue object that represents an existing topic or queue by that name. These methods are not for creating the physical topic of queue. The physical creation of topics and queues are administrative tasks and are not done by JMS. The one exception is the creation of temporary topics and queues which is done using the createTemporaryTopic and createTemporaryQueue methods.

Note that the setObject method of ObjectMessage places a copy of the input object in the messages.

Note that a connection is created in stopped mode and, for incoming messages to be delivered to the message listeners of its sessions, its start method must be called.

Documentation of Message default constants has been added.

Note that the readBytes method of StreamMessage and BytesMessage returns -1 with an empty byte array if it reads a null value.