IBM MQ V8 and JMS 2.0
An Introduction

Matthew Whitehead
WebSphere MQ Development
mwhitehead@uk.ibm.com

Please note

IBM’s statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM’s sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user’s job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Trademarks

- IBM and the IBM logo are trademarks of International Business Machines Corporation, registered in many jurisdictions. Other marks may be trademarks or registered trademarks of their respective owners.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
- Other company, product and service names may be trademarks, registered marks or service marks of their respective owners.
- References in this publication to IBM products and services do not imply that IBM intends to make them available in all countries in which IBM operates.

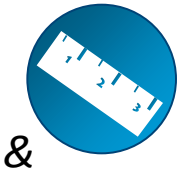
Agenda

Intro & MQ V8

JMS 2.0

- Simplified API
- Messaging Features
- JavaEE
- Updates

IBM MQ V8 – Best in class enterprise messaging



Platforms & Standards

- 64-bit for all platforms
- Support for JMS 2.0
- Improved support for .Net and WCF
- Changes to runmqsc
- SHA-2 for z, i & NSS

Security

- Userid authentication via OS & LDAP
- User-based authorisation for Unix
- AMS for IBM i & z/OS
- DNS Hostnames in CHLAUTH records
- Multiple certificates per queue manager

Scalability

- Multiplexed client performance
- Queue manager vertical scaling
- Publish/Subscribe improvements
- Routed publish/subscribe
- Multiple Cluster Transmit Queue on all platforms

System z exploitation

- 64-bit buffer pools in MQ for z/OS means less paging, more performance
- Performance and capacity
- Performance enhancements for IBM Information Replicator (QRep)
- Exploit zEDC compression accelerator
- SMF and shared queue enhancements

IBM MQ V8 – Summary of changes

- New version of the Java Runtime – version 7
- Packaging Changes
  - Fewer JARs
  - Simpler and quicker access to the just the JARs
- Removal of DirectIP function
- Username and Password updates
- Improved Control of Tracing

Java 7

JMS 2.0 – prereqs use of java.lang.AutoCloseable.

Interfaces in jms.jar built using Java 7 class file format

JRE now shipped with MQ v8 is IBM (or hybrid) Java 7

IBM Java 7.1 is coming with additional platform support

Features:

- http://radar.oreilly.com/2011/09/java7-features.html
- Try with resources specifically as JMS objects are 'resources' now
- Try with multiple catch block of specific interest for coding exception handling
- NOT adopted the new io classes

```
c:\Program Files\IBM\WebSphere MQ_2\java\jre\bin>.\java -version
java version "1.7.0"
Java(TM) SE Runtime Environment (build pwa6470sr6-20131015_01(SR6))
IBM J9 VM (build 2.6, JRE 1.7.0 Windows 7 amd64-64 Compressed References 20131013_170512
J9VM - R26_Java726_SR6_20131013_1510_B170512
JIT  - r11.b05_20131003_47443
GC   - R26_Java726_SR6_20131013_1510_B170512_CMPRSS
J9CL - 20131013_170512)
JCL - 20131011_01 based on Oracle 7u45-b18
```

Packaging - DirectIP Removal

Ability to create a DirectIP (TCP or HTTP) connection has been removed.

Function not supported exception thrown

```
-------------HANDLING EXCEPTION-----------------------------
      Message : JMSFMQ1006: The value 'DirectIP' for property 'Transport Type' is not valid.
        Class : class com.ibm.msg.client.jms.DetailedJMSException
     Identity : ff7f61a6
  Explanation : The value specified for the property is not supported.
  User Action : Modify the value to be within the range of accepted values.
   Error Code : JMSFMQ1006
  Suppressing : nothing
        Stack : sun.reflect.NativeConstructorAccessorImpl.newInstance0(NativeConstructorAccessorImpl.java:-2)
              : sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:80)
              : sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:57)
              : java.lang.reflect.Constructor.newInstance(Constructor.java:539)
              : com.ibm.msg.client.commonservices.j2se.NLSServices.createException(NLSServices.java:311)
              : com.ibm.msg.client.commonservices.nls.NLSServices.createException(NLSServices.java:225)
              : com.ibm.msg.client.wmq.factories.WMQConnectionFactory.createProviderConnection(WMQConnectionFactory.java:6080)
```

APIs controlling properties retain settings – specifically so administration of objects is possible via JMSAdmin or Explorer

No Normal Support of a Message Broker that supports this is available

Packaging – JAR files

Files that have gone:

- ldap.jar
- jndi.jar
  - *Both now included in the standard class libaries*
- connector.jar
  - *Outdated dependency in Base Java – not required*
- CL3Export.jar
- CL3NonExport.jar
- dhbcore.jar
  - *DirectIP support - removed*

Files that are new

- com.ibm.mq.allclient.jar
  - *Base Java, JMS, Headers and PCF classes*
- com.ibm.mq.traceControl.jar
  - *Remote trace control*

Changes

- jms.jar is the new jms.jar – no distinguishing features except the size – now 57kb (was aprox 23kb)

Security – Using MQ V8 CONNAUTH

Motivation: Start to properly use username and password for authentication in Client mode

Compatibility: We haven't historically done this, so we make it optional

Enabling the option

- Globally – configuration option Configuration.USE_MQCSP_AUTHENTICATION (boolean, Y/N)
  - Environment Variable *com.ibm.mq.jmqi.useMQCSPauthentication*
  - System property *com.ibm.mq.cfg.jmqi.useMQCSPauthentication*
  - Client ini file stanza *JMQI*, attribute *useMQCSPauthentication*

Effect if option is set

- Flow the username and password data in the MQCSP as part of the CONAUTH flow
- Flow the “real” username in the UID flow

This may affect your security exits

Security – Using MQ V8 Password Protection

Configuration.PASSWORD_PROTECTION

- Choices - "Compatible", "Always", "Optional"
- Environment Variable *com.ibm.mq.jmqi.PasswordProtection*
- System Property *com.ibm.mq.cfg.jmqi.PasswordProtection*
- Client ini file Stanza *Channels*, attribute *PasswordProtection*
- Only “Always” is really relevant to Clients
  - If set, then the list offered to the QM only allows for 3DES
  - If unset, the list allows for 3DES and NULL encryption

Configuration AMQ_RANDOM_NUMBER_TYPE

- Choices - “Standard", "Fast"
- Environment Variable *AMQ_RANDOM_NUMBER_TYPE*
- System Property *com.ibm.mq.cfg.jmqi.AmqRandomNumberType*
- Standard – use java.security.SecureRandom
- Fast – use java.util.Random

This is only relevant if the previous “flow userid and password in the CONAUTH flow” options are in force, otherwise the userid and password is in the ID flow which will not be encrypted.

Dynamic Trace Control

```
C:>java -jar MQ_INSTALL/java/lib/com.ibm.mq.traceControl.jar -list
10008 : 'MQSample'
 9004 : 'MQ_INSTALL/java/lib/com.ibm.mq.traceControl.jar -list'


C:>java -jar MQ_INSTALL/java/lib/com.ibm.mq.traceControl.jar -i 10008 -status
Tracing enabled : false
User Directory : C:\Users\IBM_ADMIN\RTCworkspace\sandpit
Trace File Name : mqjms.trc
Package Include/Exclude tree
root - Included


C:>java -jar MQ_INSTALL/java/lib/com.ibm.mq.traceControl.jar -i 10008 -enable
Enabling trace
Tracing enabled : true


C:>java -jar MQ_INSTALL/java/lib/com.ibm.mq.traceControl.jar -i 10008 -status
Tracing enabled : true
User Directory : C:\Users\IBM_ADMIN\RTCworkspace\sandpit
Trace File Name : mqjms_10008.trc
Package Include/Exclude tree
root - Included
```

Agenda

Intro & MQ V8

JMS 2.0

- Simplified API
- Messaging Features
- JavaEE
- Updates

JMS 2.0 Headlines – What's new?

- ***Specification updates and clarifications***
  - Point to Point *AND* Publish/Subscribe domains required
- ***API Changes***
  - *Use of Java7's java.lang.AutoCloseable*
  - *JMS Simplified API*
  - *Session doesn't need parameters (for JavaEE)*
- ***New Messaging Features***
  - Delivery Delay
  - Asynchronous Send
  - Subscriptions can be shared across a messaging provider
- ***JavaEE Updates***
  - *Recommendation* on provision of a Resource Adapter
  - Specification clarifications
  - Part of Java EE 7

Agenda

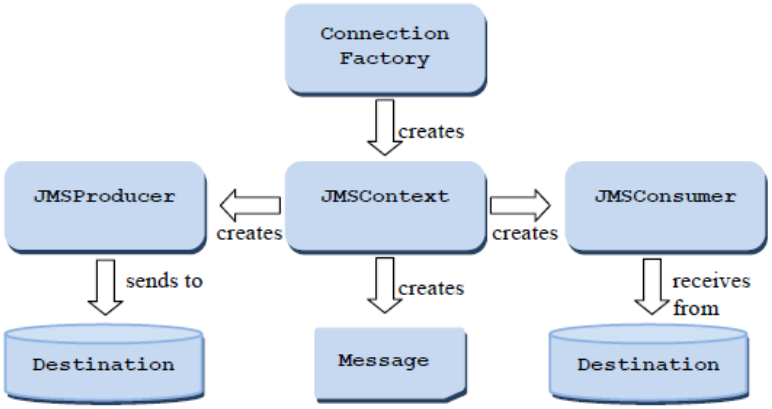
Intro & MQ V8

JMS 2.0

- **Simplified API**
- Messaging Features
- JavaEE
- Updates

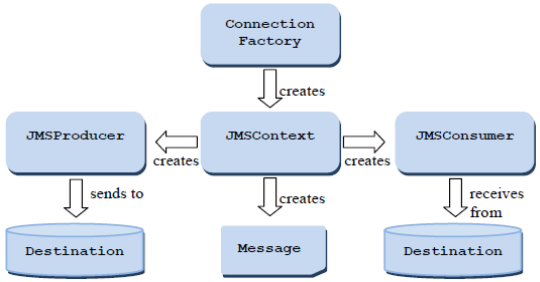
Simplified API

- **ConnectionFactory**
  an administered object to create a Connection. As used by the classic API.
- **JMSContext**
  an active connection to a JMS provider and a single-threaded context for sending and receiving messages
- **JMSProducer**
  created by a JMSContext, used for sending messages to a queue or topic
- **JMSConsumer**
  created by a JMSContext, used for receiving messages sent to a queue or topic



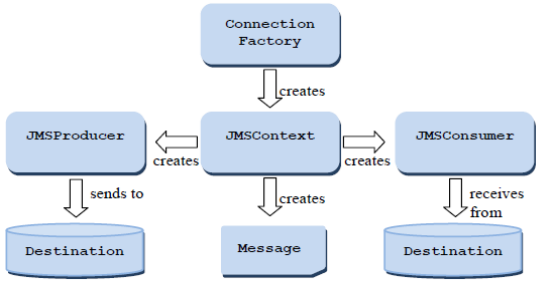
All based on JMS1.1
Unified Domain concepts

JMSContext Features



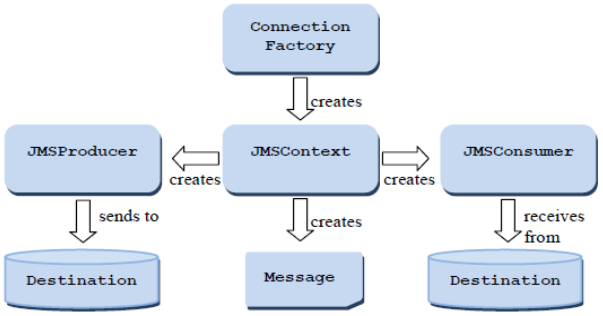
- Retains the same semantics as JMS 1.1 Connection & Session, e.g. Temporary destinations for JMSContext scoped by concept of underlying connection
- AutoStart of underlying connection – i.e. can forget to forget starting the connection
- Can be application managed – from `createContext()` on ConnectionFactory, or...
- can be container managed using @Inject annotation

JMSContext Features



- **JMSContext = 2 HCONNs**
  - 1 for the JMS connection
  - 1 for the JMS session
- **Can create 'child' contexts, i.e.**
  - jmsContext.getContext();
  - Each child context creates 1 more HCONN

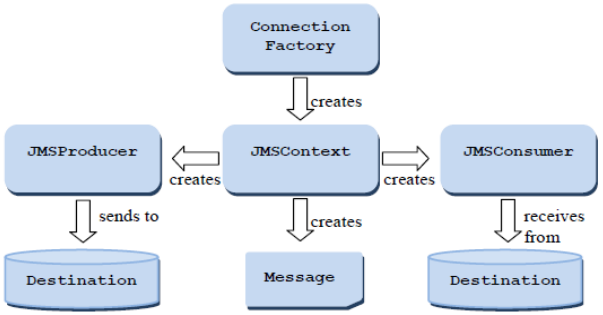
Producing Messages



- **JMSProducer**
  - Can take on role of being 'proxy' message object.
  - Message properties set on the producer object prior to sending a 'body'
  - Existing MessageProducers can't do that; though have been extended for new messaging styles
  - Method chaining
  - 'lightweight object' therefore no close

```
11      producer.setProperty("MyProperty", "JMS2.0").send(destination, "SimplePTP: ");
12
13      context.createProducer().setTimeToLive(1000).setDeliveryMode(NON_PERSISTENT).send(dataQueue, body);
```

Messages



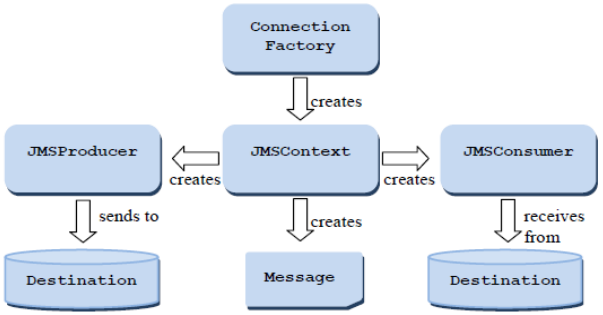
- **Message Body only**
  - JMSProducer & JMSConsumers can now work with the message bodies without message objects
  - Messages have a `getBody()` method
- **After sending message, application free to modify message**
- **JMSXDeliveryCount is now mandatory [§3.5.11]**
  - Doesn't have to be exactly correct
    - i.e. no persistence of value required
  - If JMSRedelivered=true, then JMSXDeliveryCount>=2
  - MQ JMS has always set this

Message Body Conversions

```
15      Message receivedMessage = consumer.receive(15000);
16      System.out.println("Received message:\n" + receivedMessage.getBody(String.class));
17      System.out.println(receivedMessage.getStringProperty("MyProperty"));
18
19      System.out.println("Message Body"+consumer.receiveBody(String.class));
```

MessageType	Parameter
TextMessage	String.class
ObjectMessage	java.io.Serializable
MapMessage	java.util.Map or java.util.Object
BytesMessage	byte[].class
Message	Always returns null

Consuming Messages



- **JMSConsumer**
  - `receiveBody(...)` methods - message bodies only.
  - Not completely symmetrical with sending
    - i.e. if want properties still need a message object
- **Async and Sync consumption as before**
- **Can closed from another thread**
  - Waits on in-progress consumption to finish

Receiving Message Bodies Synchronously

- `<T> T receiveBody( Class<T> c)`
  - Receives the next message produced for this JMSConsumer and returns its body as an object of the specified type
- `<T> T receiveBody( Class<T> c, long timeout)`
  - Receives the next message produced for this JMSConsumer that arrives within the specified timeout period, and returns its body as an object of the specified type
- `<T> T receiveBodyNoWait( Class<T> c)`
  - Receives the next message produced for this JMSConsumer if one is immediately available and returns its body as an object of the specified type
- **Can throw a MessageFormatRuntimeException if wrong class used**
  - If AUTO_ACK, will be as if method never called

  ..else will be as if method failed, app must force rollback

Message Body Conversions: MQ V8 Notes

- Applications needs to be careful how these are used, especially with Message Consumers

- Consumer's using `receiveBody` if AUTO_ACK or DUPS_OK needs lock message on queue so unavailable to anybody else
  - Message is locked on the queue using GMO_LOCK
  - Body conversion is attempted at the client
  - Success causes message to be destructively removed from the queue
  - Failure means message is unlocked
    - This **will not** increment JMSXDeliveryCount

- Transactional `receiveBody` – gets message and then examines the message body
  - Failure of this is returned to application
  - Application expected to rollback transaction
  - This **will** increment JMSXDeliveryCount

JMS 1.1 Interfaces Updates

- **Connections**
  - New methods for creating session with no transaction arguments
  - Shared connection consumers
- **Sessions**
  - Creating share consumers
  - Creating JMS1.1 MessageConsumer for durable subscriptions
- **Message Producer**
  - New set/get for Delivery Delay
  - Send methods extended to supply Completion Listener
- **Message Consumer**
  - No change
- **Message**
  - For delivery delay – `getJMSDeliveryTime()`
  - New methods to directly access body data, `getBody()` `isBodyAssignableTo()`

Java 7 Auto-Closeable

- **Java 7 gives large variety of**
  - –new functions – class libraries
  - –Java Language enhancements
  - –New class file version
- **JMS2.0 drives Java 7 specifically because of the try-with-resources**

```
2     ConnectionFactory cf = createConnectionFactory();
3     try {
4       // Create JMS objects
5       connection = cf.createConnection();
6     } catch (JMSException jmsex) {
7       recordFailure(jmsex);
8     } finally {
9       if (connection != null) {
10        try {
11          connection.close();
12        } catch (JMSException jmsex) {
13          System.out.println("Connection could not be closed.");
14          recordFailure(jmsex);
15        }
16      }
17    }
```

```
2     ConnectionFactory cf = createConnectionFactory();
3
4     try (JMSContext jmsctx = cf.createContext();) {
5       producer = jmsctx.createProducer();
6     } catch (Exception jmsex) {
7       recordFailure(jmsex);
8     }
```

Exceptions and Exception Listeners

- **New exception `JMSRuntimeException` thrown from new API calls**
- **JavaDoc shows mandatory cases**
  - –...but... provider may thrown these for other cases as well
- **Exceptions thrown on a JMS calls, must not be delivered to an ExceptionListener**
- **Compiler will not force you to do error checking**
- **Must take responsibility for doing it at suitable points**

Agenda

Intro & MQ V8

JMS 2.0

- Simplified API
- **Messaging Features**
- JavaEE
- Updates

New Messages Features - Recap

- **Asynchronous Send**
  - Application sends messages via API that returns before server has processed messages
  - Confirmation via listener
- **Delayed Delivery**
  - Allow delivery at a later point in time
  - (remember this is not a database)
- **Shared Subscriptions**
  - Subscription which can opened by multiple consumers
  - Messages shared amongst consumers (no fairness rules provided)

Shared Subscriptions

- **Shared non-durable subscription**
  - Identified by 'sharedSubscriptionName' and 'clientId' if set
  - If clientId is set, all consumers must share the same clientId
  - Subscription/undelivered messages deleted when last consumer is closed
- **Shared Durable Subscriptions**
  - Has the features of a regular durable subscription but pulls in multiple consumers aspect
  - ClientId is optional
- **Same 'sharedSubscriptionName' can be used for durable and non-durable**
- **Shared subscriptions are scoped at the queue manager**

Asynchronous Send

- ***Asynchronous Send* means**
  - –Send message and return from the send call before response from server
- **Closing must wait for failure or completion of async sends**
- **Callbacks made in the same order the messages where sent**
- **`CompletionListener` called when response has been received.**
  - –Does not work like message listener in terms of thread of control
- **Quality of Service**
  - –After the `onCompletion` callback it's as if the application called the synchronous send(...) method

Message Delivery Delay

- **Earliest Time JMS provider may give message to consumer**
- **Sets the minimum length of time (in ms) that must elapse after a message is sent before the JMS provider may deliver the message to a consumer**
- **For transacted sends, this time starts when the client sends the message, not when the transaction is committed.**
- **Delivery Delay set longer than expiry is an error!**
  - Will throw an exception

Message Delivery Delay – MQ V8 Notes

- **Implemented using a single internal staging queue on the Qmgr**
- **Header added to messages placed on this queue**
- **Qmgr component 'delayed delivery processor' monitors**
  - Delivery performed when delay completes
- **Only available for use by JMS**
- **Queue is created by default on Distributed, needs creating manually on zOS**

Agenda

Intro & MQ V8

JMS 2.0

- Simplified API
- Messaging Features
- **JavaEE**
- Updates

JMS and Java EE

- JMS one of the constituent specifications of JavaEE
- JavaEE 7 brings in JMS 2.0
- Context APIs also include XA variants for global transactions managed by the container
- MDB Activation Specifications main way of driving messages into JavaEE
- JMS 2.0 *strongly recommends* that a provider offer JCA Resource Adapter, so we do!

JMS and Java EE – API Updates

- **Prescriptive list of APIs that can not be called in the containers**
  - No message listeners
  - Single session / connection
  - No connection consumers
  - No Asynchronous Send
  - Transaction control etc.
- **Session/Context APIs - No local transactions or client acknowledgement**
  - `javax.jms.Connection.createSession()`
  - `javax.jms.ConnectionFactory.createContext()`

Resource Adapter

- **Two additional properties for an activation specification**

- ***destinationLookup***
  - the JNDI name of javax.jms.Queue or javax.jms.Topic that defines the JMS queue or topic for the MDB

- ***connectionFactoryLookup***
  - the JNDI name of javax.jms.ConnectionFactory, used to connect to the JMS provider

Resource Adapter – MQ V8

- **Stand alone applications use JMS 2.0 JARs supplied with MQ**
- **JavaEE servers use the resource adapter**
- **The app server has to have support for JavaEE 7 or JCA 1.7**
  - WebSphere App Server doesn't have this yet
  - WebSphere Liberty Profile is in beta with JMS 2.0 support
- **WMQ v8 Resource Adapter only deployed in JavaEE7**
  - Can't deploy V8 resource adapter into WAS 8.5.5 or earlier
- **WMQ V7.* Resource Adapters can be used to connect to WMQ V8 queue manager in client or bindings modes**

Agenda

Intro & MQ V8

JMS 2.0

- Simplified API
- Messaging Features
- JavaEE
- **Updates**

Migration of Applications

- **JMS 2.0 supports all the JMS1.1 APIs**
  - The JMS2.0 version of the specification though defines the JMS1.1 usage
- **Existing applications, with recompile, can run with a JMS2.0 implementation**
- **JMS1.1 apps though written against the new JMS2.0 implementation will not work in a JMS 1.1 only environment (e.g. WebSphere V8)**
  - Existing interfaces reference some new JMS2.0 interfaces
- **Remember that the JVM for JMS2.0 implementations and Java7 runtime is needed**

JMS and the Compliance Test Suite

- **To ensure that the JMS provider has implemented the specification correctly there is a new compliance test suite that has to be run.**
- **To pass the JMS 2.0 compliance tests we have had to change a few default behaviours (see next slide)**
- **'out of the box' MQ JMS needs to comply but old behavior can be switched back on**
  - the property `com.ibm.mq.jms.SupportMQExtensions`, which can be set to TRUE, to revert these changed behaviors back to previous implementations.

supportMQExtensions=true

- Message priority
  - Messages can be assigned a priority, 0 - 9. Before JMS 2.0, messages could also use the value -1, indicating that a queue's default priority is used. JMS 2.0 does not allow a message priority of -1 to be set. Turning on SupportMQExtensions allows the value of -1 to be used.

- Client id
  - The JMS 2.0 specification requires that non-null client ids are checked for uniqueness when they make a connection. Turning on SupportMQExtensions, means that this requirement is disregarded, and that a client id can be reused.

- NoLocal
  - The JMS 2.0 specification requires that when this constant is turned on, a consumer cannot receive messages that are published by the same client id. Before JMS 2.0, this attribute was set on a subscriber to prevent it receiving messages that are published by its own connection. Turning on SupportMQExtensions reverts this behaviour to its previous implementation.

Provider Version

- **We've introduced a new property for the ProviderVersion field**
  - This is for WebSphere MQ 'Normal Mode'.
    - This is represented by a PROVIDERVERSION=8.
    - This means that we need to connect to a QM that has a cmd level of 800 (WMQ v8). When this is achieved you can use all the JMS 2.0 features.
  - Connecting via 'Normal Mode with Restrictions'
    - This is how PROVIDERVERSION=7 is defined
    - This can connect to a queue manager with cmd level greater than 700
    - You can use the JMS2.0 API but not Aysnc Send, Delayed Delivery or Shared Subs.
  - Connecting via the 'Migration mode' means that everything is JMS1.1
    - No matter the QM connected to
- **Leaving PROVIDERVERSION unset means**
  - 'Normal Mode' is attempted first
  - If the command level is less than 800 then 'Normal Mode with Restrictions' is used (established hConn is re-used)
  - If the command level is less than 700, the connection is closed and recreated with 'Migration Mode'

Error codes to watch out for

Error Code	Message
JMSCC5007	Use of the JMS2.0 API "{0}" is not supported with this instance of this connection
	Only connections with a correct type of connection can support using the JMS2.0 API
JMSCC5008	Use of the JMS2.0 Function "{0}" is not supported with this instance of this connection
	The use of the JMS2.0 functionality mentioned in the message is only supported when connecting to a WebSphere MQ V8 queue manager using WebSphere MQ messsaging provider V8 mode

WebSphere Liberty – Java EE 7 Certified

- In June WebSphere Liberty profile announced Java EE 7 support and certification
- WebSphere Liberty 8.5.5.6+ now supports Java EE 7
- First production-supported application server to be Java EE 7 certified

Thank You – Questions?

Legal Disclaimer

- © IBM Corporation 2014. All Rights Reserved.
- The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.
- References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.