Using webLogic web server and the IBM MQ resource adapter.

The IBM MQ resource adapter is the latest way of deploying Message Driven Beans using IBM MQ in a web server. It supports advanced techniques like connection pooling which allows an MQ connection handle to be reused, and avoids the overhead of MQDISC and MQCONN when a message is processed (and so could save you millions of MQCONNs and MQDISCs a day).

Before the Resource Adapter (RA) was available configuration information was stored in an external JDNI server (and so could be shared by other web servers or batch applications).

The IBM MQ RA can be used in all major web servers that support JCA 1.5 (Java Connector Architecture), but each web server has its own way of specifying configuration. The use of IBM MQ RA in IBM WebSphere is well documented. The use of IBM MQ RA in webLogic is not well documented, and this document tries to fill some of the holes.

There are many steps to configuring the resource adapter. I had to do some steps – but they made no sense, but without these steps it didnt work! It is also seems more complex because of webLogic using confusing terminology. It uses the term connection pool when it means connection, as once you have defined a connection you then configure the connection pool.

IBM provides a wmqra.rar file which can be used as a basis for your webLogic resource adapter.
It needs to have webLogic specific definitions added to it. Once you have installed the resource adapter, you can use the Admin Console to define additional resources – such as additional connectionFactories.

Create the weblogic-ra.xml file.

For webLogic, the .rar file needs a weblogic-ra.xml file. See here for an example

The structure of this can be seen from the weblogic-ra.xsd file see here or the xsd itself. See the xsi definition at the top of the file.

Key configuration

From playing with different configurations I have found the following key configuration information. Note: It may not be 100% correct.

- <jndi-name>wmq.jmsra</jndi-name> - the specified name is used by MDBs to link up with the required RA.

- You need one and only one <connection-factory-interface> **javax.jms.ConnectionFactory**… definition to define a connection for an MDB and the applications.
- If I tried to define more than one connectionFactory, it had errors when it was deployed.
- You need one and only one <connection-factory-interface> **javax.jms.QueueConnectionFactory** … definition. Without a QueueConnectionFactory, you do not get connection pooling – and every message put, has an MQCONN… and an MQDISC. However it looks as if it is configured as a ConnectionFactory, not a QueueConnectionFactory. Strange but true.
- If you are using topics you need one <connection-factory-interface> javax.jms.ConnectionFactory…
- You can provide default configuration information at the high level.
  - You can provide configuration information for a connection factory. If a parameter is not specified, it will take a value from the high level defaults, if specified.
- For a connection pool, typical defaults include
  - applicationName a_comment
  - hostName 127.0.0.1
  - channel SYSTEM.DEF.SVRCONN
  - port 1414
  - failIfQuiesce true
  - queueManager QMA
- you specify properties using tags <properties> <property> <name>...</name><value>…. </value> <property>… </property></properties>

I suggest you use the weblogic-ra.xml file I provide and get an MDB working, and then enhance it.
If the .rar file has been changed, you have to redeploy it. I found it easier to restart the webLogic instance after I changed the .rar file, otherwise I got strange error messages.

Create your resource adapter.

You should copy the IBM provided MQ .rar file and extend it. See instructions here.
Create a directory for your resource adapter
- *mkdir myresourceadapter.*
- *cd myresourceadapter*
- *mkdir META-INF*

Create or or copy the weblogic-ra.xml file to META-INF/weblogic-ra.xml
copy the IBM supplied resource adapter to the current directory
- *cp /opt/mqm/java/lib/jca/wmq.jmsra.rar webLogic.rar*

create your .rar file by adding the weblogic specific file to the jar file
- *jar -uvf webLogic.rar META-INF/weblogic-ra.xml.*

Deploy your resource adapter.

Go to the weblogic admin console and click Deployments.
Click install.
In the path: enter the fully qualified path to the directory containing your .rar file
Click next.
Select “Install this deployment as an application”, click next.
Click finish.
It takes a few seconds to deploy.

If you get a web page called *Save Deployment Plan*, give the fully qualified name of a path. If you deploy multiple webLogic instances you may want to specify a directory specific to the instance. This deployment plan contains the changes you make to the configuration. You should make sure you back it up. If you reuse it, this can cause an error message

Error is: "weblogic.application.ModuleException: java.lang.ClassCastException: com.ibm.mq.connector.DefaultRuntimeHelperImpl cannot be cast to com.ibm.mq.connector.JCARuntimeHelper"
Shut down and restart the instance.

Check the webLogic server log, correct any problem and repeat the install, Once the resource adapter has been installed you will need to use the “update button” process, described below.

Select the box in front of the resource adapter
Click update
Select *Update this application in place with new deployment plan changes*
Click finish
Check the log to ensure it has deployed successfully.
If you get
Error is: "weblogic.application.ModuleException: java.lang.ClassCastException: com.ibm.mq.connector.DefaultRuntimeHelperImpl cannot be cast to com.ibm.mq.connector.JCARuntimeHelper"
Shut down and restart the instance.

It is worth using
$ *java weblogic.appmerge -verbose ~/myrar/weblogic.rar 1>aa 2>bb* to check the syntax and consistency of the file before deploying it.

Post install check

Go to the home of the admin console page. On the left hand panel is *+Environment*. Click to expand this.
Click servers.
Click on your admin server in the list.
Above the list is View JNDI Tree click on this link.
You should see the connection factory from your .rar in this list. If not, check the deployment in the weblogic server log, and restart the web server instance.

You have deployed a resource adapter!

Configure the resource adapter.

Go the the home page of the admin console, and click Deployments.   This lists the applications you have installed.
Click on your resource adapter.
Click on the configuration tab (at the top of the page)
Click on the Outbound Configuration Pool tab.
Expand the +javax.jms.ConnectionFactory.
You should see the connectionFactory from the .rar file.
Click New
Select javax.jms.ConnectionFactory
Click next
Fill in a JNDI name
Click finish
Expand the +java.jmx.ConnectionFactory
Click on the entry you just added
You can now change the parameters, change a value and press enter.
Click save.
Click Next to go to the next page etc..

The default values should be those specified in the .rar file
If you display the JNDI tree you should see your connectionFactories in the list. Go for a beverage of your choice to celebrate this major milestone

Configuring an MDB

Edit the ejb-jar.xml file for the MDB (in the META-INF directory) add the activation-config information. For example

```
<enterprise-beans>
  <message-driven>
          ….
    <message-destination-type>javax.jms.Queue</message-destination-type>
       <activation-config>
        <activation-config-property>
          <activation-config-property-name>connectionfactory
               </activation-config-property-name>
          <activation-config-property-value>CF3</activation-config-property-value>
        </activation-config-property>

       <activation-config-property>
          <activation-config-property-name>destination</activation-config-property-
name>
          <activation-config-property-value>JMSQ2</activation-config-property-
value>
       </activation-config-property>
```

```
        **<activation-config-property>
           <activation-config-property-name>destinationType
               </activation-config-property-name>
           <activation-config-property-value>javax.jms.Queue
              </activation-config-property-value>
         </activation-config-property>
        </activation-config>**
      </message-driven>
</enterprise-beans>
```

The bold text is new, and defines the parameters connection factory CF3 and queue JMSQ2.

Rather than use the verbose
<property ><name>**destination**</name><value> **JMSQ2** </value> <property> syntax of the weblogic-ra.xml, there is the even more verbose

```
**<activation-config-property>
     <activation-config-property-name>connectionfactory</activation-config-property-name>
      <activation-config-property-value>CF3</activation-config-property-value>
</activation-config-property>**
```

This is where cut and paste is your friend.

The destination (JMSQ2) is the real queue name. (If you are not using the resource adapter, this tends to be the name of a JNDI object which points to the queue name).

The weblogic-jar.xml file has

```
<?xml version="1.0"?>
<!DOCTYPE weblogic-ejb-jar
 PUBLIC '-//BEA Systems, Inc.//DTD WebLogic 8.1.0 EJB//EN'
 'http://www.bea.com/servers/wls810/dtd/weblogic-ejb-jar.dtd'>

<weblogic-ejb-jar>
  <weblogic-enterprise-bean>
   <ejb-name>WMQ_IVT_MDB</ejb-name>
   <message-driven-descriptor>
**<resource-adapter-jndi-name>wmq.jmsra</resource-adapter-jndi-name>**
~~     <destination-jndi-name>JMSQ1</destination-jndi-name>~~
~~     <initial-context-factory>com.sun.jndi.fscontext.RefFSContextFactory</initial-context-factory>~~
~~     <provider-url>file:/home/colinpaice/JNDI-Directory</provider-url>~~
~~     <connection-factory-jndi-name>CF1</connection-factory-jndi-name>~~
   </message-driven-descriptor>
  </weblogic-enterprise-bean>
</weblogic-ejb-jar>
```

Where the jndi.name wmq.jmsra is the value in the your resource adapter <jndi-name>…. .

Rebuild your applications by recreating the jar file using the existing classes and the updated files in the META-INF directiom.

Check that the connection factories you are using are in the JNDI tree (see above).
Use
java weblogic.appmerge -verbose ~/myMDB/MDB.jar 1>aa 2>bb to check the syntax and consistency of the file before deploying it.

Deploy it
Use it
Celebrate your success.

How can I tell if my connection factory is JMS or resource adapter?

If you start at the Admin Console, home page. In the pane on the left expand "Environment", click on "Servers". This gives a list of your servers. Click on one. Near the top is "View JNDI tree", click on it. In the left pane is a list of the JNDI tree elements. Locate and click on your connection factory.
This gave me

Binding Name:	CF1
Class:	com.ibm.mq.**jms.MQ**ConnectionFactory
toString Results:	| com.ibm.mq.jms.MQConnectionFactory@c79b6829 :- …. XMSC_CONNECTION_TYPE_NAME :- com.ibm.msg.client.wmq ...

And

Binding Name:	CF3RA
Class:	com.ibm.mq.**connector.outbound.**ConnectionFactoryImpl
toString Results:	com.ibm.mq.connector.outbound.ConnectionFactoryImpl@1abb7687