

# WA1359 WebSphere v6 Performance Tuning



## JMS and Message Driven Bean Tuning

(C) Web Age Solutions Inc.



## Overview

---

- JMS Connection Factory
- Connection Pool
- Session Pool
- Message Listener Service Thread Pool
- Listener Port
- Guidelines
- Example

(C) Web Age Solutions Inc.



## Queue Connection Factory

- Represents a connection to the queue manager.
- Through this Connection Factory one can control how many connections are opened at the same time.
  - Using too few connections will under-utilize the CPU
  - Too many and you may overload the CPU.

(C) Web Age Solutions Inc.

From the web based admin console you can create a JMS Queue Connection Factory. From the left side of your administrative console expand Resources - > JMS Providers and click on WebSphere MQ. Under Additional Properties choose WebSphere MQ Queue Connection Factories. Click New and create your factory. Ensure that the Queue Managers name is entered in the property.



## Connection Pool

- Can set the maximum number of connections in the pool for the Connection Factory.
- Sets the limit to how many connections can be opened to the queue manager at the same time.
- In admin console, create new WebSphere MQ Connection factory
  - Resources->JMS Providers->WebSphere MQ->WebSphere MQ queue connection factories.
- After the factory is created, open its properties page.
  - Ensure that Enable MQ connection pooling is checked.
- Click on Connection Pool.
- Enter
  - Maximum connections
  - Minimum connections
  - Unused timeout – A connection is physically closed if it has not been used in a while.

(C) Web Age Solutions Inc.



## Session Pool

- A JMS session is equivalent to a transaction. Multiple sessions can be opened from the same connection and transactions can be started in parallel.
  - A session can only be used from a single thread.
- Each connection to the queue manager (connection factory) can maintain a pool of sessions.
  - This optimizes the need to create a new session on demand
- After the queue connection factory is created, open its properties page.
- Click on Session Pool.
- Enter
  - Maximum connections – Maximum number of sessions per connection.
  - Minimum connections
  - unused timeout
- Note, use of the word "connection" in this page is misleading. It should really be "sessions".

(C) Web Age Solutions Inc.

Sessions introduce the primary difference between JMS and JDBC. In JDBC, a transaction is started directly using the connection. As a result, to start multiple transactions in parallel, you need multiple connections. In JMS, you can have a single connection. Obtain multiple sessions from the same connection and start several transactions using them from separate threads.

[WebSphere MQ messaging provider](#) > [WebSphere MQ queue connection factories](#) > [Queue Connection Factory](#) > [Session pool](#)

Configuration

---

**General Properties**

Scope

Connection timeout

Maximum connections

Minimum connections

Reap time

Unused timeout

Aged timeout

Purge policy  
 ▼



## Using Sessions

- Sessions and connections are used by any application code written using the JMS API. This includes:
  - Custom developed Servlets and EJBs
  - WebSphere itself as it feeds messages to Message Driven Beans.
- In custom developed code, you will most likely:
  - Open a connection to the queue manager from the pool.
  - Get a session from the connection's session pool. Start a transaction.
  - Perform message input/output.
  - End the transaction. Return the session and connection to the pool.
  - In this mode, it is sufficient to have only one session per connection in the pool. You will rarely run parallel transactions using the same connection.
- For Message Driven Beans, a Listener Port component retrieves messages as follows:
  - Only one connection is opened per listener port. This connection is returned back to the pool only when the listener port shuts down.
  - A listener port starts a transaction using a session from the pool, retrieves a message, invokes the onMessage method of MDB and then commits or rolls back the transaction. The session is sent back to the pool.
  - You can configure the system so that the listener port processes multiple messages in parallel. In this case, several sessions from the pool are used. It continues to use only one connection.

(C) Web Age Solutions Inc.

In real life, if you are using JMS API from a Servlet or EJB, you will need only one session per connection. Multiple sessions per connection becomes a possibility only when you have Message Driven Beans. We will discuss later, how we can configure the system so that a listener port processes multiple messages in parallel transactions. Each transaction requires one session from the pool.



# Listener Port

- A listener port is essentially a reference to a queue connection factory and queue or topic.
  - Each MDB is associated with a listener port. When a message arrives in the queue, the listener port notifies the MDB.
- By default, the listener port starts a transaction using a session, processes a single message and then ends the transaction.
  - You can configure how many messages will be processed in "batch" per transaction.
  - This is configured using the "Maximum messages" property.
- You can configure how many parallel transactions will be used by the port to process messages.
  - Each parallel transaction will need a separate thread and session.
  - This is configured as the "Maximum sessions" property of the listener port.
- Creating listener ports:
  - Open the Messaging Listener Service properties screen of the application server. Then choose Listener Port.
  - Enter name, JMS Queue Connection Factory name, JMS Queue Connection name and set the maximum sessions.
  - Configure the "Maximum sessions" and "Maximum messages" property as needed.
- When installing an enterprise application, specify the listener port name for each MDB.

(C) Web Age Solutions Inc.

[Application servers](#) > [server1](#) > [Message Listener Service](#) > [Listener Ports](#) > [New](#)

Listener ports for Message Driven Beans to listen upon for messages. Each port specifies the JMS Connection Factory and JMS Destination that an MDB, deployed against that port, will listen upon.

Runtime Configuration

---

### General Properties

\* Name

\* Initial State  
Started

Description

\* Connection factory JNDI name

\* Destination JNDI name

Maximum sessions

Maximum retries

Maximum messages



## Message Listener Service Thread Pool

- Each message is processed by an MDB in a separate thread.
- The message listener service maintains a pool of threads for this purpose.
  - Can specify minimum and maximum number of threads in the pool.
- The max. number should be that the system is using 80% of CPU under load.
  - Monitor CPU usage during stress test.
- Only one message listener service thread pool exists per application server. Keep in mind that, all listener ports in that server will use threads from the same thread pool.

(C) Web Age Solutions Inc.

Each listener port needs as many threads as the number of "Maximum sessions" configured for the port. If you have configured three listener ports and the sum total of all "Maximum sessions" is 15 then you need at least 15 threads in the thread pool. Under full load, system will instantiate 15 MDB beans and process messages in 15 parallel transactions.

If you have fewer number of threads in the thread pool, some of the listener ports will not be able to start as many transactions as stated by the "Maximum sessions" setting.

In a way the thread pool provides a single point of throttle for how many messages should be processed in parallel. This is a quick way to control CPU usage. However, you should also carefully control the maximum sessions for the listener ports. This will require knowledge of how expensive it is to processes a specific type of message. This is discussed in more details later.

## Message Listener Service Thread Pool

- The Thread Pool is configured from the listener port.
- From administrative console expand Servers - > Application Servers. Click on the application server that will host the listener service.
- Under Communications expand Messaging and click on Message Listener Service.
- Click on Thread Pool and set minimum and maximum.

(C) Web Age Solutions Inc.

[Application servers](#) > [server1](#) > [Message Listener Service](#) > [Thread Pool](#)

A thread pool allows components of the server to reuse threads to eliminate the need to create new threads at runtime. Creating new threads is typically a time and resource intensive operation.

Configuration

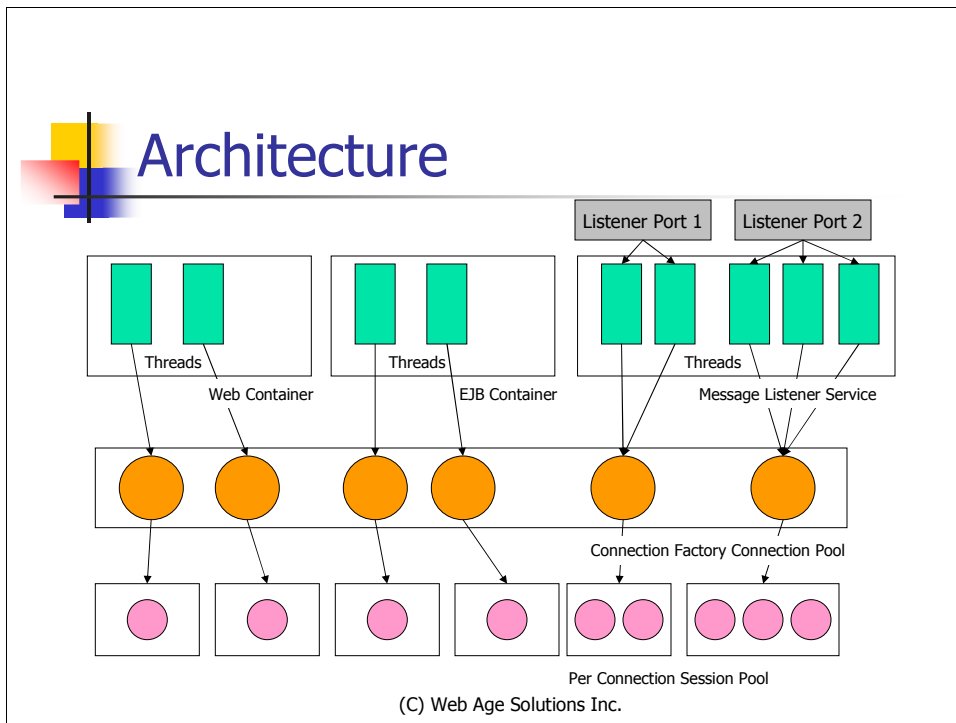
**General Properties**

\* Minimum Size  threads

\* Maximum Size  threads

\* Thread inactivity timeout  milliseconds

Allow thread allocation beyond maximum thread size



In the example above, Listener Port 1 is configured with a maximum sessions of 2 and Listener Port 2 with 3. They each use only one connection from the pool. The number of maximum sessions opened using this connection is 2 and 3 respectively. Also, their thread usage is same as maximum sessions. One session = one transaction = one thread.

In contrast, most Servlet and EJB code will open only one session using a single connection. As a result, their per connection session pool demand will not exceed 1.



## Guidelines

- First, find out which components in your application use JMS.
  - Servlet, EJB and MDB.
- Calculate maximum demand for connections to the connection factory (queue manager)
  - Number of Servlets requests likely to open connection at the same time +
  - Number of EJBs likely to open connection at the same time +
  - Number of listener ports configured to use the connection factory.
- Configure the connection pool accordingly.
- If you are noticing heavy CPU usage under load, lower the demand for connection factory connections. This can be done in many ways:
  - Lower web container thread pool maximum if too many Servlets opening connection.
  - Lower the queue connection factory connection pool maximum.
  - Lower the message listener service thread pool maximum if MDBs are taking up CPU.
- Carefully set the maximum sessions property of a listener port, as this will determine how many messages are processed in parallel.
  - A high value will increase CPU usage.

(C) Web Age Solutions Inc.



## Guidelines

- Make sure that per connection session pool maximum is as much as listener port maximum sessions.
- Make sure that the listener service thread pool maximum is as much as the sum total of maximum sessions of all listener ports.
  - Otherwise, under load, some of the listener ports will not be able to process as many messages in parallel as configured using "maximum sessions".
- Some queues contain messages that require extra CPU to process.
  - Allocate lower number of sessions for the listener port associated with that queue to keep the CPU usage within control.
- Play with the maximum number of messages processed per transaction.
- Make sure the queue connection factory has Enable MQ connection pooling checked.
- For large messages, make sure that MQ queue manager is running in the same machine as the WebSphere application server where the MDB is installed.
  - Then choose BINDINGS as the value of the Transport type property of the queue connection factory.
  - If the queue manager is remote and Transport type value is CLIENT, the listener service will take a long time to download the message there by keeping a thread busy for a long time.

(C) Web Age Solutions Inc.

### **Maximum Messages Per Transaction**

This is a property of the listener port. By default, only one message is processed (one call to onMessage of the associated MDB) per transaction. If your message size is small, you may get better performance by processing several messages in a single transaction. This reduces the need to setup and clean up the number of transactions.

Keep in mind, multiple messages will be processed in a batch only if messages are available in the queue. If for example, the maximum message is set to 5 and you have only 3 messages remaining in the queue, system will process 3 messages and end the transaction.

If the transaction is rolled back for some reason, all messages processed so far within that transaction will be re-processed at a later time.



## Example

- There are two MDBs, MDB1 and MDB2. They need to process messages from two separate queues – Q1 and Q2 respectively.
  - For the sake of simplicity, both queues are from the same queue manager.
- There are no Servlet or EJB that use JMS.
- Create a connection factory for the queue manager.
- Create two listener ports – LP1 for Q1 and LP2 for Q2.
  - Associate MDB1 with LP1 and MDB2 with LP2.
- As we have two listener ports, our maximum demand for connections is two. Set that as the connection pool maximum.
- Let's say messages in Q2 are large and complex and take more CPU to process.
  - Set the maximum sessions property of LP1 to 5 and LP2 to 3.
- We need a total of 5 + 3 or 8 threads. Set that as the message listener service thread pool maximum.
- Set the per connection session pool maximum to 5.
- These are our initial settings. Run stress test by putting messages in Q1 and Q2 at a realistic rate.
- If CPU usage is too high, lower maximum sessions property of the listener ports.
  - Another quick way to clamp down will be to lower the listener service thread pool maximum.
- If CPU usage is too low, increase listener port maximum sessions and listener service thread pool maximum.

(C) Web Age Solutions Inc.



## Summary

---

- In this chapter we found out:
  - How to calculate the demand for connections to a queue connection factory (queue manager)
  - How to configure the per connection session pool.
  - How Listener Ports allow us to control how many messages from a queue will be processed in parallel.
  - How to process multiple messages "in batch" within a single transaction.

(C) Web Age Solutions Inc.