Administration and Monitoring SAP JMS Provider Using Telnet Commands

Applies to:

SAP NetWeaver Composition Environment 7.1 SP3 and later

Summary

This article is a short reference to the Telnet commands for administration and configuration of the SAP JMS Provider. They can be used for checking the status of the runtime JMS resources (Connections, Sessions, Producers, Consumers, etc.) and they may be very helpful investigating some issues in applications using JMS.

Authors: Rositza Andreeva, Ekaterina Anachkova

Company: SAP Labs Bulgaria

Created on: 21 April 2008

Author Bio

Rositza Andreeva is a developer at the JMS team. She is responsible for JMS monitoring and administration modules.

Ekaterina Anachkova is an information developer at SAP Labs Bulgaria. She writes in the areas of Java Message Service (JMS) and Enterprise JavaBeans (EJB).

© 2008 SAP AG

Table of Contents

Administration and Monitoring 3
Telnet Commands Related to the SAP JMS Provider 3
The `jms_list` command....3
The `list_temp_destinations` command....5
Example Scenario 5
Summary....10

© 2008 SAP AG

Administration and Monitoring

There are different tools for administration and monitoring the Application Server (AS) Java – the SAP NetWeaver Administrator, the Config Tool, the Java EE Telnet, etc. In this article, we will review the available Telnet commands for JMS administration and monitoring.

Telnet Commands Related to the SAP JMS Provider

The SAP JMS Provider service provides Telnet commands for administering and monitoring the JMS resources in a server. To use these commands, you have to connect to the AS Java using Telnet and enable the JMS commands group with **`add jms`**. To show the help information about the available commands under the `jms` group, you can type one of the following at the Telnet command prompt:

- **`jms –h`**
- **`jms -?`**
- **`man jms`**

There are several subgroups of JMS Telnet commands:

- **`jms_list`** – this command provides details about the JMS runtime environment
- **`list_temp_destinations`** – this one displays all currently active temporary destinations.

The `jms_list` command

The **`jms_list`** command shows information about the JMS runtime objects – Connections, Sessions, Producers, Consumers, etc.

The generic command is **`jms_list [<command>] [<destinationName>] [<virtualProviderName>]`**, where the available options are listed below, the `destinationName` specifies the JMS destination and the `virtualProviderName` specifies the JMS Virtual Provider.

Command	Description	Example
jms_list **destinations** [<virtualProviderName>]	Displays the currently active destinations for the specified Virtual Provider. Note that this command provides runtime information – only destinations with currently registered producers and/or consumers are listed.	`jms_list destinations default`
	If you do not specify a Virtual Provider, the active destinations for all Virtual Providers are displayed.	`jms_list destinations`
jms_list **config_destinations** [<virtualProviderName>]	Displays all destinations for the specified Virtual Provider. This command provides a snapshot of all the configured destinations, including the ones that currently do not have any registered producers and/or consumers.	`jms_list config_destinations default`
	If you do not specify a Virtual	`jms_list`

	Provider, all destinations for all Virtual Providers are displayed.	`config_destinations`
jms_list **config_factories** [<virtualProviderName>]	Displays all connection factories for the specified Virtual Provider.	`jms_list config_factories default`
	If you do not specify a Virtual Provider, all connection factories for all Virtual Providers are displayed.	`jms_list config_factories`
jms_list **connections** [<virtualProviderName>]	Displays the currently active connections created from connection factories belonging to the specified Virtual Provider.	`jms_list connections default`
	If you do not specify a Virtual Provider, all currently active connections created from connection factories belonging to any Virtual Provider are displayed.	`jms_list connections`
jms_list **sessions** [<virtualProviderName>]	Displays the currently active sessions created from connection of the specified Virtual Provider.	`jms_list sessions default`
	If you do not specify a Virtual Provider, all currently active sessions for all Virtual Providers are displayed.	`jms_list sessions`
jms_list **producers** [<virtualProviderName>]	Displays the currently active producers registered to destinations belonging to the specified Virtual Provider.	`jms_list producers default`
	If you do not specify a Virtual Provider, all currently active producers registered to destinations belonging to any Virtual Provider are displayed.	`jms_list producers`
jms_list **consumers** [<virtualProviderName>]	Displays the currently active consumers registered to destinations belonging to the specified Virtual Provider.	`jms_list consumers default`
	If you do not specify a Virtual Provider, all currently active consumers registered to destinations belonging to any Virtual Provider are displayed.	`jms_list consumers`

© 2008 SAP AG

jms_list **browsers** [<virtualProviderName>]	Displays the currently active browsers registered to Queues belonging to the specified Virtual Provider.	`jms_list browsers default`
	If you do not specify a Virtual Provider, all currently active browsers registered to Queues belonging to any Virtual Provider are displayed.	`jms_list browsers`
jms_list **vps**	Displays information about all Virtual Providers.	`jms_list vps`
jms_list **subscriptions** [<virtualProviderName>]	Displays the durable subscriptions registered to Topics belonging to the specified Virtual Provider.	`jms_list subscriptions default`
	If you do not specify a Virtual Provider, all registered durable subscriptions registered to Topics belonging to any Virtual Provider are displayed.	`jms_list subscriptions`
jms_list **msg** [<destinationName>] [<virtualProviderName>]	Lists the messages present in the database for the particular destination. You must specify both `destinationName` and `virtualProviderName`.	`jms_list msg sapDemoQueue default`

The `list_temp_destinations` command

The **`list_temp_destinations`** command displays details about the currently existing temporary JMS destinations.

The generic command is **`list_temp_destinations <virtualProviderName>`**, where the `<virtualProviderName>` specifies the JMS Virtual Provider, for which the temporary destinations will be listed.

Note: It is obligatory to specify the **`<virtualProviderName>.`**

Example: `list_temp_destinations default`

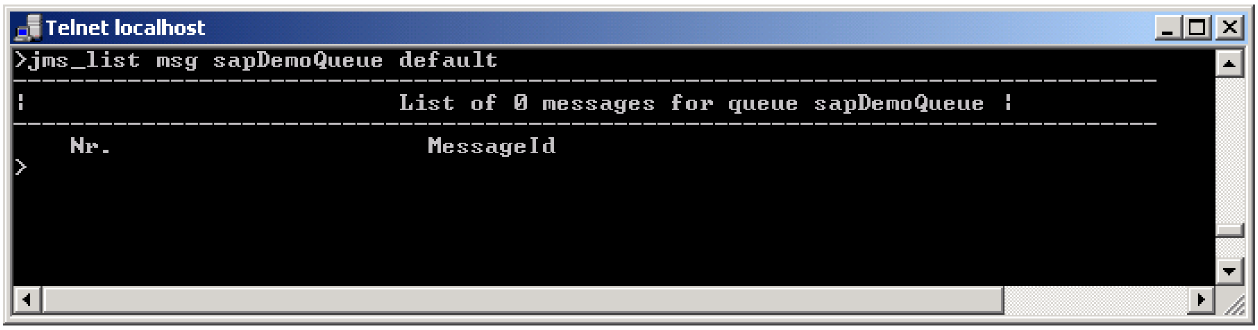
Example Scenario

All JMS-related Telnet commands give you the possibility to monitor the runtime state of the SAP JMS Provider. To illustrate the usage of the described Telnet commands, let us consider one simple scenario. Imagine we just discovered that some persistent messages are not delivered to our application and we want to investigate what might have happened with them. For the purposes of this example, we will use the `sapDemoQueue` destination and the `default` JMS Virtual Provider.

The following procedure describes one possible path of investigation and the respective sequence of commands.

1. **`jms_list msg sapDemoQueue default`**

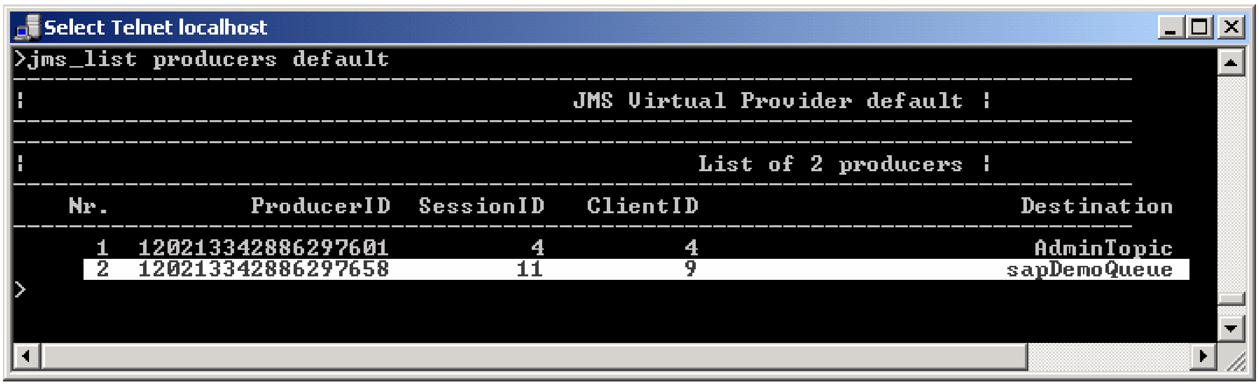
   First, we need to make sure the messages we expect are actually *produced*. The `jms_list msg` command lists all messages sent to the `sapDemoQueue` destination that are present in the database. If there are no messages in this list, we know that there are currently no messages pending for delivery - either no messages have been produced, or all that have been produced have already been consumed and acknowledged. We can try to determine which producer was supposed to send them.



```
>jms_list msg sapDemoQueue default
--------------------------------------------------------------------------------
|                          List of 0 messages for queue sapDemoQueue |
--------------------------------------------------------------------------------
    Nr.                       MessageId
>
```

2. **`jms_list producers default`**

   This command lists all producers registered to destinations belonging to the `default` JMS Virtual Provider. Note that this is the same Virtual Provider to which our destination belongs. From this list we can determine the producer ID, the destination to which the producer sends messages, its session ID and client ID. By the client ID, we can later on find out the consumer that is supposed to receive the messages. In this case, we look for producers registered to the `sapDemoQueue` destination. This is a way to determine if there is a currently active producer registered to our destination.

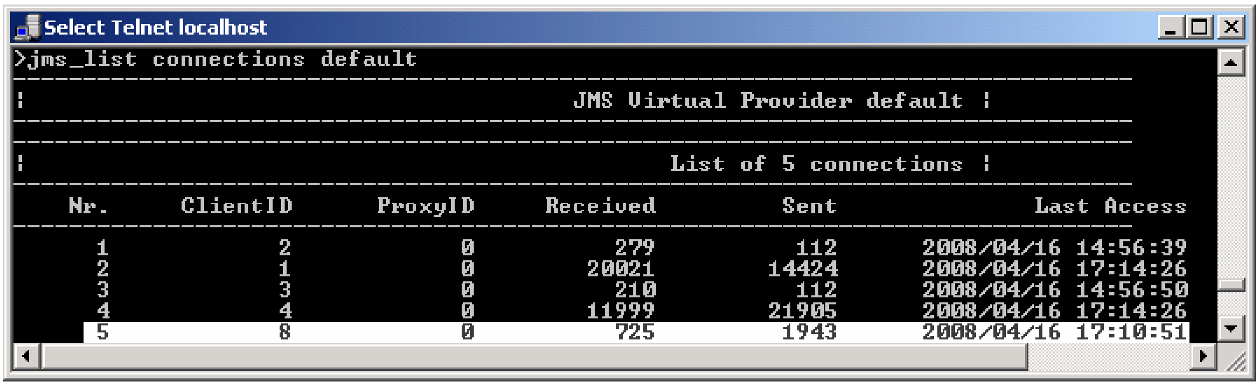


```
>jms_list producers default
--------------------------------------------------------------------------------
|                                         JMS Virtual Provider default |
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
|                                                  List of 2 producers |
--------------------------------------------------------------------------------
    Nr.          ProducerID  SessionID    ClientID                 Destination
--------------------------------------------------------------------------------
      1  120213342886297601          4           4                  AdminTopic
      2  120213342886297658         11           9                sapDemoQueue
>
```

If there are messages pending to be delivered, then we have to continue our investigation with the consumers that are expected to receive them. We can check the status of the JMS connection – how many bytes have been sent and received through it and when it was last accessed.

3. **`jms_list connections default`**

   We use the client ID to check if there are any active connections and when for the last time was the particular connection accessed. The JMS Virtual Provider again has to be the same.



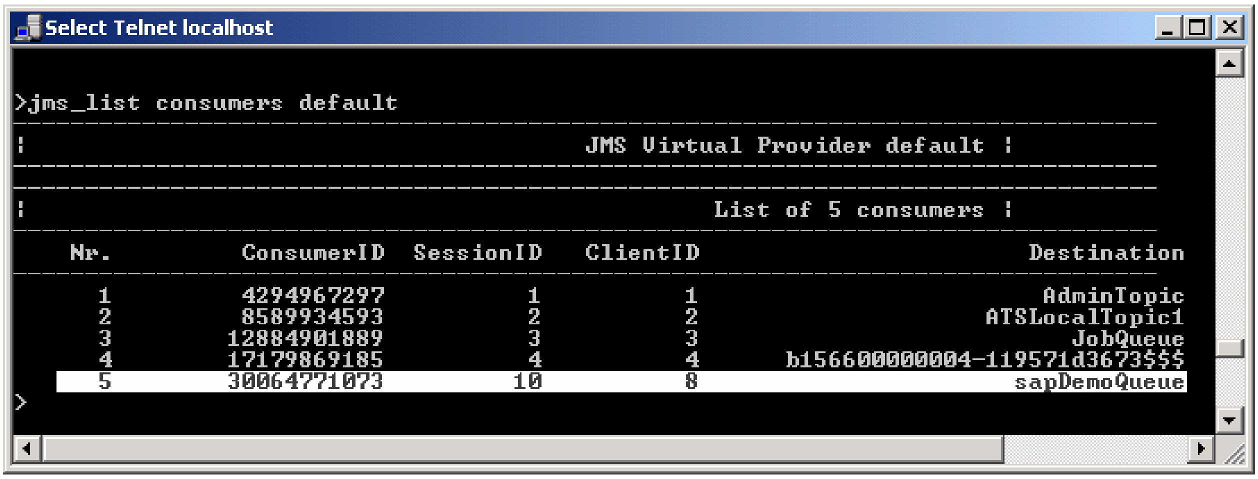
```
>jms_list connections default
-------------------------------------------------------------------------------
|                                        JMS Virtual Provider default |
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
|                                               List of 5 connections |
-------------------------------------------------------------------------------
    Nr.       ClientID       ProxyID     Received         Sent          Last Access
-------------------------------------------------------------------------------
     1             2             0          279           112   2008/04/16 14:56:39
     2             1             0        20021         14424   2008/04/16 17:14:26
     3             3             0          210           112   2008/04/16 14:56:50
     4             4             0        11999         21905   2008/04/16 17:14:26
     5             8             0          725          1943   2008/04/16 17:10:51
```

Note: If you want to find the corresponding connection to your consumer, you need the connection with client ID that is equal to the one of the already found consumer.

We can also check the status of the consumer(s) registered to the `sapDemoQueue` destination.

4. **`jms_list consumers default`**

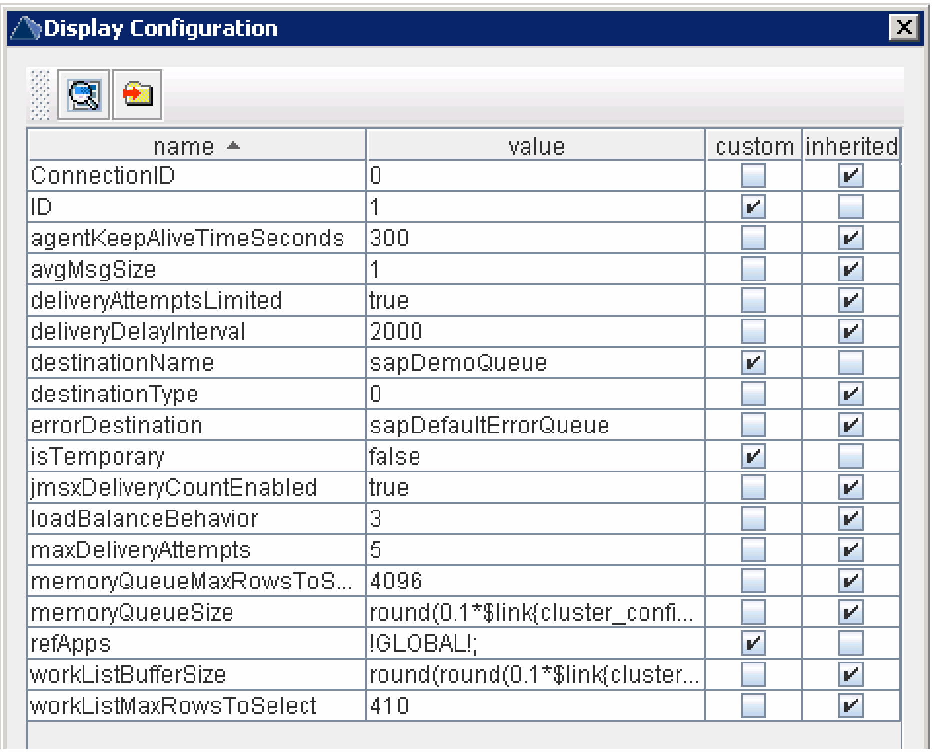
This command lists all currently active consumers registered to destinations belonging to the `default` JMS Virtual Provider. From this list we can determine the consumer ID, the destination to which the consumer is registered, the session ID and the client ID. If there is no consumer registered to `sapDemoQueue`, then we know that our application does not receive messages because it failed for some reason to create the respective consumer(s) and we can continue the investigation in this direction, for example by checking the server traces for relevant exceptions.



```
>jms_list consumers default
---------------------------------------------------------------------------------
|                                        JMS Virtual Provider default |
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
|                                                   List of 5 consumers |
---------------------------------------------------------------------------------
    Nr.         ConsumerID  SessionID   ClientID                    Destination
---------------------------------------------------------------------------------
     1          4294967297          1          1                     AdminTopic
     2          8589934593          2          2                 ATSLocalTopic1
     3         12884901889          3          3                       JobQueue
     4         17179869185          4          4  b156600000004-119571d3673$$$
     5         30064771073         10          8                   sapDemoQueue
>
```

If there is an active consumer but it still does not receive any of the pending messages, it is possible that there is an issue in the application message processing logic which causes the messages to be redelivered again and again. By default, message delivery attempts are limited and once they are exhausted for a particular message, it is considered undeliverable (dead) and it is skipped by the consumer and moved to the configured error destination of the original destination. To determine the error destination of the `sapDemoQueue` destination, we have to use the *Configuration Editor*. In the *Display configuration* tab, expand *Configurations → jms_provider → default → queues → sapDemoQueue → Propertysheet data*. In the Property Sheet you can find the error destination of a particular destination. In our case, the error destination of `sapDemoQueue` is `sapDefaultErrorQueue`.

© 2008 SAP AG



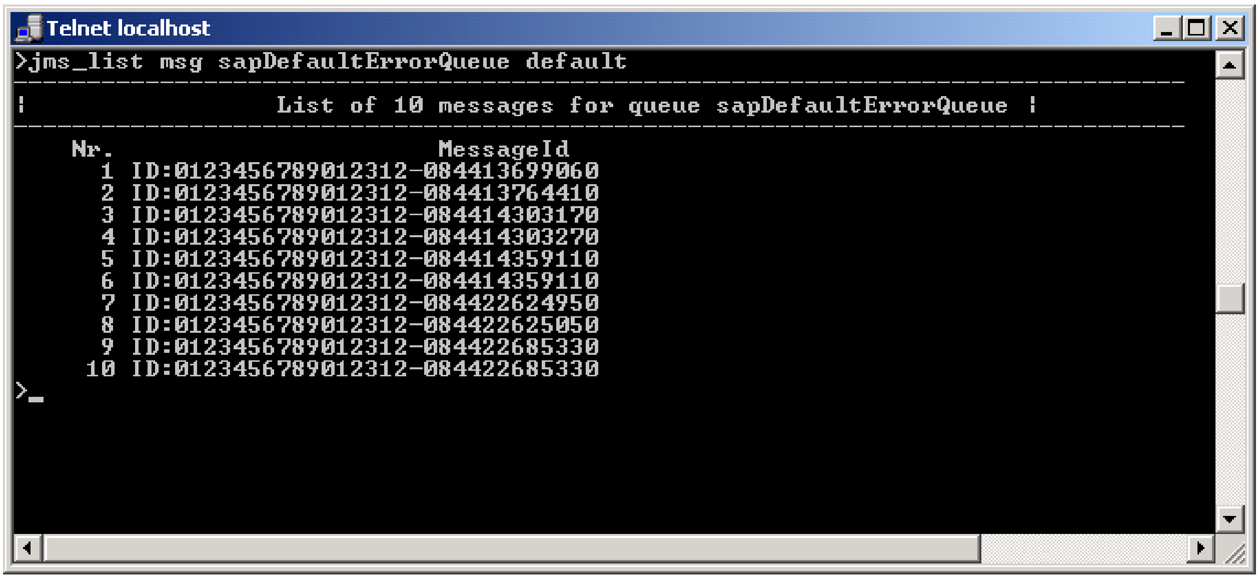
Display Configuration

name ▲	value	custom	inherited
ConnectionID	0	☐	☑
ID	1	☑	☐
agentKeepAliveTimeSeconds	300	☐	☑
avgMsgSize	1	☐	☑
deliveryAttemptsLimited	true	☐	☑
deliveryDelayInterval	2000	☐	☑
destinationName	sapDemoQueue	☑	☐
destinationType	0	☐	☑
errorDestination	sapDefaultErrorQueue	☐	☑
isTemporary	false	☑	☐
jmsxDeliveryCountEnabled	true	☐	☑
loadBalanceBehavior	3	☐	☑
maxDeliveryAttempts	5	☐	☑
memoryQueueMaxRowsToS...	4096	☐	☑
memoryQueueSize	round(0.1*$link{cluster_confi...	☐	☑
refApps	!GLOBAL!;	☑	☐
workListBufferSize	round(round(0.1*$link{cluster...	☐	☑
workListMaxRowsToSelect	410	☐	☑

Then we can check if there are any messages in the error destination.

5. **`jms_list msg sapDefaultErrorQueue default`**

With this command we check if the missing messages are present in the error destination.



Telnet localhost

```
>jms_list msg sapDefaultErrorQueue default
--------------------------------------------------------------------------------
|                  List of 10 messages for queue sapDefaultErrorQueue |
--------------------------------------------------------------------------------
    Nr.                        MessageId
      1  ID:0123456789012312-084413699060
      2  ID:0123456789012312-084413764410
      3  ID:0123456789012312-084414303170
      4  ID:0123456789012312-084414303270
      5  ID:0123456789012312-084414359110
      6  ID:0123456789012312-084414359110
      7  ID:0123456789012312-084422624950
      8  ID:0123456789012312-084422625050
      9  ID:0123456789012312-084422685330
     10  ID:0123456789012312-084422685330
>_
```

If our application is unable to consume some of the messages, we have to check why and then we may want to do something with the undelivered messages. Since error destinations are just ordinary JMS destinations, you can access dead messages using the standard JMS API – for example, your application (or a dedicated tool) can consume and process the messages from the error destination – it can even return them back to the original destination, if that is the error handling logic of the application.

Note that we can configure the following properties on the `jms-resources.xml` related to the dead messages functionality:

- `deliveryAttemptsLimited` - a Boolean property that indicates whether the message delivery attempts are limited. The default value is "true".
- `maxDeliveryAttempts` - an Integer property that indicates the maximum number of delivery attempts before the message is considered undeliverable (dead). The default value is 5.
- `deliveryDelayInterval` - the delay in milliseconds between two consecutive message delivery attempts. The default value of this property is 2000 milliseconds.
- `errorDestination` - the name of a JMS Queue where dead messages will be forwarded. If you leave this property blank (“”), this means that you want dead messages to be discarded.

These four properties are configurable per JMS destination.

Note: The default error destination has an empty string for the `errorDestination` property, otherwise, when a message becomes dead in its original destination and then it also becomes dead in the error destination, this may lead to several transfers of this message through error destinations and potentially this may even lead to a message delivery endless loop.

Note: The value of the `errorDestination` property must be the name of an already existing Queue.

Here is an example of a jms-resources.xml our application may have which contains configurations of the above properties.

```
<?xml version="1.0" encoding="UTF-8"?>
<jms-resources xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="jms-resources.xsd">
   <application-name>MyApplication</application-name>
   <connection-factory>
         <name>MyQueueFactory</name>
         <sap-local-factory-type>
                <type>javax.jms.QueueConnectionFactory</type>
                <virtual-provider>default</virtual-provider>
                <property>
                       <description>Client ID</description>
                       <config-property-name>clientID</config-property-name>
                       <config-property-value>
                             myClientApp
                       </config-property-value>
                </property>
         </sap-local-factory-type>
   </connection-factory>
   <destination>
         <name>sapDemoQueue</name>
         <type>javax.jms.Queue</type>
         <sap-local-destination-type>
                <virtual-provider>default</virtual-provider>
                <property>
                       <description>
                             Message Delivery Attempts Limited
                       </description>
                       <config-property-name>
```

```
                                    deliveryAttemptsLimited
                            </config-property-name>
                            <config-property-value>true</config-property-value>
                    </property>
                    <property>
                            <description>Maximum Delivery Attempts</description>
                            <config-property-name>
                                    maxDeliveryAttempts
                            </config-property-name>
                            <config-property-value>30</config-property-value>
                    </property>
                    <property>
                            <description>Delay in Milliseconds</description>
                            <config-property-name>
                                    deliveryDelayInterval
                            </config-property-name>
                            <config-property-value>60000</config-property-value>
                    </property>
                    <property>
                            <description>Error Destination</description>
                            <config-property-name>
                                    errorDestination
                            </config-property-name>
                            <config-property-value>
                                    sapDefaultErrorQueue
                            </config-property-value>
                    </property>
            </sap-local-destination-type>
    </destination>
</jms-resources>
```

Summary

In this article we reviewed the available Telnet commands related to the SAP JMS Provider. We have also shown how you can use these commands in combination to investigate a particular problem or just to monitor the runtime status of the JMS Provider.

© 2008 SAP AG

Copyright

© 2008 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, System i, System i5, System p, System p5, System x, System z, System z9, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, POWER5+, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

These materials are provided “as is” without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

Any software coding and/or code lines/strings (“Code”) included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.