Qpid JMX Management Console

User Guide

Contents

Introduction

The Qpid JMX Management Console is a standalone Eclipse RCP application for managing and monitoring the Qpid Java server utilising its JMX management interfaces.

This guide will give an overview of configuring the console, the features supported by it, and how to make use of the console in managing the various JMX Management Beans (MBeans) offered by the Qpid Java server.

Startup & Configuration

Startup

The console can be started in the following way, depending on platform:

- **Windows:** by running the *qpidmc.exe* executable file.
- **Linux:** by running the *qpidmc* executable.
- **Mac OS X:** by launching the *Qpid Management Console.app* application bundle.

SSL configuration

Newer Qpid Java servers can protect their JMX connections with SSL, and this is enabled by default. When attempting to connect to a server with this enabled, the console must be able to verify the SSL certificate presented to it by the server or the connection will fail.

If the server makes use of an SSL certificate signed by a known Signing CA (Certification Authority) then the console needs no extra configuration, and will make use of Java's default system-wide CA TrustStore for certificate verification (you may however have to update the system-wide default CA TrustStore if your certified is signed by a less common CA that is not already present in it).

If however the server is equipped with a self-signed SSL certificate, then the management console must be provided with an appropriate SSL TrustStore containing the public key for the SSL certificate, so that it is able to validate it when presented by the server. The server ships with a script to create an example self-signed SSL certificate, and store the relevant entries in a KeyStore and matching TrustStore. This script can serve as a guide on how to use the Java Keytool security utility to manipulate your own stores, and more information can be found in the JSSE Reference Guide: **http://java.sun.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.html#CustomizingStores**

Supplying the necessary details to the console is performed by setting the *javax.net.ssl.trustStore* and *javax.net.ssl.trustStorePassword* environment variables when starting it. This can be done at the command line, but the preferred option is to set the configuration within the *qpidmc.ini* launcher configuration file for repeated usage. This file is equipped with a template to ease configuration, this should be uncommented and edited to suit your needs. It can be found in the root of the console releases for Windows, and Linux. For Mac OS X the file is located within the consoles *.app* application bundle, and to locate and edit it you must select *'Show Package Contents'* when accessing the context menu of the application, then browse to the *Contents/MacOS* sub folder to locate the file.

JMXMP configuration

Older releases of the Qpid Java server can make use of the Java Management Extensions Messaging Protocol (JMXMP) to provide protection for their JMX connections. This occurs when the server has its main configuration set with the management *'security-enabled'* property set to true.

In order to connect to this configuration of server, the console needs an additional library that is not included within the Java SE platform and cannot be distributed with the console due to licensing restrictions.

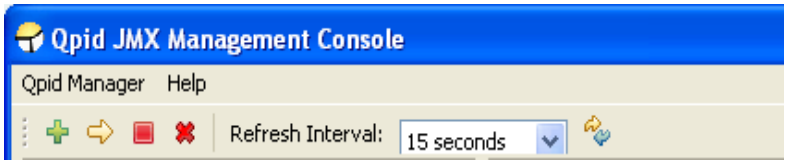
You can download the JMX Remote API 1.0.1_04 Reference Implementation from the Sun website here. The included *jmxremote-1_0_1-bin/lib/jmxremote_optional.jar* file must be added to the *plugins/jmxremote.sasl_1.0.1* folder of the console release (again, in Mac OS X you will need to select *'Show package contents'* from the context menu whilst selecting the management console bundle in order to reveal the inner file tree).

Following this the console will automatically load the JMX Remote Optional classes and negotiate the SASL authentication profile type when encountering a JMXMP enabled Qpid Java server.

Managing Server Connections

Main Toolbar

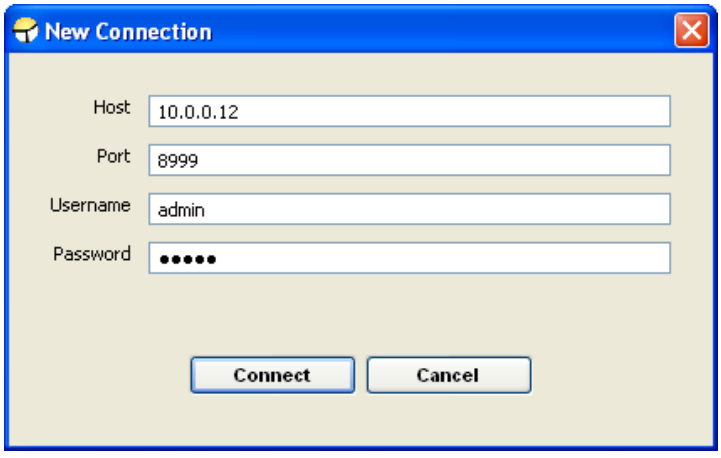
The main toolbar of the console can be seen in the image below. The left most buttons respectively allow for adding a new server connection, reconnecting to an existing server selected in the connection tree, disconnecting the selected server connection, and removing the server from the connection tree.



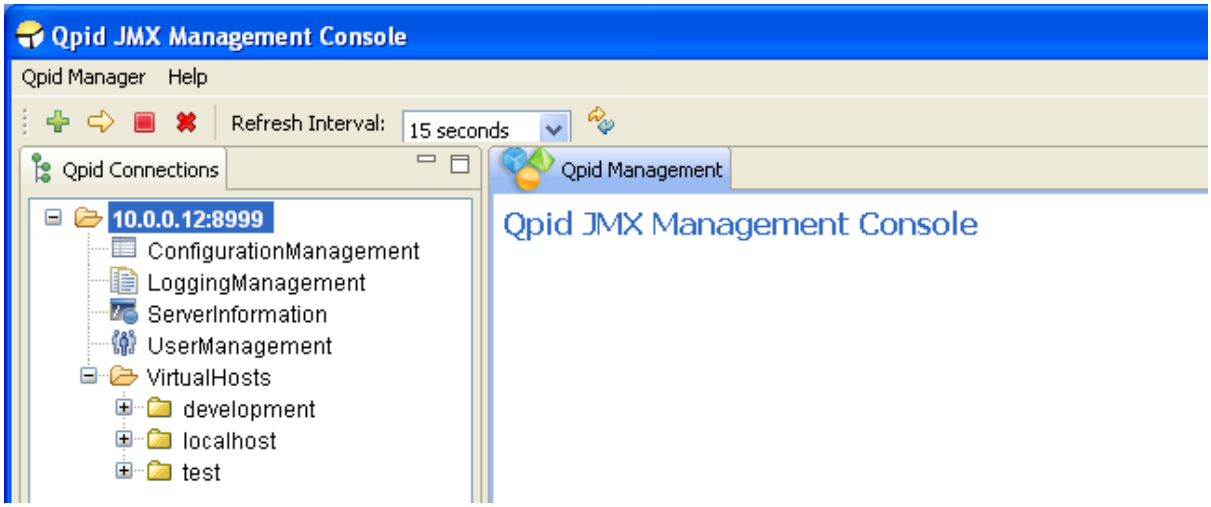
Beside these buttons is a combo for selecting the refresh interval; that is, how often the console requests updated information to display for the currently open area in the main view. Finally, the right-most button enables an immediate update.

Connecting to a new server

To connect to a new server, press the *Add New Server* toolbar button, or select the *Qpid Manager -> Add New Connection* menu item. At this point a dialog box will be displayed requesting the server details, namely the server hostname, management port, and a username and password. An example is shown below:



Once all the required details are entered, pressing Connect will initiate a connection attempt to the server. It the attempt fails a reason will be shown and the server will not be added to the connection tree. If the attempt is successful the server will be added to the connections list and the entry expanded to show the initial administration MBeans the user has access to and any VirtualHosts present on the server, as can be seen in the figure below.



If the server supports a newer management API than the console in use, once connected this initial screen will contain a message on the right, indicating an upgraded console should be sought by the user to ensure all management functionality supported by the server is being utilised.

Reconnecting to a server

If a server has been connected to previously, it will be saved as an entry in the connection tree for further use. On subsequent connections the server can simply be selected from the tree and using the *Reconnect* toolbar button or *Qpid Manager -> Reconnect* menu item. At this stage the console will prompt simply for the username and password with which the user wishes to connect, and following a successful connection the screen will appear as shown previously above.

Disconnecting from a server

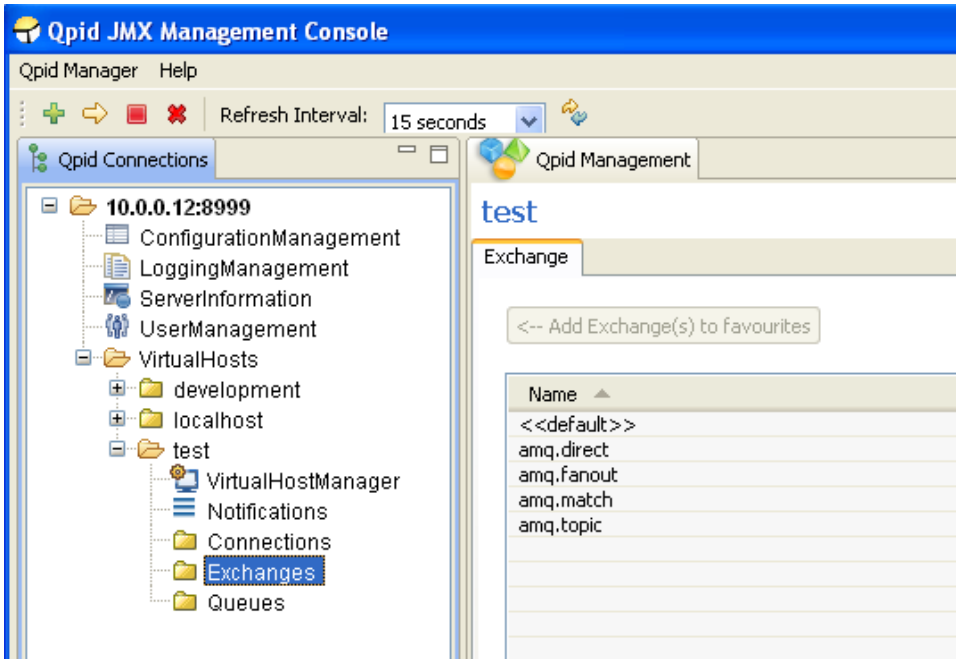
To disconnect from a server, select the connection tree node for the server and press the *Disconnect* toolbar button, or use the *Qpid Manager -> Disconnect* menu option.

Removing a server

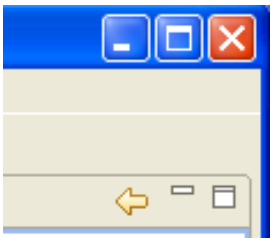
To remove a server from the connection list, select the connection tree node for the server and press the *Remove* toolbar button, or use the *Qpid Manager -> Remove Connection* menu option.

Navigating a connected server

Once connected to a server, the various areas available for administration are accessed using the Qpid Connections tree at the left side of the application. To open a particular MBean from the tree for viewing, simply select it in the tree and it will be opened in the main view.



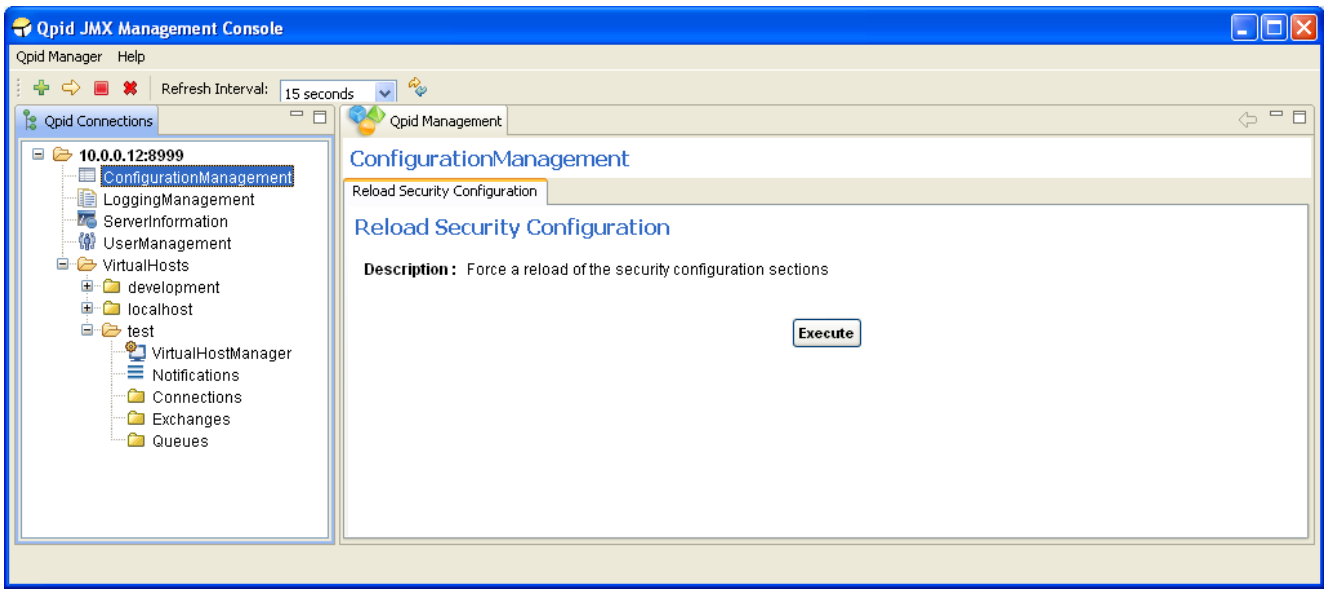
As there may be vast numbers of Queues, Connections, and Exchanges on the server these MBeans are not automatically added to the tree along with the general administration MBeans. Instead, dedicated selection areas are provided to allow users to select which Queue/Connection/Exchange they wish to view or add to the tree. These areas can be found by clicking on the Connections, Exchanges, and Queues nodes in the tree under each VirtualHost, as shown in the figure above. One or more MBeans may be selected and added to the tree as Favourites using the button provided. These settings are saved for future use, and each time the console connects to the server it will check for the presence of the MBean previously in the tree and add them if they are still present. Queue/Connection/Exchange MBeans can be removed from the tree by right clicking on them to expose a context menu allowing deletion.



As an alternative way to open a particular MBean for viewing, without first adding it to the tree, you can simply double click an entry in the table within the Queue/Connection/Exchange selection areas to open it immediately. It is also possible to open some MBeans like this whilst viewing certain other MBeans. When opening an MBean in either of these ways, a Back button is enabled in the top right corner of the main view. Using this button will return you to the selection area or MBean you were previously viewing. The history resets each time the tree is used to open a new area or MBean.

ConfigurationManagement MBean

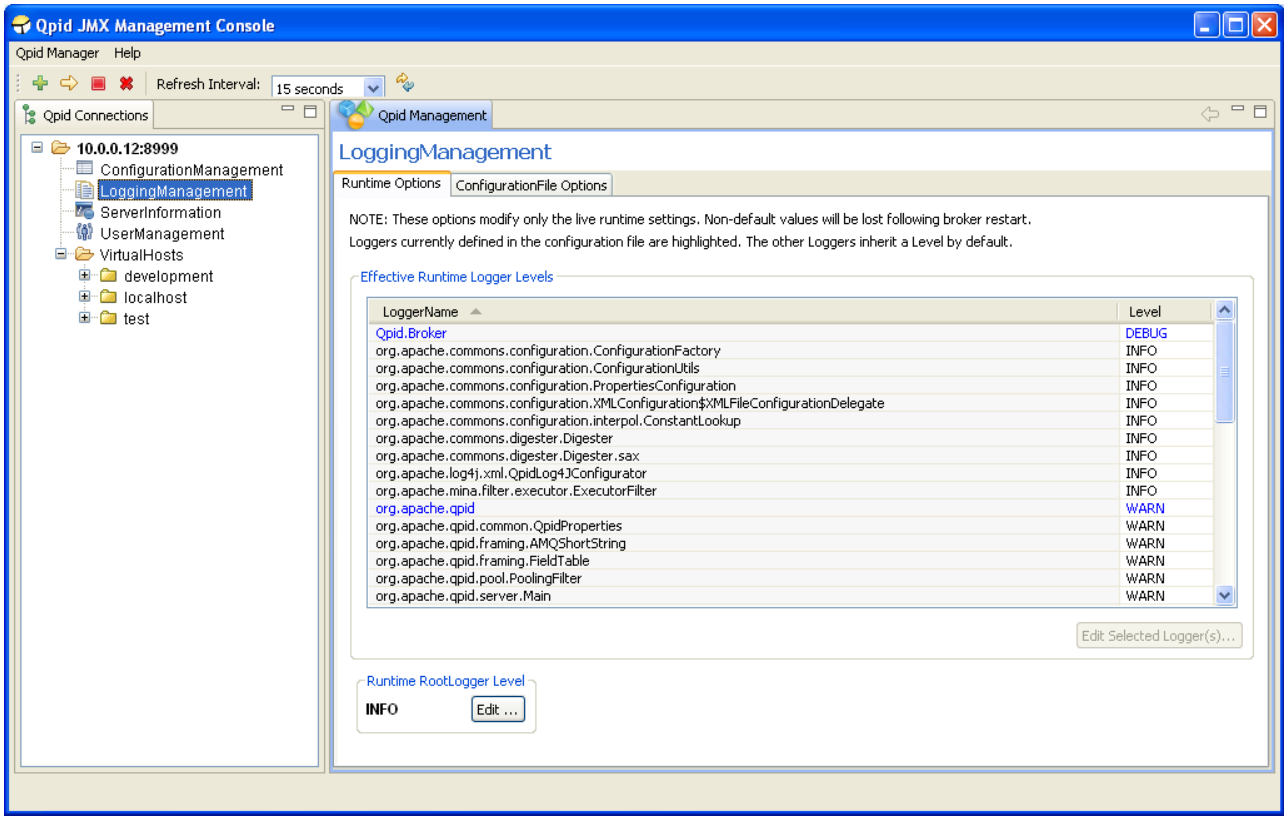
The ConfigurationManagement MBean is available on newer servers, to users with admin level management rights. It offers the ability to perform a live reload of the Security section of the server configuration file, main for allowing update of the firewall configuration to new settings without restarting the server. This can be performed by clicking the Execute button and confirming the prompt which follows.



LoggingManagement MBean

The LoggingManagement MBean is available on newer servers, and accessible by admin level users. It allows live alteration of the logging behaviour, both at a Runtime-only level and at the configuration file level. The latter can optionally affect the Runtime configuration, either through use of the servers automated LogWatch ability which detects changes to the configuration file and reloads it, or by manually requesting a reload. This functionality is split across two management tabs, Runtime Options and ConfigurationFile Options.

Runtime Options

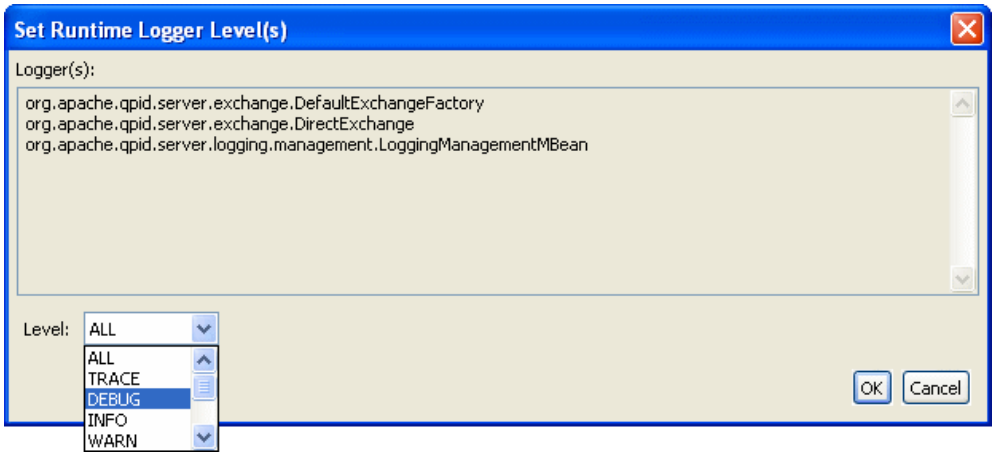


The Runtime Options tab allows manipulation of the logging settings without affecting the configuration files (this means the changes will be lost when the server restarts), and gives individual access to every Logger active within the server.

As shown in the figure above, the table in this tab presents the Effective Level of each Logger. This is because the Loggers form a hierarchy in which those without an explicitly defined (in the logging configuration file) Level will inherit the Level of their immediate parent; that is, the Logger whose full name is a prefix of their own, or if none satisfy that condition then the RootLogger is their parent. As example, take the *org.apache.qpid* Logger. It is parent to all those below it which begin with *org.apache.qpid* and unless they have a specific Level of their own, they will inherit its Level. This can be seen in the figure, whereby all the children Loggers visible have a level of WARN just like their

parent, but the RootLogger Level is INFO; the children have inherited the WARN level from *org.apache.qpid* rather than INFO from the RootLogger.

To aid with this distinction, the Logger Levels that are currently defined in the configuration file are highlighted in the List. Changing these levels at runtime will also change the Level of all their children which haven't been set their own Level using the runtime options. In the latest versions of the LoggingManagement MBean, it is possible to restore a child logger that has had an explicit level se, to inheriting that of its parent by setting it to an INHERITED level that removes any previously set Level of its own.



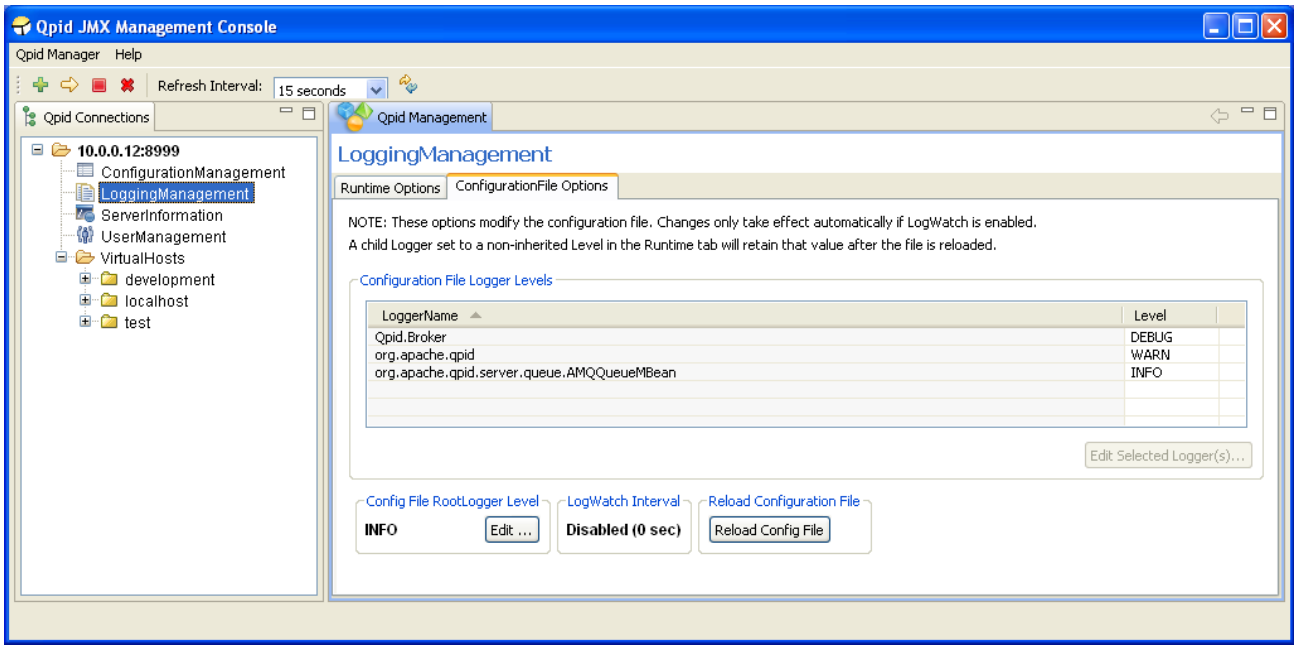
In order to set one of more Loggers to a new Level, they should be selected in the table (or double click an individual Logger to modify it) and the *Edit Selected Logger(s)* button pressed to load the dialog shown above. At this point, any of the available Levels supported by the server can be applied to the Loggers selected and they will immediately update, as will any child Loggers without their own specific Level.

The RootLogger can be similarly edited using the button at the bottom left of the window.

ConfigurationFile Options

The ConfigurationFile Options tab allows alteration of the Level settings for the Loggers defined in the configuration file, allowing changes to persist following a restart of the server. Changes made to the configuration file are only applied automatically while the sever is running if it was configured to enable the LogWatch capability, meaning it will monitor the configuration file for changes and apply the new configuration when the change is detected. If this was not enabled, the changes will be picked up when the server is restarted. The status of the LogWatch feature is shown at the bottom of the tab. Alternatively, in the latest versions of the LoggingManagement MBean it is possible to reload the logging configuration file on demand.

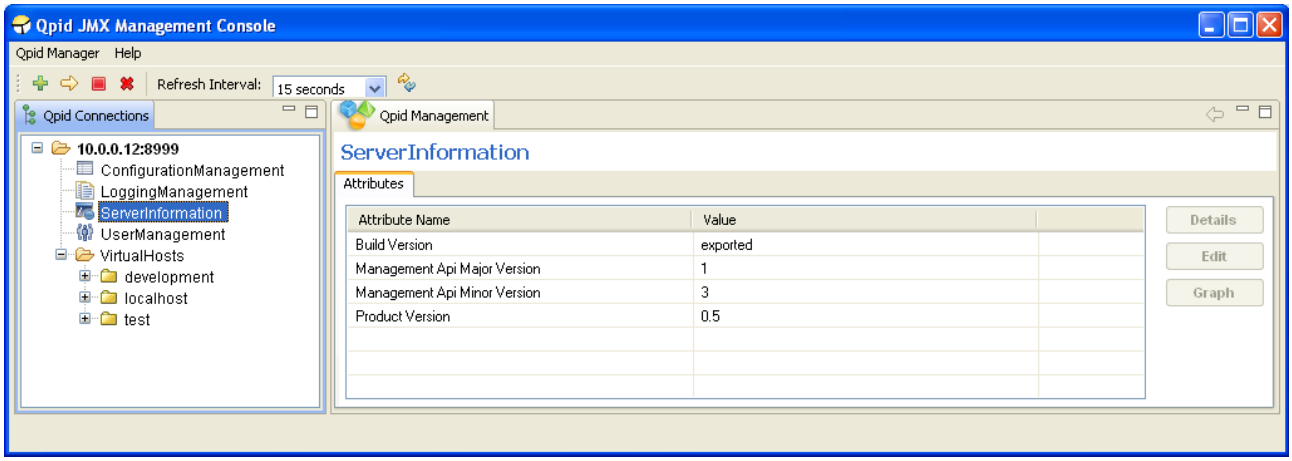
Manipulating the Levels is as on the Runtime Options tab, either double-click an individual Logger entry or select multiple Loggers and use the button to load the dialog to set the new Level.



One issue to note of when reloading the configuration file settings, either automatically using LogWatch or manually, is that any Logger set to a specific Level using the Runtime Options tab that is not defined in the configuration file will maintain that Level when the configuration file is reloaded. In other words, if a Logger is defined in the configuration file, then the configuration file will take precedence at reload, otherwise the Runtime options take precedence.

This situation will be immediately obvious by examining the Runtime Options tab to see the effective Level of each Logger – unless it has been altered with the RuntimeOptions or specifically set in the configuration file, a Logger Level should match that of its parent. In the latest versions of the LoggingManagement MBean, it is possible to use the RuntimeOptions to restore a child logger to inheriting from its parent by setting it with an INHERITED level that removes any previously set Level of its own.

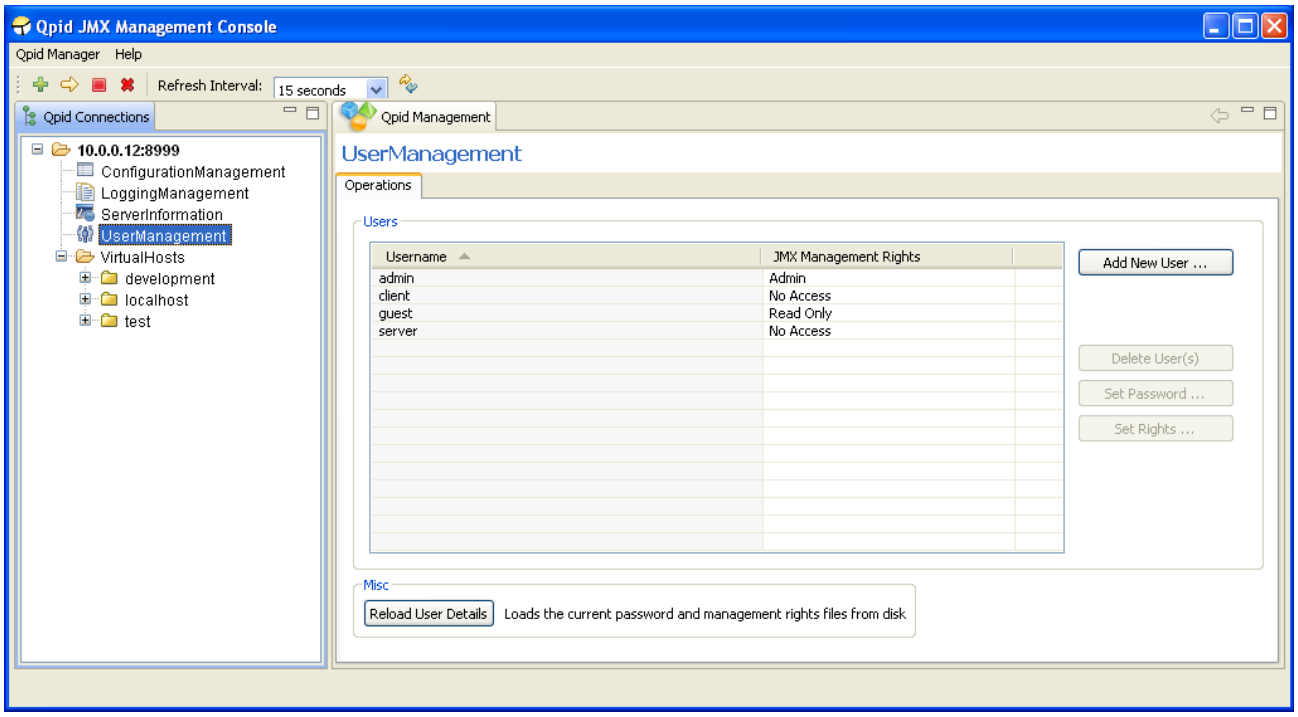
ServerInformation MBean



The ServerInformation MBean currently only conveys various pieces of version information to allow precise identification of the server version and its management capabilities. In future it is likely to convey additional server-wide details and/or functionality.

UserManagement MBean

The UserManagement MBean is accessible by admin level users, and allows manipulation of existing user accounts and creation of new user accounts.



To add a new user, press the *Add New User* button, which will load the dialog shown below.

Here you may enter the new users Username, Password, and select their JMX Management Rights. This controls whether or not they have access to the management interface, and if so what capabilities are accessible. *Read Only* access allows undertaking any operations that do not alter the server state, such as viewing messages. *Read + Write* access allows use of all operations which are not deemed admin-only (such as those in the UserManagement MBean itself). *Admin* access allows a user to utilize any operation, and view the admin-only MBeans (currently these are ConfigurationManagement, LoggingManagement, and UserManagement).

One or more users at a time may be deleted by selecting them in the table and clicking the *Delete User(s)* button. The console will then prompt for confirmation before undertaking the removals.

Similarly, the access rights for one or more users may be updated by selecting them in the table and clicking the *Set Rights* button. The console will then display a dialog enabling selection of the new access level and confirmation to undertake the update.

An individual user password may be updated by selecting the user in the table in and clicking the *Set Password* button. The console will then display a dialog enabling input of the new password and confirmation to undertake the update.

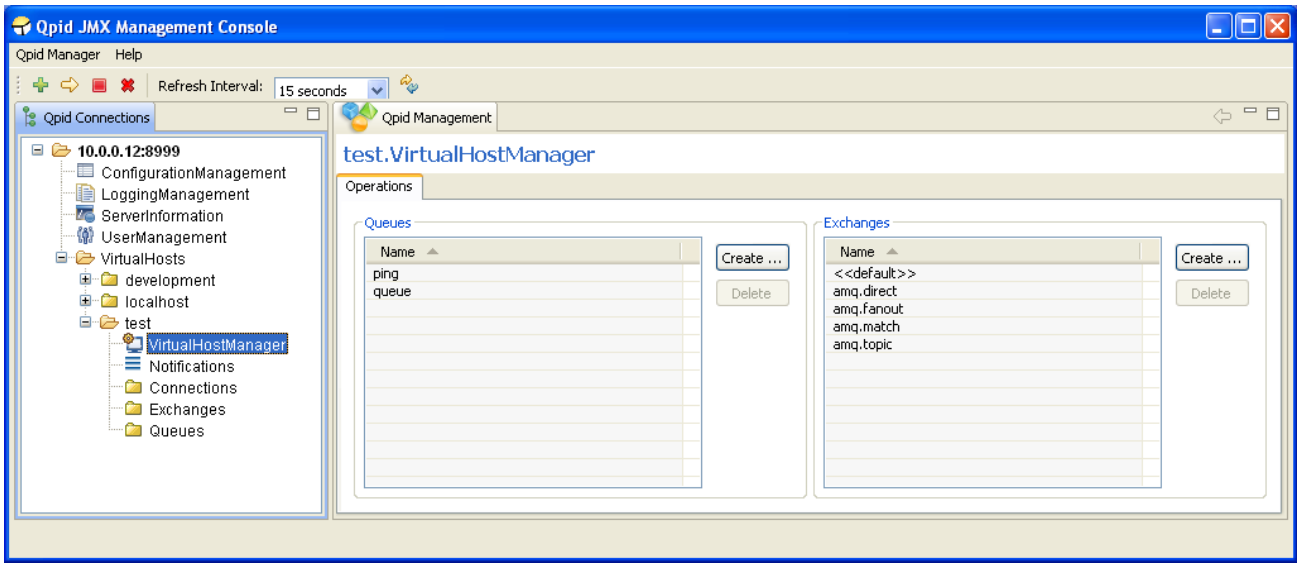
The server caches the user details in memory to aid performance. If may sometimes be necessary to externally modify the password and access right files on disk. In order for these changes to be known to the server without a restart, it must be instructed to reload the file contents. This can be done using the provided *Reload User Details* button (on older servers, only the management rights file is reloaded, on newer servers both files are. The description on screen will indicate the behaviour). After pressing this button the console will seek confirmation before proceeding.

VirtualHostManager MBean

Each VirtualHost in the server has an associated VirtualHostManager MBean. This allows viewing, creation, and deletion of Queues and Exchanges within the VirtualHost.

Clicking the *Create* button in the Queue section will open a dialog allowing specification of the Name, Owner (optional), and durability properties of the new Queue, and confirmation of the operation.

One or more Queues may be deleted by selecting them in the table and clicking the *Delete* button. This will unregister the Queue bindings, remove the subscriptions and delete the Queue(s). The console will prompt for confirmation before undertaking the operation.



Clicking the *Create* button in the Exchange section will open a dialog allowing specification of the Name, Type, and Durable attributes of the new Exchange, and confirmation of the operation.

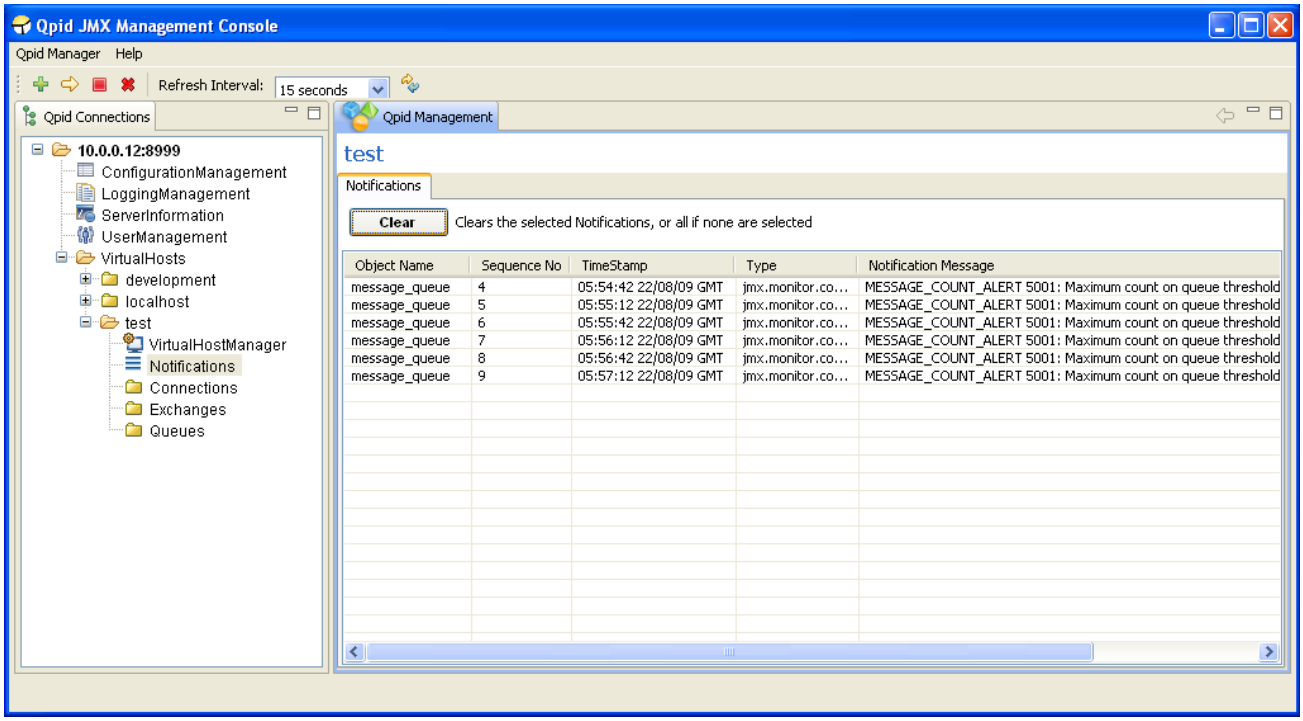
One or more Exchanges may be deleted by selecting them in the table and clicking the *Delete* button. This will unregister all the related channels and Queue bindings then delete the Exchange(s). The console will prompt for confirmation before undertaking the operation.

Double-clicking on a particular Queue or Exchange name in the tables will open the MBean representing it.

Notifications

MBeans on the server can potentially send Notifications that users may subscribe to. When managing an individual MBean that offers Notifications types for subscription, the console supplies a Notifications tab to allow (un)subscription to the Notifications if desired and viewing any that are received following subscription.

In order to provide quicker access to/awareness of any received Notifications, each VirtualHost area in the connection tree has a Notifications area that aggregates all received Notifications for MBeans in that VirtualHost. An example of this can be seen in the figure below.

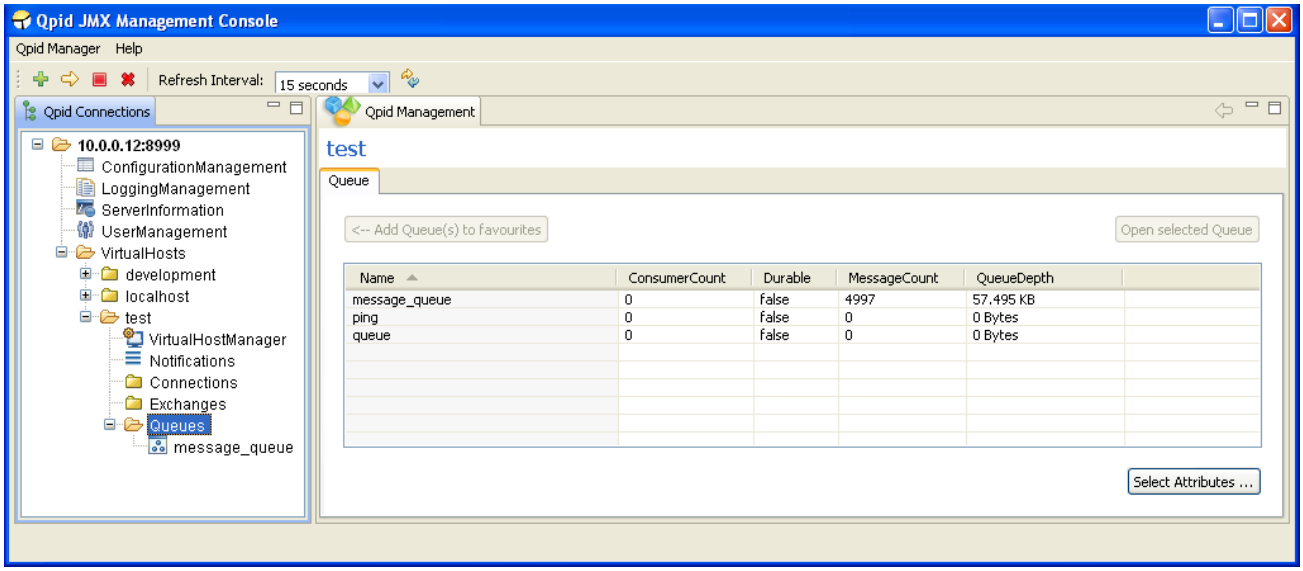


All received Notifications will be displayed until such time as the user removes them, either in this aggregated view, or in the Notifications area of the individual MBean that generated the Notification.

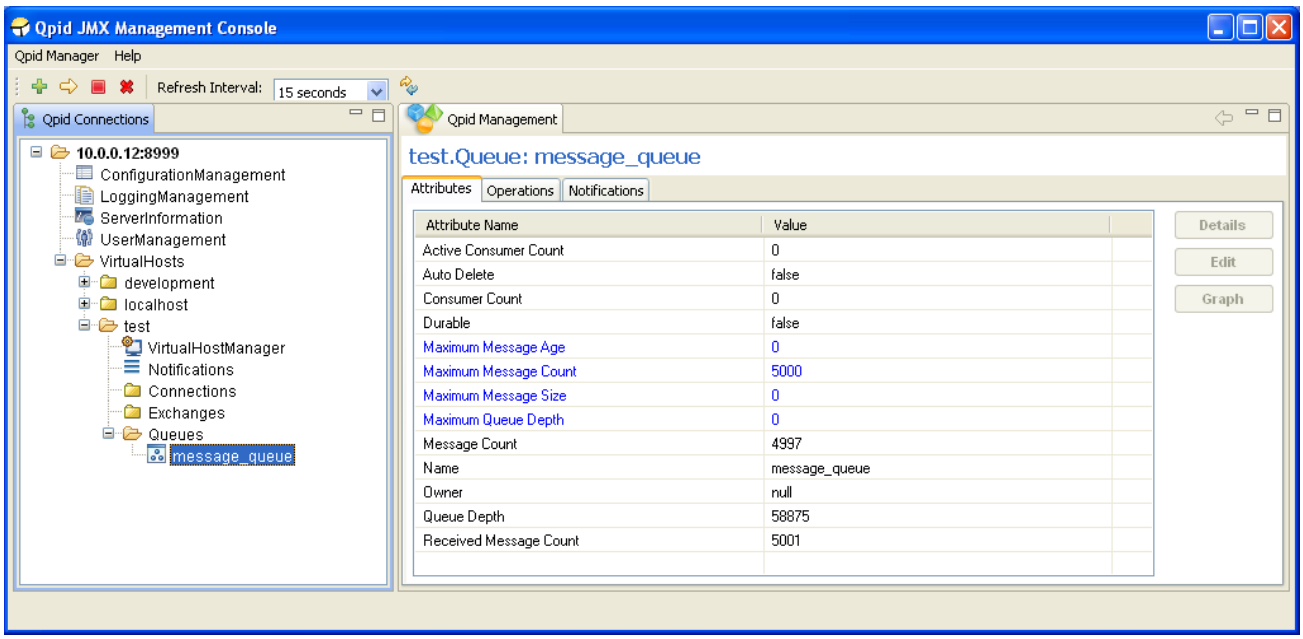
They may be cleared selectively or all at once. To clear particular Notifications, they should be selected in the table before pressing the *Clear* button. To clear all Notifications, simply press the *Clear* button without anything selected in the table, at which point the console will request confirmation of this clear-all action.

Managing Queues

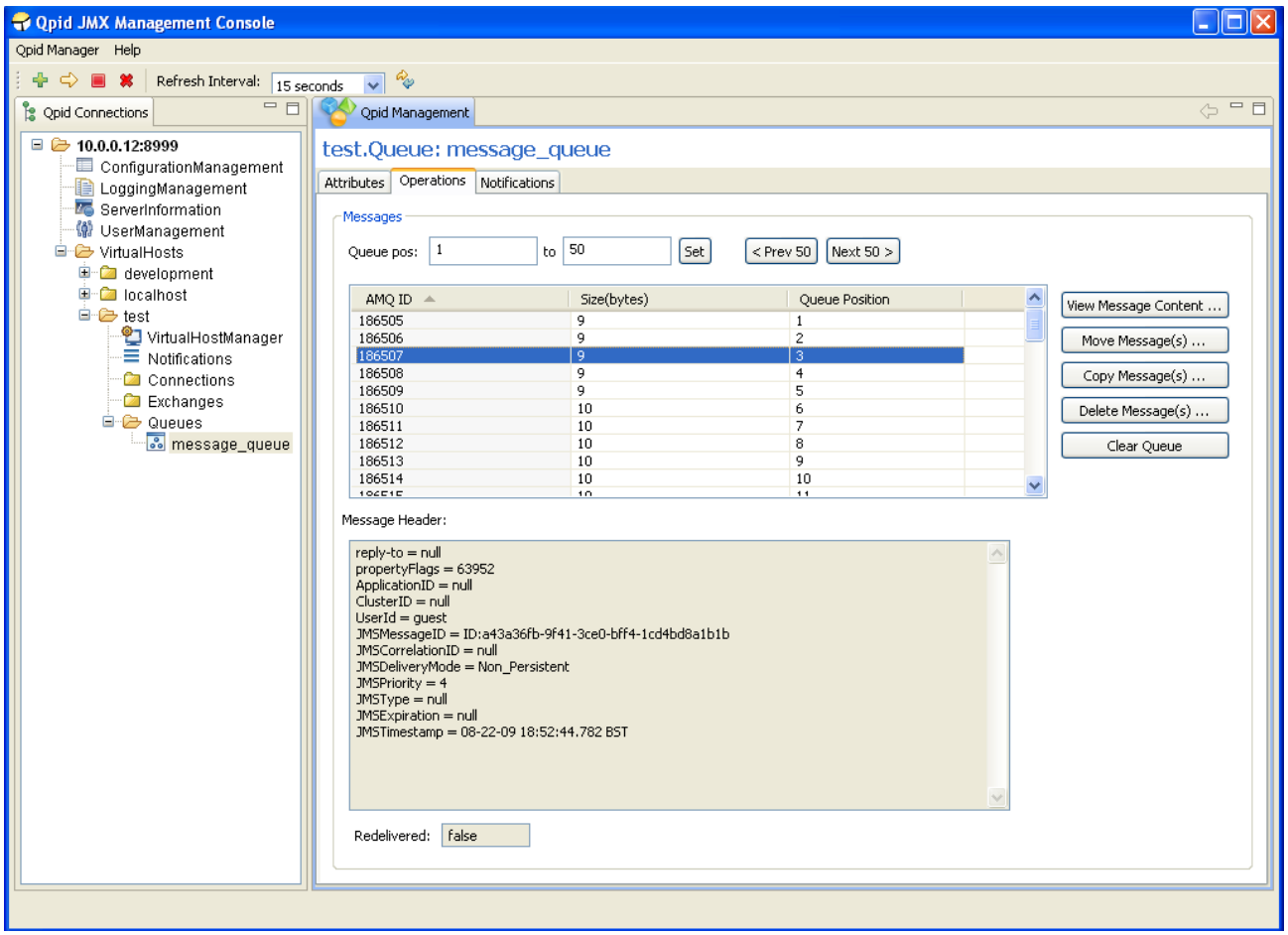
As mentioned in earlier discussion of Navigation, Queue MBeans can be opened either by double clicking an entry in the Queues selection area, or adding a queue to the tree as a favourite and clicking on its tree node. Unique to the Queue selection screen is the ability to view additional attributes beyond just that of the Queue Name. This is helpful for determining which Queues satisfy a particular condition, e.g. having <X> messages on the queue. The example below shows the selection view with additional attributes *Consumer Count, Durable, MessageCount, and QueueDepth* (selected using the *Select Attributes* button at the bottom right corner of the table).



Upon opening a Queue MBean, the Attributes tab is displayed, as shown below. This allows viewing the value all attributes, editing those which are writable values (highlighted in blue) if the users management permissions allow, viewing descriptions of their purpose, and graphing certain numerical attribute values as they change over time.



The next tab contains the operations that can be performed on the queue. The main table serves as a means of viewing the messages on the queue, and later for selecting specific messages to operate upon. It is possible to view any desired range of messages on the queue by specifying the visible range using the fields at the top and pressing the *Set* button. Next to this there are helper buttons to enable faster browsing through the messages on the queue; these allow moving forward and back by whatever number of messages is made visible by the viewing range set. The Queue Position column indicates the position of each message on the queue, but is only present when connected to newer servers as older versions cannot provide the necessary information to show this (unless only a single message position is requested).



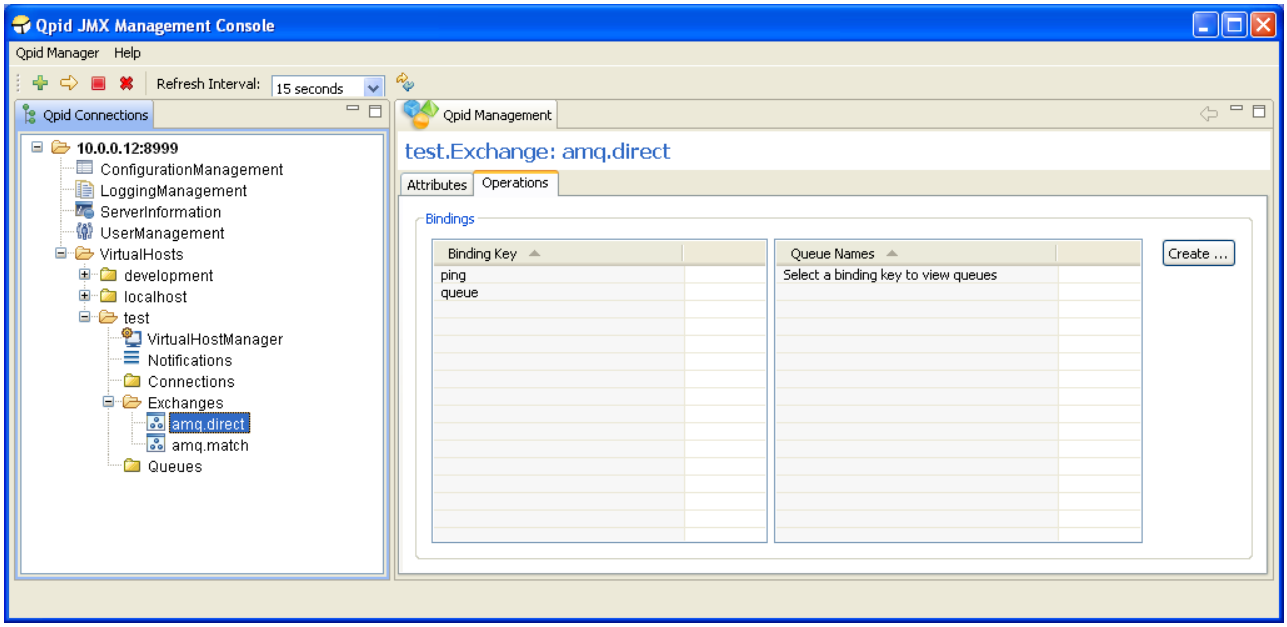
Upon selecting a message in the table, its header properties and redelivery status are updated in the area below the table. Double clicking a message in the table (or using the *View Message Content* button to its right) will open a dialog window displaying the contents of the message.

One or more messages can be selected in the table and moved to another queue in the VirtualHost by using the *Move Message(s)* button, which opens a dialog to enable selection of the destination and confirmation of the operation. Newer servers support the ability to similarly copy the selected messages to another queue in a similar fashion, or delete the selected messages from the queue after prompting for confirmation.

Finally, all messages (that have not been acquired by consumers) on the queue can be deleted using the *Clear Queue* button, which will generate a prompt for confirmation. On newer servers, the status bar at the lower left of the application will report the number of messages actually removed.

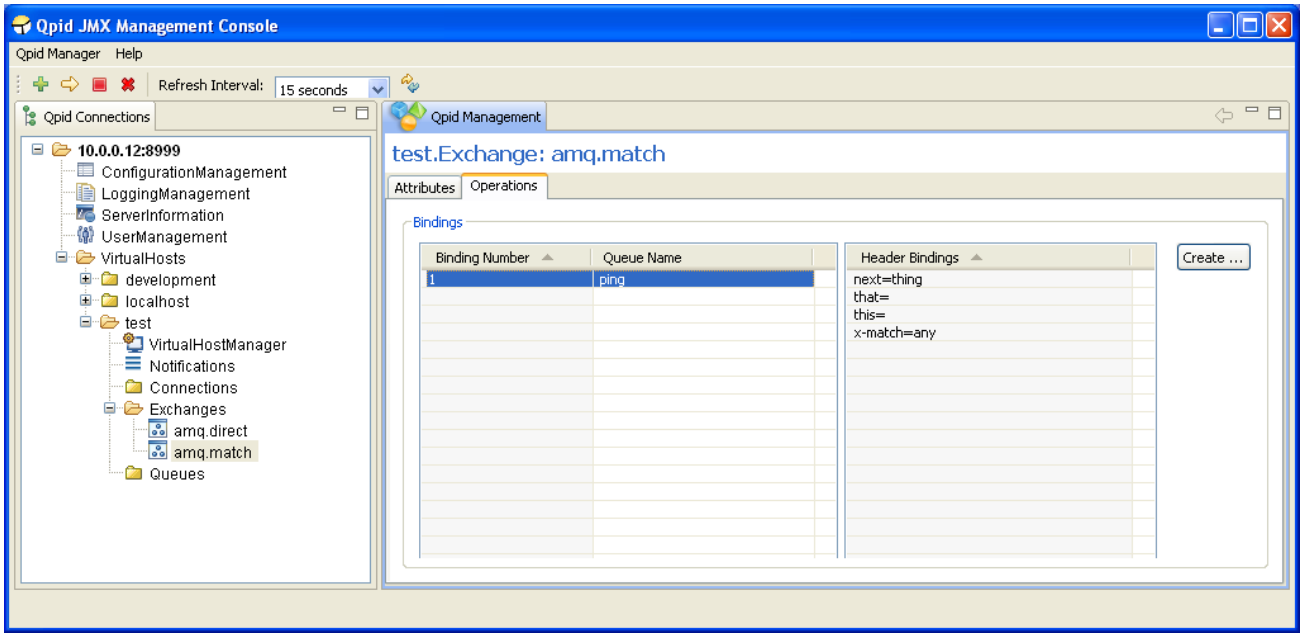
Managing Exchanges

Exchange MBeans are opened for management operations in similar fashion as described for Queues, again showing an Attributes tab initially, with the Operations tab next:



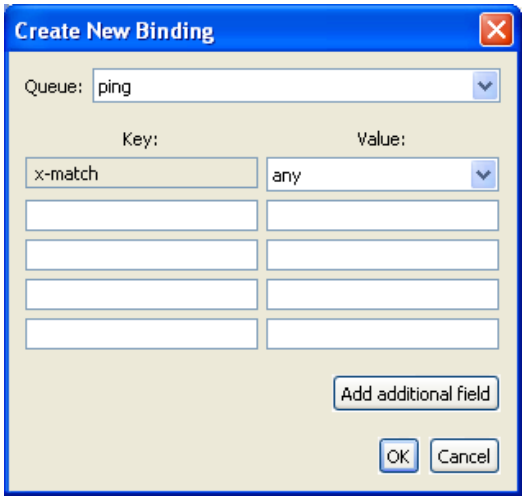
Of the four default Exchange Types *(direct, fanout, headers, and topic)* all but *headers* have their bindings presented in the format shown above. The left table provides the binding/routing keys present in the exchange. Selecting one of these entries in the table prompts the right table to display all the queues associated with this key. Pressing the *Create* button opens a dialog allowing association of an existing queue with the entered Binding.

The *headers* Exchange type (default instantiation *amq.match or amq.headers*) is presented as below:



In the previous figure, the left table indicates the binding number, and the Queue associated with the binding. Selecting one of these entries in the table prompts the right table to display the header values that control when the binding matches an incoming message.

Pressing the *Create* button when managing a *headers* Exchange opens a dialog allowing creation of a new binding, associating an existing Queue with a particular set of header keys and values. The *x-match* key is required, and instructs the server whether to match the binding with incoming messages based on ANY or ALL of the further key-value pairs entered. If it is desired to enter more than 4 pairs, you may press the *Add additional field* button to create a new row as many times as is required.

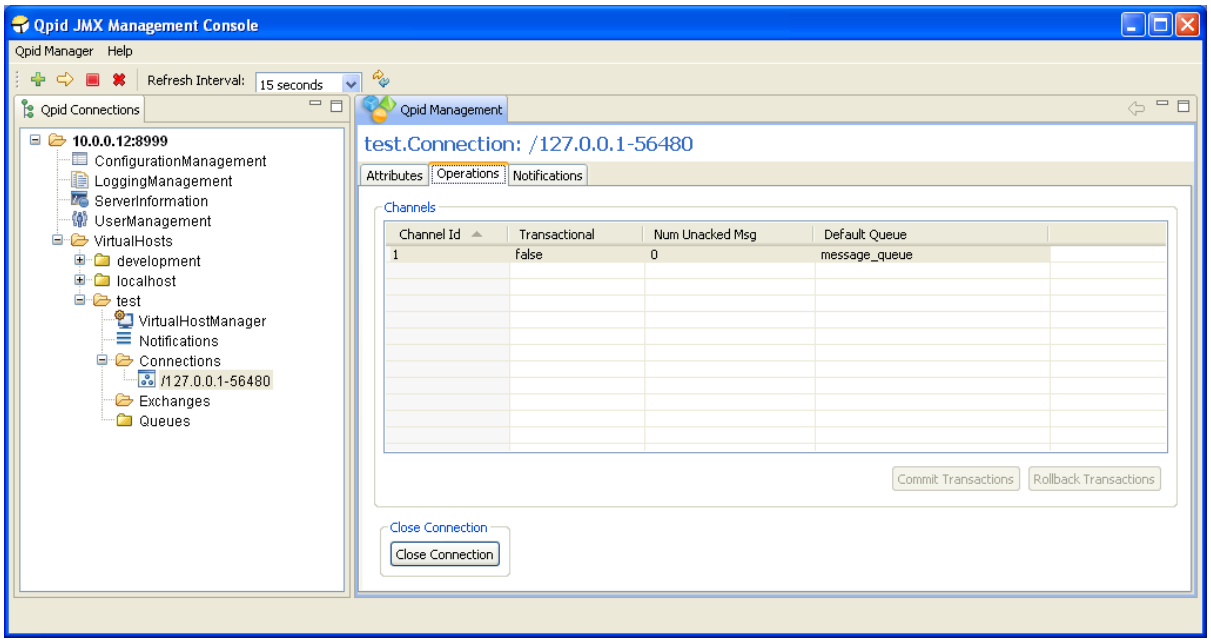


When managing a *headers* Exchange, double clicking an entry in the left-hand table will open the MBean for the Queue specified in the binding properties.

When managing another Exchange Type, double clicking the Queue Name in the right-hand table will open the MBean of the Queue specified.

Managing Connections

Exchange MBeans are opened for management operations in similar fashion as described for Queues, again showing an Attributes tab initially, with the Operations tab next, and finally a Notifications tab allowing subscription and viewing of Notifications. The Operations tab can be seen in the figure below.



The main table shows the properties of all the Channels that are present on the Connection, including whether they are *Transactional*, the *Number of Unacked Messages* on them, and the *Default Queue* if there is one (or *null* if there is not).

The main operations supported on a connection are Commiting and Rolling Back of Transactions on a particular Channel, if the Channel is Transactional. This can be done by selecting a particular Channel in the table and pressing the *Commit Transactions* or *Rollback Transactions* buttons at the lower right corner of the table, at which point the console will prompt for confirmation of the action. These buttons are only active when the selected Channel in the table is Transactional.

The final operation supported is closing the Connection. After pressing the *Close Connection* button, the console will prompt for confirmation of the action. If this is carried out, the MBean for the Connection being managed will be removed from the server. The console will be notified of this by the server and display an information dialog to that effect, as it would if any other MBean were to be unregistered whilst being viewed.

Double clicking a row in the table will open the MBean of the associated *Default Queue* if there is one.