QuartzDesk JVM Agent Installation and Upgrade Guide for Oracle WebLogic AS 12.2.x (12cR2) and 14.1.x (14c)

QuartzDesk Version: 5.x

May 7, 2024

Table of Contents

1. Purpose

This document describes the installation and upgrade process for QuartzDesk JVM Agent 5.x on Oracle WebLogic Application Server 12.2.x (12cR2) and 14.1.x (14c).

QuartzDesk JVM Agent is a Java Virtual Machine (JVM) plugin that must be installed in all JVMs powering Quartz-based applications managed by QuartzDesk Standard or Enterprise editions.

If you experience any problems installing or upgrading QuartzDesk JVM Agent, please let us know at support@quartzdesk.com.

2. Definitions

The following table lists all acronyms and shortcuts used throughout this document.

Acronym / Shortcut	Definition
AS	Application Server.
EAR	Enterprise Application Archive. A file with `.ear` extension.
JAR	Java Application Archive. A file with `.jar` extension.
JVM	Java Virtual Machine.
WLAC	WebLogic Administrative Console.
WLAS	WebLogic Application Server.
WAR	Web Application Archive. A file with `.war` extension.

The following table lists all locations and properties used throughout this document.

Location / Property	Example	Description
AGENT_WORK_DIR	/var/quartzdesk-agent.work	QuartzDesk JVM Agent work directory.
DB_HOST	localhost	QuartzDesk JVM Agent database server host.
DB_PORT	5432	QuartzDesk JVM Agent database server port.
DB_NAME	quartzdesk_agent	QuartzDesk JVM Agent database name.
DB_SCHEMA	quartzdesk_agent	QuartzDesk JVM Agent database schema.
DB_USER	quartzdesk_agent	QuartzDesk JVM Agent database user.
DB_PASSWORD	quartzdesk_agent	QuartzDesk JVM Agent database user password.
JAVA_HOME	/usr/local/java	Java home directory.
MW_HOME	/opt/oracle/middleware	Oracle Middleware installation directory.
WL_DOMAIN	domain1	
WL_DOMAIN_HOME	/opt/oracle/user_projects/domain1	WebLogic Application Server domain directory.
WL_SERVER	MyServer	WebLogic Application Server name.

3. Requirements

3.1 Software Requirements

3.1.1 Operating System

Windows 7-11.
Linux (any distribution) with kernel 2.6.x and above.
Solaris 11.x and above.

3.1.2 JVM

Oracle JDK 8–12.
IBM JDK 8.
OpenJDK 8–22.

3.1.3 Application Server

Oracle WebLogic Application Server 12.2.x (12cR2).
Oracle WebLogic Application Server 14.1.x (14c).

3.1.4 Database

Database	Minimum Version
DB2	10.1
H2	1.3.174
Microsoft SQL Server	2008 R2 SP1
MySQL	5.6.4
Oracle	10.2 (10g R2)
PostgreSQL	9.1

3.1.5 Database JDBC Driver

Database	JDBC Driver
DB2	IBM DB2 JDBC 4.0 driver available at http://www-01.ibm.com/support/docview.wss?uid=swg21363866.
H2	Database engine including the JDBC driver is available at http://www.h2database.com.
Microsoft SQL Server	Microsoft JDBC driver 4.0 for SQL Server available at http://msdn.microsoft.com/en-us/sqlserver/aa937724.aspx. We strongly advise against using the alternative JTDS JDBC driver, because it currently does not support the datetime2 data type. As a result, all datetime values written by the QuartzDesk application would end up rounded up, or down. For datetime data type rounding details, please refer to http://msdn.microsoft.com/en-us/library/ms187819.aspx.

MySQL	Connector/J JDBC driver available at http://dev.mysql.com/downloads/connector/j/.
Oracle	Oracle JDBC driver available at http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html. For a comprehensive overview of JDBC driver versions vs. supported database versions, please refer to http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-faq-090281.html#01_02.
PostgreSQL	JDBC4 PostgreSQL driver available at http://jdbc.postgresql.org/.

3.1.6 QuartzDesk JVM Agent Library

To install QuartzDesk JVM Agent, you need to obtain the QuartzDesk JVM Agent JAR. The latest version can be downloaded at www.quartzdesk.com (click Downloads → Latest Release → View files → quartzdesk-agent-x.y.z.jar).

3.1.7 QuartzDesk Public API Library

QuartzDesk JVM Agent requires all applications with embedded Quartz schedulers deployed on the given JVM to have the QuartzDesk Public API Library on their class path. The latest version can be downloaded at www.quartzdesk.com (click Downloads → Latest Release → View files → quartzdesk-api-x.y.z.jar).

The QuartzDesk Public API library is also available in the Maven Central repository – see http://search.maven.org/#search|ga|1|quartzdesk-api.

3.2 Hardware Requirements

QuartzDesk JVM Agent runs on any physical or virtualized hardware that supports the above software requirements.

4. Installation

4.1 Database

Create a new database user named `quartzdesk_agent` (`DB_USER`) with an arbitrary password (`DB_PASSWORD`).

Create a new QuartzDesk JVM Agent database named `quartzdesk_agent`[1] (`DB_NAME`) owned by the `DB_USER`.

If the database supports database schemas, create a new schema named `quartzdesk_agent` (`DB_SCHEMA`). The schema must be owned by the `DB_USER`. Make the created DB_SCHEMA the default schema of the `DB_USER` and/or add the schema to the `DB_USER`'s schema search path.

Please contact your DBA, or refer to the database engine documentation for instructions on how to complete the above database-specific tasks.

Please note that you do not have to create any other database objects (tables, keys, indices etc.) in the QuartzDesk JVM Agent database. These objects will be automatically created by QuartzDesk JVM Agent during its first run.

4.2 JVM Agent Work Directory

Create the QuartzDesk JVM Agent work directory (AGENT_WORK_DIR) anywhere on the local file system. The directory must be readable and writeable by the user the WLAS process runs under.

Copy your QuartzDesk license key file (`license.key`) to AGENT_WORK_DIR.

You can obtain a free 30-day trial license key at www.quartzdesk.com (open the Try / Purchase menu).

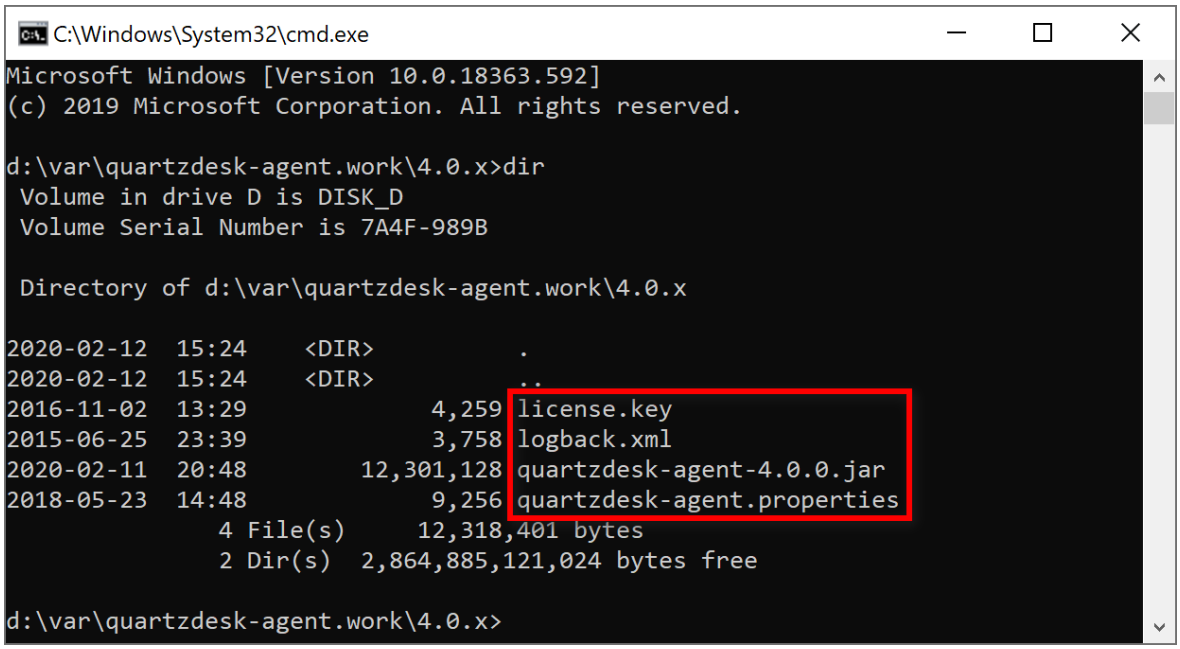
Copy the QuartzDesk JVM Agent JAR file (`quartzdesk-agent-x.y.z.jar`) to AGENT_WORK_DIR.

Open the QuartzDesk JVM Agent JAR file and copy all files from the `extras/work` directory into AGENT_WORK_DIR.

If you cannot open the JAR file directly, rename it to *.zip and then open it. Do not forget to rename the file back to *.jar once you have extracted the required files.

In the following figure you can see an example of a QuartzDesk JVM Agent work directory correctly set up on a Microsoft Windows machine.

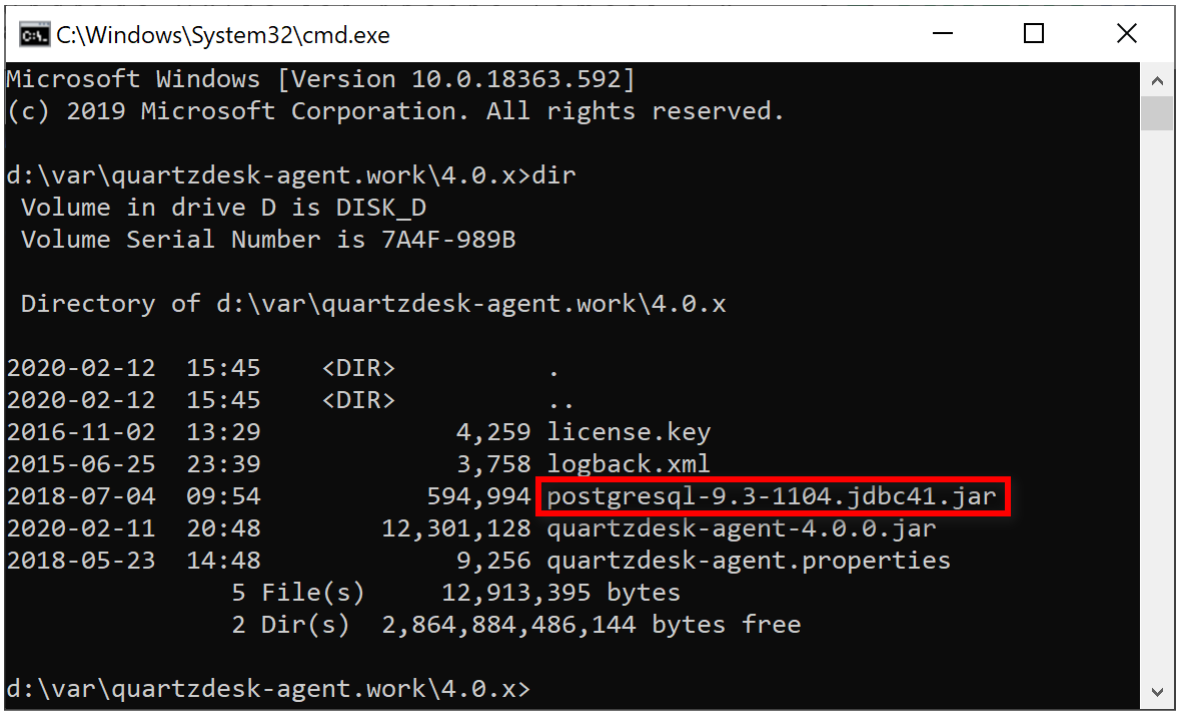
[1] If you use DB2, the database name length is restricted to the maximum of 8 characters. Please adjust the database name accordingly (e.g. qdagent).



4.3 JDBC Driver

Download and install the JDBC driver for the created database. For a list of supported JDBC drivers please refer to chapter 3.1.5.

Copy the JDBC driver JAR file to AGENT_WORK_DIR. Make sure the JAR file is readable by the user the WLAS process runs under.



4.4 JVM Agent Configuration

Open the QuartzDesk JVM Agent configuration file `AGENT_WORK_DIR/quartzdesk-agent.properties`.

Based on the type and version of the database created in step 4.1, change the value of the `db.profile` configuration property according to the following table.

Database	Database Version	db.profile Value
DB2	>= 10.0	db2
H2	>= 1.3.170	h2
Microsoft SQL Server	>= 2008	mssql
MySQL	>= 5.6	mysql
MySQL (Inno)	>= 5.6	mysql_inno
Oracle	== 8i	oracle8
Oracle	>= 9i	oracle9
PostgreSQL	== 8.1	postgres81
PostgreSQL	>= 8.2	postgres82

Uncomment the Agent JDBC pool configuration section based on the QuartzDesk JVM Agent database type. Make sure the JDBC pool configuration sections for other database types are commented out (prefixed with '#'). The default sample `quartzdesk-agent.properties` file assumes the use of a PostgreSQL database.

Adjust values of the JDBC pool configuration parameters to match your database configuration. You will typically need to change the default host value (localhost) in the `jdbc.url` parameter to point to DB_HOST. Please refer to the JDBC driver manual for a description of the JDBC URL format and related details.

Set the value of the `jdbc.pool.maxTotal` JDBC pool configuration parameter to be 10-20% higher than the maximum number of **concurrently executing** Quartz jobs on the JVM QuartzDesk JVM Agent will be installed on.

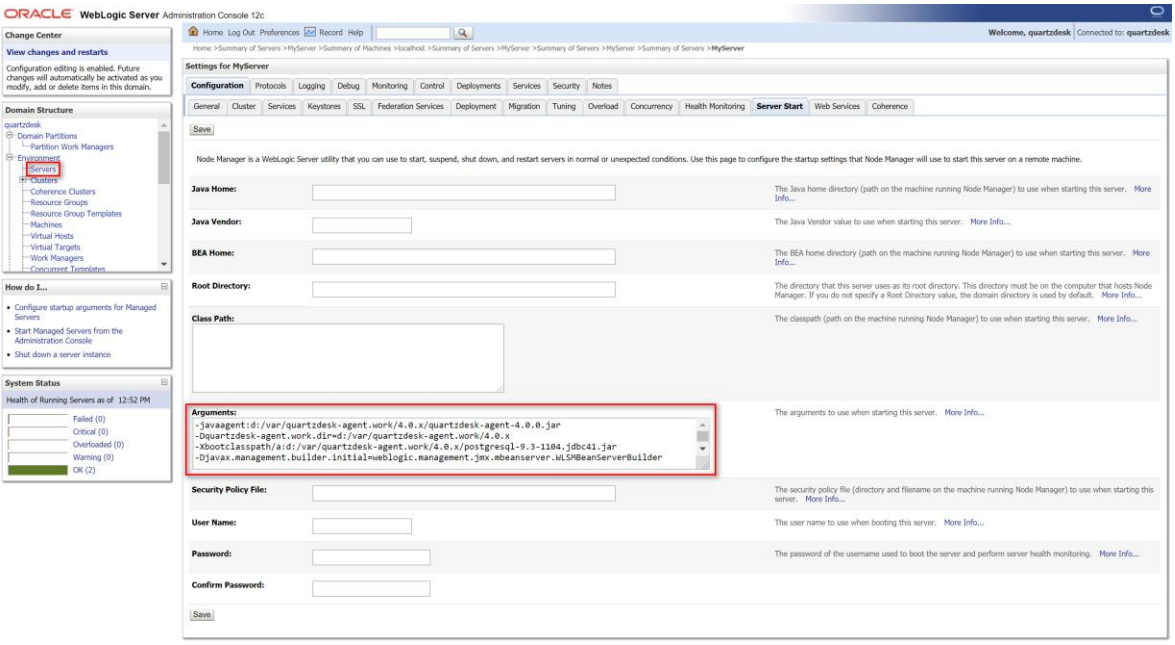
To adjust QuartzDesk JVM Agent logging parameters, edit the `AGENT_WORK_DIR/logback.xml` configuration file. The default sample `logback.xml` configuration file creates the QuartzDesk JVM Agent log under the `AGENT_WORK_DIR/logs` directory that is automatically created when QuartzDesk JVM Agent starts. Please refer to the Logback Manual for Logback configuration details.

4.5 Install JVM Agent

In WLAC edit server start configuration (WL_DOMAIN → Environment → Servers → WL_SERVER → Configuration → Server Start) and in the Arguments field add the following JVM arguments:

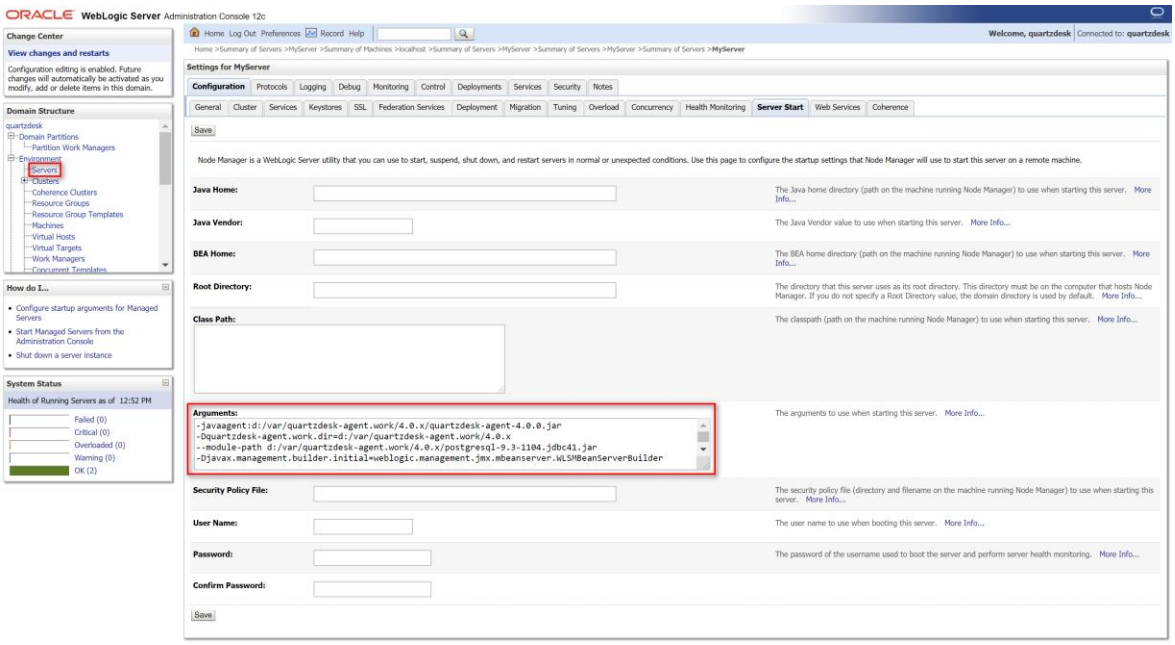
Java 8

```
-javaagent:<AGENT_WORK_DIR>/quartzdesk-agent-x.y.z.jar
-Dquartzdesk-agent.work.dir=<AGENT_WORK_DIR>
-Xbootclasspath/a:<JDBC_DRIVER_JAR_FILE_PATH>
-
Djavax.management.builder.initial=weblogic.management.jmx.mbeanserver.WLSMBeanServerB
uilder
```



Java 9–13

```
-javaagent:<AGENT_WORK_DIR>/quartzdesk-agent-x.y.z.jar
-Dquartzdesk-agent.work.dir=<AGENT_WORK_DIR>
--module-path <JDBC_DRIVER_JAR_FILE_PATH>
-
Djavax.management.builder.initial=weblogic.management.jmx.mbeanserver.WLSMBeanServerB
uilder
```



Click the Save button.

Restart the updated server (WL_SERVER).

4.6 Install Public API Library

QuartzDesk Public API Library[2] works as an interface between the Quartz library[3] used by a Java application and QuartzDesk JVM Agent. **QuartzDesk Public API Library must be loaded by the same Java class loader that loads the Quartz library.**

In WLAS, there are two typical cases how the Quartz library is deployed.

(1) Quartz library is embedded in the application, typically in its `WEB-INF/lib` folder. In this case, the QuartzDesk Public API Library JAR must be copied to this folder.

Please note that the QuartzDesk Public API Library JAR is available in the Maven Central repository and if you add it as a runtime dependency to the application's POM, it can be automatically copied to the application's `WEB-INF/lib` folder by Maven.

(2) Quartz library is placed in the WLAS shared lib folder located at `WL_DOMAIN_HOME/lib`, and as such it is shared by all applications deployed on the WLAS instance. In this case, the QuartzDesk Public API Library JAR must be copied to the `WL_DOMAIN_HOME/lib` folder.

Please note that the installation of QuartzDesk Public API Library requires no application code changes.

4.7 Stop WebLogic AS

Stop WLAS by executing the following command:

Windows
```
WL_DOMAIN_HOME\bin\stopManagedWebLogic.bat WS_NAME
```

Unix / Linux
```
WL_DOMAIN_HOME/bin/stopManagedWebLogic.sh WS_NAME
```

Wait for the action to complete.

4.8 Start WebLogic AS

Start WLAS by executing the following command:

Windows
```
WL_DOMAIN_HOME\bin\startManagedWebLogic.bat WS_NAME
```

Unix / Linux

[2] JAR file name: `quartzdesk-api-<version>.jar`
[3] JAR file name name: `quartz-<version>.jar` or `quartz-all-<version>.jar`

`WL_DOMAIN_HOME/bin/startManagedWebLogic.sh WS_NAME`
Wait for the action to complete.

Check the WLAS logs under `WL_DOMAIN_HOME/servers/WL_SERVER/logs` for errors.

Check the QuartzDesk JVM Agent logs (in `AGENT_WORK_DIR/logs` directory) for errors and verify the version number of the installed QuartzDesk JVM Agent.

Verify that all applications deployed on WLAS work as expected.

5. Upgrading

5.1 Stop WebLogic AS

Stop WLAS by following the steps outlined in 4.7.

5.2 Backup

Backup your QuartzDesk JVM Agent database. We recommend performing a **full database backup**.

Backup the contents of the QuartzDesk JVM Agent work directory.

Store the backups in a safe place so that you can restore the original QuartzDesk JVM Agent version if the need arises.

5.3 Upgrade JVM Agent

Delete the old QuartzDesk JVM Agent JAR file in AGENT_WORK_DIR. Copy the new `quartzdesk-agent-x.y.z.jar` to AGENT_WORK_DIR.

Rename the `AGENT_WORK_DIR/quartzdesk-agent.properties` configuration file to `quartzdesk-agent.properties.old`.

Open the QuartzDesk JVM Agent JAR file (`quartzdesk-agent-x.y.z.jar`) and copy the `extras/work/quartzdesk-agent.properties` configuration file to `AGENT_WORK_DIR`.

If you cannot open the JAR file directly, rename it to *.zip and then open it. Do not forget to rename the file back to *.jar once you have extracted the required files.

Adjust the values of the configuration properties in the new configuration file `AGENT_WORK_DIR/quartzdesk-agent.properties` to match your system setup. You can use the old configuration file as a reference.

Please refer to 4.4 for a description of the configuration parameters that you need to adjust.

5.4 Upgrade Public API Library

The steps necessary to upgrade this library depend on the way it has been deployed. Please refer to 4.6 for details.

5.5 Start WebLogic AS

Start WLAS by following the steps outlined in 4.8.

6. Version 2.x to 5.x Migration Notes

6.1 Configuration Properties Changes

The following two `quartzdesk-agent.properties` configuration properties have been deprecated in QuartzDesk JVM Agent 3.x and will be removed in the future. Make sure your `quartzdesk-agent.properties` file uses the new property names.

Deprecated Configuration Property Name	New Configuration Property Name
`jdbc.pool.maxActive`	`jdbc.pool.maxTotal`
`jdbc.pool.maxWait`	`jdbc.pool.maxWaitMillis`

6.2 Upgrade Steps

To upgrade QuartzDesk JVM Agent 2.x to 5.x, apply upgrade steps outlined in **Error! Reference source not found.**.

7. Version 3.x to 5.x Migration Notes

7.1 Configuration Properties Changes

The following two `quartzdesk-agent.properties` configuration properties have been deprecated in QuartzDesk JVM Agent 3.x and will be removed in in the future. Make sure your `quartzdesk-agent.properties` file uses the new property names.

Removed Configuration Property Name	New Configuration Property Name
`jdbc.pool.maxActive`	`jdbc.pool.maxTotal`
`jdbc.pool.maxWait`	`jdbc.pool.maxWaitMillis`

7.2 Upgrade Steps

To upgrade QuartzDesk JVM Agent 3.x to 5.x, apply upgrade steps outlined in **Error! Reference source not found.**.

8. Version 4.x to 5.x Migration Notes

8.1 Configuration Properties Changes

No changes in `quartzdesk-agent.properties` are required.

8.2 Upgrade Steps

To upgrade QuartzDesk JVM Agent 4.x to 5.x, apply upgrade steps outlined in **Error! Reference source not found.**.

9. Cluster Deployment Notes

When configuring QuartzDesk JVM Agent in a WebLogic cluster you need to follow the configuration steps described in preceding chapters. In addition to these, there are several extra configuration steps that must be performed in cluster deployments.

9.1 Shared Work Directory

We recommend that you put the QuartzDesk JVM Agent work directory, described in chapter 4.2, on a shared drive and make this work directory available to all WebLogic cluster members.

9.2 Logging Configuration

If you set up your cluster to use a shared QuartzDesk JVM Agent work directory, as described in the previous chapter, you will need to edit the QuartzDesk JVM Agent logging configuration file `AGENT_WORK_DIR/logback.xml` and decide where QuartzDesk JVM Agent instances running on individual cluster members should log. There are two options:

1) Logging into the same (shared) log files.
2) Logging into separate log files.

QuartzDesk JVM Agent uses two log files – quartzdesk.log and quartzdesk-trace.log that are stored in `AGENT_WORK_DIR/logs` directory. The following chapters discuss these two options.

9.2.1 Using Shared Log Files

In order to make individual QuartzDesk JVM Agent instances log into the same log files, you must enable the prudent mode on both file appenders used in the `AGENT_WORK_DIR/logback.xml` configuration file:

```
...

<appender name="FILE"
class="ext.ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-agent.log</file>
  <append>true</append>
  <prudent>true</prudent>
  ...
</appender>


<appender name="TRACE_FILE"
class="ext.ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-agent-trace.log</file>
  <append>true</append>
  <prudent>true</prudent>
  ...

  <!--
    We must use the TimeBasedRollingPolicy because the
    FixedWindowRollingPolicy is not supported in prudent mode!
  -->
  <rollingPolicy
class="ext.ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <!-- daily rollover -->
    <fileNamePattern>${logs.dir}/quartzdesk-agent-trace.log.%d{yyyy-MM-
dd}</fileNamePattern>
    <!-- keep 10 days' worth of history -->
    <maxHistory>10</maxHistory>
  </rollingPolicy>

  <!--
    The SizeBasedTriggeringPolicy removed because it is used only in
    conjunction with the FixedWindowRollingPolicy.
  -->

  <encoder>
    <charset>UTF-8</charset>
    <pattern>[%date] %.-1level [%thread] [%mdc] [%logger:%line] -
%msg%n</pattern>
  </encoder>
</appender>

...
```

For details on the Logback prudent mode, please refer to http://logback.qos.ch/manual/appenders.html#FileAppender.

Because prudent mode relies on exclusive file locks to manage concurrent access to the log files and these locks can have negative impact on the QuartzDesk JVM Agent's performance, we generally discourage using the prudent mode and shared log files.

9.2.2 Using Separate Log Files

In order to make individual QuartzDesk JVM Agent instances log into separate log files, you can use a JVM system property set on all cluster member JVMs. The value of this property must be unique for all cluster members. The property can then be referred to from the `AGENT_WORK_DIR/logback.xml` logging configuration file.

The following examples assume the use of the `cluster.member.instanceId` JVM system property, but any JVM system property name can be used.

There are two common approaches as to where the separate log files produced by individual QuartzDesk JVM Agent instances are stored:

1) Log files created under a common log root directory.

```
...

<appender name="FILE"
class="ext.ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-agent-${cluster.member.instanceId}.log</file>
  <append>true</append>

  ...

  <rollingPolicy
class="ext.ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <!-- daily rollover -->
    <fileNamePattern>${logs.dir}/quartzdesk-agent-
${cluster.member.instanceId}.log.%d{yyyy-MM-dd}</fileNamePattern>
    <!-- keep 10 days' worth of history -->
    <maxHistory>10</maxHistory>
  </rollingPolicy>

  ...

</appender>


<appender name="TRACE_FILE"
class="ext.ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-agent-${cluster.member.instanceId}-
trace.log</file>
  <append>true</append>

  ...

  <rollingPolicy
class="ext.ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
    <fileNamePattern>${logs.dir}/quartzdesk-agent-
${cluster.member.instanceId}-trace.log.%i</fileNamePattern>
    <minIndex>1</minIndex>
    <maxIndex>5</maxIndex>
  </rollingPolicy>

  ...

</appender>

...
```

2) Log files created in separate (cluster member specific) log root directories.

```
...

<!--
  Logback context property logback.config.dir is set by the
  LogbackInitContextListener to point to the parent directory of the Logback
  configuration file (logback.xml).
-->
<property name="logs.dir" value="${logback.config.dir:-
.}/${cluster.member.instanceId}/logs"/>

...
```

9.3 Installation and Upgrade Roll-Out

As described in chapter 4.1, QuartzDesk JVM Agent automatically creates all required database objects in the configured database upon its first start. Similarly, upon every QuartzDesk JVM Agent upgrade the agent automatically applies required changes to the configured database.

If you have configured multiple QuartzDesk JVM Agents to use the same database, collisions are likely to occur if multiple agents are started concurrently and all attempt to realize the database initialization/upgrade procedure described above. To avoid these collisions, please start a single JVM with the configured QuartzDesk JVM Agent and let the agent apply the database changes. Once the database changes have been successfully applied, it is possible to start the other agents (JVMs).

You can check for the following line in the QuartzDesk JVM Agent log to see if the database has been successfully initialized/upgraded. This log line indicates that the agent has been successfully started at which point all database schema changes have been applied.

```
...

[2017-08-04 13:34:56,215] I [main] [com.quartzdesk.agent.Agent:275] -
Successfully initialized QuartzDesk JVM Agent:
com.quartzdesk.agent.Agent@97e1896 [QuartzDesk JVM Agent v3.0.1], enabled:
true

...
```