QuartzDesk Web Application Installation and Upgrade Guide for Oracle GlassFish AS 5.x

QuartzDesk Version: 4.x

March 3, 2020

Table of Contents

1. Purpose

This document describes the installation and upgrade process for QuartzDesk Web Application 4.x on Oracle GlassFish Application Server 5.x.

If you experience any problems installing or upgrading QuartzDesk Web Application, 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.
GAC	GlassFish Administrative Console.
GAS	GlassFish 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
DB_HOST	localhost	QuartzDesk Web Application database server host.
DB_PORT	5432	QuartzDesk Web Application database server port.
DB_NAME	quartzdesk	QuartzDesk Web Application database name.
DB_SCHEMA	quartzdesk	QuartzDesk Web Application database schema.
DB_USER	quartzdesk	QuartzDesk Web Application database user.
DB_PASSWORD	quartzdesk	QuartzDesk Web Application database user password.
GAS_INSTALL_ROOT	/usr/local/glassfish4	GlassFish Application Server installation directory.
GAS_CONFIG	server-config	GlassFish Application Server configuration.
GAS_DOMAIN_NAME	domain1	GlassFish Application Server domain name.
GAS_DOMAIN_DIR	/usr/local/glassfish4/domains/domain1	GlassFish Application Server domain directory.
GAS_HTTP_HOST	localhost	GlassFish HTTP listener host.
GAS_HTTP_PORT	9080	GlassFish HTTP listener port.
WORK_DIR	/var/quartzdesk-web.work	QuartzDesk Web Application work directory.

3. Requirements

3.1 Software Requirements

3.1.1 Browser

The QuartzDesk Web Application's GUI requires a modern JavaScript-enabled browser. Please make sure JavaScript is enabled and not blocked by third party anti-virus/anti-malware software.

QuartzDesk Web Application has been tested with the following browser versions. These are also the minimum browsers versions required.

Browser	Minimum Version
Chrome	64
Firefox	45
Internet Explorer	8
Microsoft Edge	12
Opera	43
Safari	10

3.1.2 Operating System

Windows 7, Windows 8, Windows 10.
Linux (any distribution) with kernel 2.6.x and above.
Solaris 11.x and above.

3.1.3 JVM

Oracle JDK 8–13.
IBM JDK 8.
OpenJDK 8–13.

3.1.4 Application Server

Oracle GlassFish Application Server 5.x.

3.1.5 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

3.1.6 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 does not support the datetime2 data type at this time. As a result, all datetime values written by QuartzDesk Web 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/.

3.1.7 QuartzDesk Web Application Archive

To install QuartzDesk Web Application, you need to obtain the quartzdesk-web-x.y.z.war file. The latest version can be downloaded at www.quartzdesk.com (click Downloads → Latest Release → View files → quartzdesk-web-x.y.z.war).

3.2 Hardware Requirements

QuartzDesk Web Application runs on any physical or virtualized hardware that supports the above software requirements.

4. Installation

This chapter describes the standard QuartzDesk Web Application installation. If you are only evaluating, you can run QuartzDesk Web Application in the **one-step installation mode** to dramatically reduce the number of required installation steps. For details, please see our FAQs and search for “one-step installation”.

4.1 Database

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

Create a new QuartzDesk Web Application database named quartzdesk[1] (DB_NAME) owned by DB_USER.

In the quartzdesk database create a new schema named quartzdesk (DB_SCHEMA). The schema must be owned by DB_USER. Make the created DB_SCHEMA the default schema of 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 database objects (tables, keys, indices etc.) in the quartzdesk database / schema. These objects will be automatically created by QuartzDesk Web Application during its first start.

4.2 JDBC Driver

Download and install the JDBC driver for the created database. For a list of supported JDBC drivers please refer to chapter 3.1.6.

Copy the JDBC driver JAR file(s) to GAS_DOMAIN_DIR/lib/ext directory and restart the application server.

4.3 JDBC Connection Pool

In GAC (Resources → JDBC → JDBC Connection Pools) create a new JDBC connection pool for the QuartzDesk Web Application database.

The following steps depend on the QuartzDesk Web Application database type and are described in the following sub-chapters.

4.3.1 DB2

In Step 1, enter the following values:

[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. qdesk).

General Settings
Pool Name: QuartzDeskDS
Resource Type: javax.sql.ConnectionPoolDataSource
Database Driver Vendor: DB2
Introspect: uncheck

Click Next.

In Step 2, enter the following values:

General Settings
Datasource Clasname: com.ibm.db2.jcc.DB2ConnectionPoolDataSource
Ping: check

Pool Settings
Initial and Minumum Pool Size: 2
Maximum Pool Size: 10
Pool Resize Quality: 2
Idle Timeout: 300
Max Wait Time: 5000

Additional Properties
portNumber: DB_PORT
databaseName: DB_NAME
serverName: DB_HOST
user: DB_USER
password: DB_PASSWORD
driverType: 4

If a property is missing in the list, add it.

Click Finish.

In GAC click on the created JDBC connection pool (Resources → JDBC → JDBC Connection Pools → QuartzDeskDS) and click on the Advanced tab and enter the following values:

Statement Cache Size: 100

Connection Validation
Connection Validation: check Required
Validation Method: table
Table Name: sysibm.sysdummy1

Click Save.

4.3.2 H2

⚠ H2 is a light-weight Java database with limited fault tolerance and recovery functionality. We recommend using H2 for evaluation and experimental purposes only.

In Step 1, enter the following values:

General Settings

Pool Name: QuartzDeskDS
Resource Type: javax.sql.DataSource
Database Driver Vendor: leave empty
Introspect: uncheck

Click Next.

In Step 2, enter the following values:

General Settings
Datasource Clasname: org.h2.jdbcx.JdbcDataSource
Ping: check

Pool Settings
Initial and Minumum Pool Size: 2
Maximum Pool Size: 10
Pool Resize Quality: 2
Idle Timeout: 300
Max Wait Time: 5000

Additional Properties
User: DB_USER
Password: DB_PASSWORD
URL: jdbc:h2:file:<H2_DB_FILE_PATH>

If a property is missing in the list, add it.

Please note that H2 can be configured to run in various operating modes by adjusting the database URL value. For details, please refer to the H2 documentation at http://www.h2database.com/html/features.html#database_url.

Click Finish.

In GAC click on the created JDBC connection pool (Resources → JDBC → JDBC Connection Pools → QuartzDeskDS) and click on the Advanced tab and enter the following values:

Statement Cache Size: 100

Connection Validation
Connection Validation: check Required
Validation Method: table
Table Name: dual

Click Save.

4.3.3 Microsoft SQL Server

In Step 1, enter the following values:

General Settings
Pool Name: QuartzDeskDS
Resource Type: javax.sql.ConnectionPoolDataSource
Database Driver Vendor: MicrosoftSqlServer

Introspect: uncheck

Click Next.

In Step 2, enter the following values:

General Settings
Datasource Clasname: com.microsoft.sqlserver.jdbc.SQLServerConnectionPoolDataSource
Ping: check

Pool Settings
Initial and Minumum Pool Size: 2
Maximum Pool Size: 10
Pool Resize Quality: 2
Idle Timeout: 300
Max Wait Time: 5000

Additional Properties
User: DB_USER
DatabaseName: DB_NAME
ApplicationName: QuartzDesk Web Application
Password: DB_PASSWORD
ServerName: DB_HOST
PortNumber: DB_PORT

If a property is missing in the list, add it.

Depending on your Microsoft SQL Server configuration, you may need to set the value of the InstanceName property.

Click Finish.

In GAC click on the created JDBC connection pool (Resources → JDBC → JDBC Connection Pools → QuartzDeskDS) and click on the Advanced tab and enter the following values:

Statement Cache Size: 100

Connection Validation
Connection Validation: check Required
Validation Method: table
Table Name: qd_schema_update

Click Save.

4.3.4 MySQL

In Step 1, enter the following values:

General Settings
Pool Name: QuartzDeskDS
Resource Type: javax.sql.ConnectionPoolDataSource
Database Driver Vendor: MySql
Introspect: uncheck

Click Next.

In Step 2, enter the following values:

General Settings
Datasource Clasname: com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSource
Ping: check

Pool Settings
Initial and Minumum Pool Size: 2
Maximum Pool Size: 10
Pool Resize Quality: 2
Idle Timeout: 300
Max Wait Time: 5000

Additional Properties
User: DB_USER
ServerName: DB_HOST
Port: DB_PORT
DatabaseName: DB_NAME
Password: DB_PASSWORD

If a property is missing in the list, add it.

Click Finish.

In GAC click on the created JDBC connection pool (Resources → JDBC → JDBC Connection Pools → QuartzDeskDS) and click on the Advanced tab and enter the following values:

Statement Cache Size: 100

Connection Validation
Connection Validation: check Required
Validation Method: table
Table Name: qd_schema_update

Click Save.

4.3.5 Oracle

In Step 1, enter the following values:

General Settings
Pool Name: QuartzDeskDS
Resource Type: javax.sql.ConnectionPoolDataSource
Database Driver Vendor: Oracle
Introspect: uncheck

Click Next.

In Step 2, enter the following values:

General Settings
Datasource Clasname: oracle.jdbc.pool.OracleConnectionPoolDataSource
Ping: check

Pool Settings
Initial and Minumum Pool Size: 2
Maximum Pool Size: 10
Pool Resize Quality: 2
Idle Timeout: 300
Max Wait Time: 5000

Additional Properties
User: DB_USER
DatabaseName: DB_NAME
Password: DB_PASSWORD
ServerName: DB_HOST
DriverType: thin
PortNumber: DB_PORT
URL: jdbc:oracle:thin:@DB_HOST:DB_PORT:ORACLE_SERVICE_NAME

If a property is missing in the list, add it.

Click Finish.

In GAC click on the created JDBC connection pool (Resources → JDBC → JDBC Connection Pools → QuartzDeskDS) and click on the Advanced tab and enter the following values:

Statement Cache Size: 100

Connection Validation
Connection Validation: check Required
Validation Method: table
Table Name: dual

Click Save.

4.3.6 PostgreSQL

In Step 1, enter the following values:

General Settings
Pool Name: QuartzDeskDS
Resource Type: javax.sql.ConnectionPoolDataSource
Database Driver Vendor: Postgresql
Introspect: uncheck

Click Next.

In Step 2, enter the following values:

General Settings
Datasource Clasname: org.postgresql.ds.PGConnectionPoolDataSource
Ping: check

Pool Settings
Initial and Minumum Pool Size: 2
Maximum Pool Size: 10
Pool Resize Quality: 2
Idle Timeout: 300
Max Wait Time: 5000

Additional Properties
User: DB_USER
ApplicationName: QuartzDesk Web Application
DatabaseName: DB_NAME
Password: DB_PASSWORD
ServerName: DB_HOST
PortNumber: DB_PORT

If a property is missing in the list, add it.

Click Finish.

In GAC click on the created JDBC connection pool (Resources → JDBC → JDBC Connection Pools → QuartzDeskDS) and click on the Advanced tab and enter the following values:

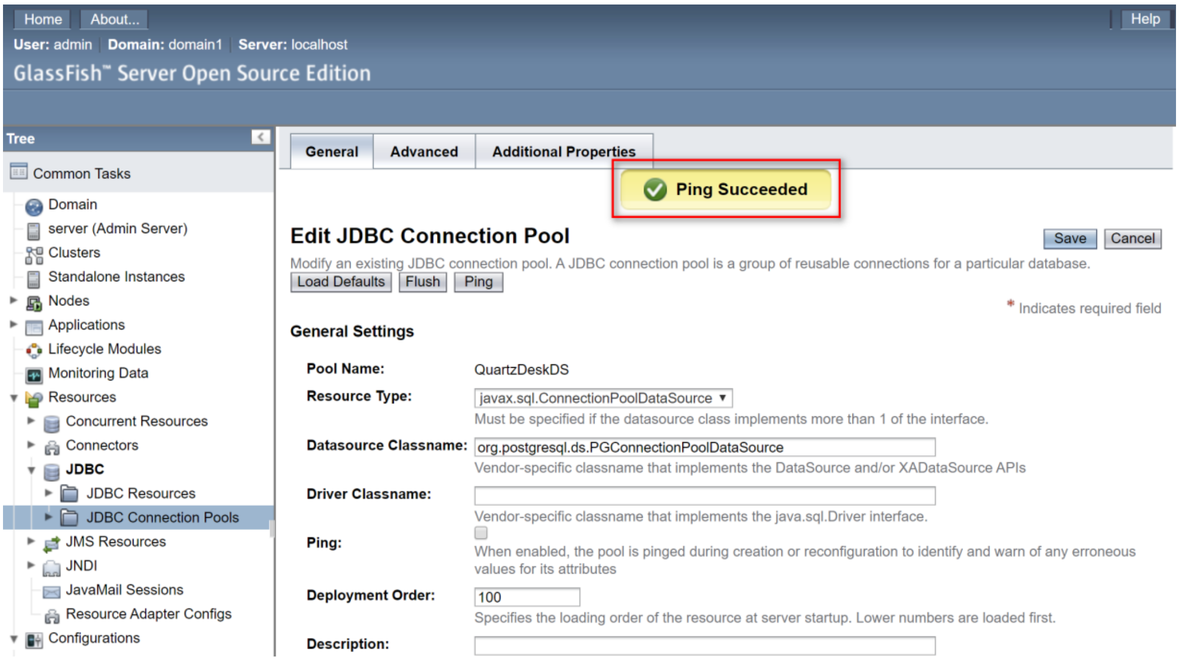
Statement Cache Size: 100

Connection Validation
Connection Validation: check Required
Validation Method: table
Table Name: qd_schema_update

Click Save.

4.4 Test JDBC Connection Pool

In GAC click on the created JDBC connection pool (Resources → JDBC → JDBC Connection Pools → QuartzDeskDS). In the General tab click on the Ping button to test the JDBC connection pool.



Ping Succeeded

Edit JDBC Connection Pool

Modify an existing JDBC connection pool. A JDBC connection pool is a group of reusable connections for a particular database.

* Indicates required field

General Settings

Pool Name: QuartzDeskDS

Resource Type: javax.sql.ConnectionPoolDataSource
Must be specified if the datasource class implements more than 1 of the interface.

Datasource Classname: org.postgresql.ds.PGConnectionPoolDataSource
Vendor-specific classname that implements the DataSource and/or XADataSource APIs

Driver Classname:
Vendor-specific classname that implements the java.sql.Driver interface.

Ping: □
When enabled, the pool is pinged during creation or reconfiguration to identify and warn of any erroneous values for its attributes

Deployment Order: 100
Specifies the loading order of the resource at server startup. Lower numbers are loaded first.

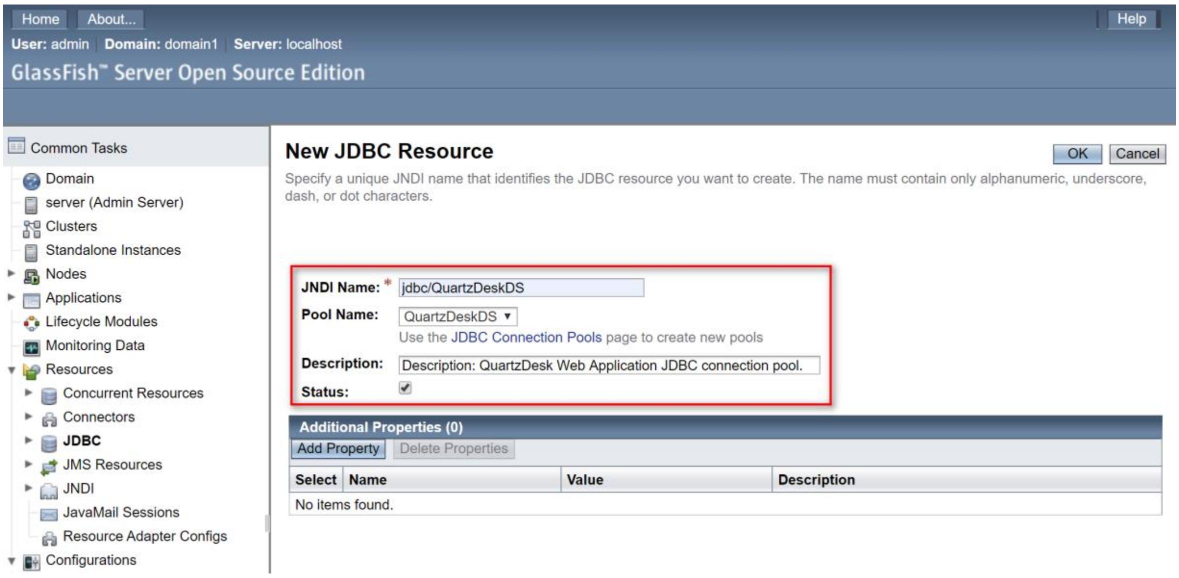
Description:

If the JDBC connection pool test fails, an error message is displayed and an exception is logged in the application server log (`GAS_DOMAIN_DIR/logs/server.log`).

4.5 JDBC Resource

In GAC (Resources → JDBC → JDBC Resources) create a new JDBC resource for the QuartzDeskDS JDBC connection pool create in the previous steps.

JNDI Name: jdbc/QuartzDeskDS
Pool Name: QuartzDeskDS
Description: QuartzDesk Web Application JDBC connection pool.
Status: checked



New JDBC Resource

Specify a unique JNDI name that identifies the JDBC resource you want to create. The name must contain only alphanumeric, underscore, dash, or dot characters.

JNDI Name: * jdbc/QuartzDeskDS

Pool Name: QuartzDeskDS
Use the JDBC Connection Pools page to create new pools

Description: Description: QuartzDesk Web Application JDBC connection pool.

Status: ☑

Additional Properties (0)

Select	Name	Value	Description
No items found.			

Click OK.

4.6 Application Work Directory

Create a QuartzDesk Web Application work directory (WORK_DIR) anywhere on the local file system. The directory must be readable and writeable by the user the GAS process runs under.

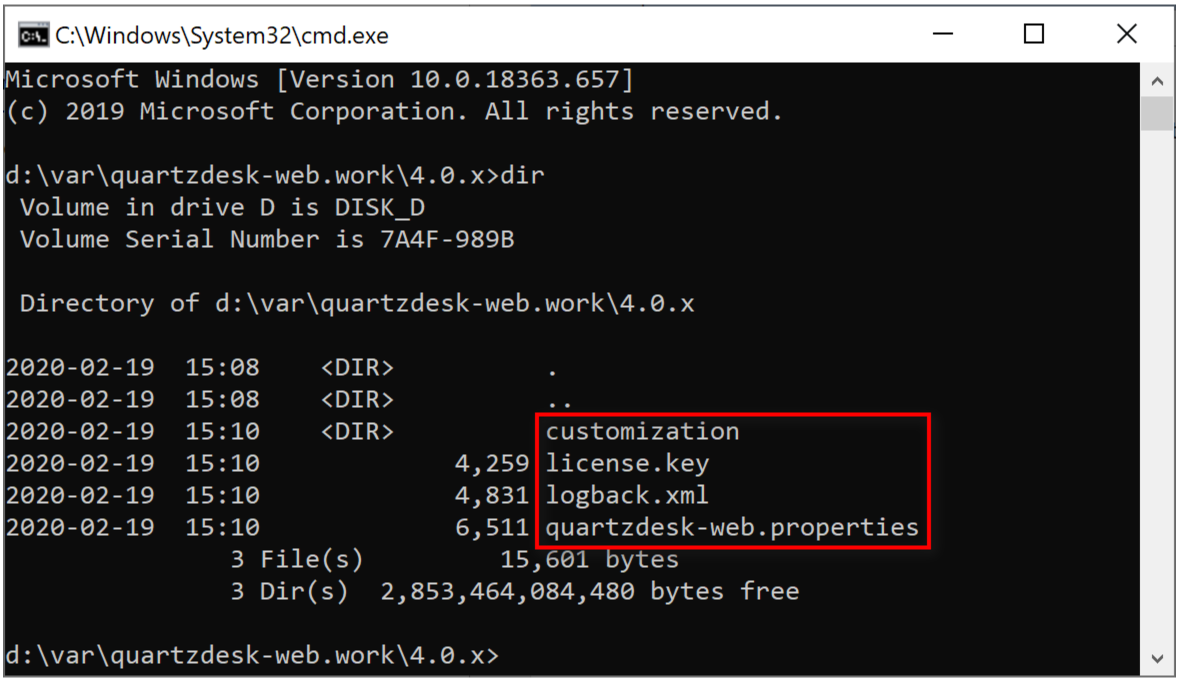
Copy your QuartzDesk license key file (`license.key`) to WORK_DIR.

ⓘ You can obtain a free 30-day trial license key at www.quartzdesk.com (go to Try / Purchase > Get Trial License Key).

Open the QuartzDesk Web Application archive (`quartzdesk-web-x.y.z.war`) and copy all files from the `extras/work` directory into WORK_DIR.

ⓘ If you cannot open the WAR file directly, rename it to *.zip. Do not forget to rename the file back to *.war once you have extracted the required files.

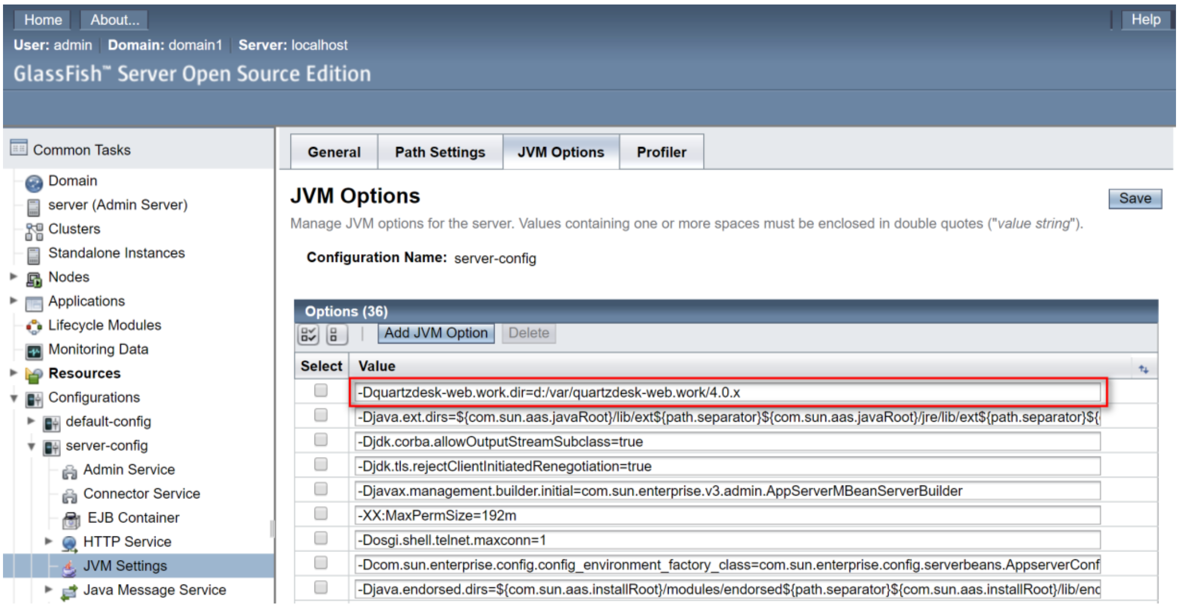
In the following figure you can see an example of a QuartzDesk Web Application work directory correctly set up on a Microsoft Windows machine.



```
Microsoft Windows [Version 10.0.18363.657]
(c) 2019 Microsoft Corporation. All rights reserved.

d:\var\quartzdesk-web.work\4.0.x>dir
 Volume in drive D is DISK_D
 Volume Serial Number is 7A4F-989B

 Directory of d:\var\quartzdesk-web.work\4.0.x

2020-02-19  15:08    <DIR>          .
2020-02-19  15:08    <DIR>          ..
2020-02-19  15:10    <DIR>          customization
2020-02-19  15:10             4,259 license.key
2020-02-19  15:10             4,831 logback.xml
2020-02-19  15:10             6,511 quartzdesk-web.properties
               3 File(s)         15,601 bytes
               3 Dir(s)  2,853,464,084,480 bytes free

d:\var\quartzdesk-web.work\4.0.x>
```

In GAC go to Configurations → GAS_CONFIG → JVM Settings → JVM Options. Add a new JVM option:

Value: -Dquartzdesk-web.work.dir=WORK_DIR



JVM Options

Manage JVM options for the server. Values containing one or more spaces must be enclosed in double quotes ("*value string*").

Configuration Name: server-config

Options (36)

Select	Value
□	-Dquartzdesk-web.work.dir=d:/var/quartzdesk-web.work/4.0.x
□	-Djava.ext.dirs=${com.sun.aas.javaRoot}/lib/ext${path.separator}${com.sun.aas.javaRoot}/jre/lib/ext${path.separator}${
□	-Djdk.corba.allowOutputStreamSubclass=true
□	-Djdk.tls.rejectClientInitiatedRenegotiation=true
□	-Djavax.management.builder.initial=com.sun.enterprise.v3.admin.AppServerMBeanServerBuilder
□	-XX:MaxPermSize=192m
□	-Dosgi.shell.telnet.maxconn=1
□	-Dcom.sun.enterprise.config.config_environment_factory_class=com.sun.enterprise.config.serverbeans.AppserverConf
□	-Djava.endorsed.dirs=${com.sun.aas.installRoot}/modules/endorsed${path.separator}${com.sun.aas.installRoot}/lib/enc

Save changes.

4.7 Application Configuration

Open the QuartzDesk Web Application configuration file `WORK_DIR/quartzdesk-web.properties`.

Based on the type and version of the database created in 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 (MyISAM)	>= 5.6	mysql
MySQL (InnoDB)	>= 5.6	mysql_innodb
Oracle	== 8i	oracle8
Oracle	>= 9i	oracle9
PostgreSQL	== 8.1	postgres81
PostgreSQL	>= 8.2	postgres82

Optionally, you can adjust the QuartzDesk Web Application logging parameters by editing the `WORK_DIR/logback.xml` configuration file. The default sample `logback.xml` configuration file makes QuartzDesk Web Application log under the `WORK_DIR/logs` directory that is automatically created when the web application starts. Please refer to the Logback Manual for Logback configuration details.

4.8 Deploy Application

In GAC go to Applications.

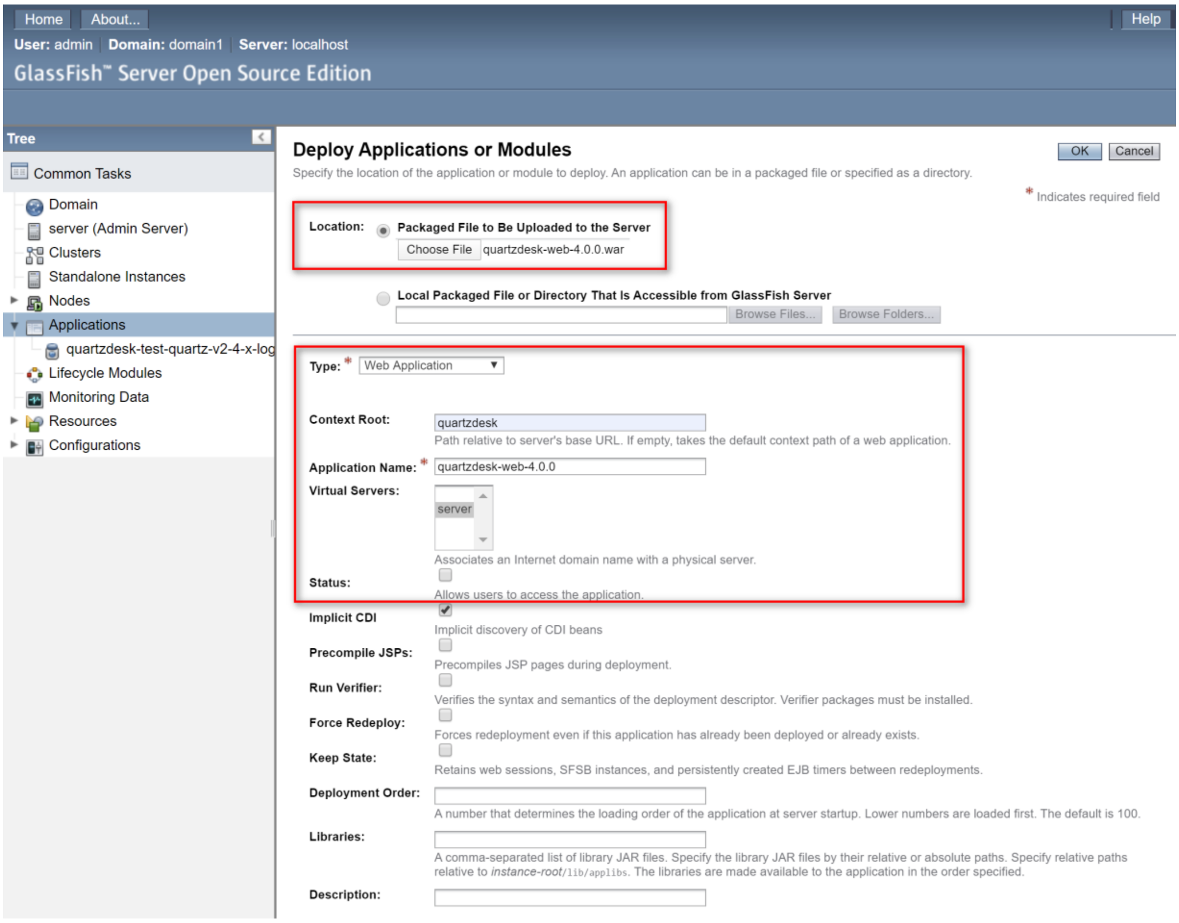
Click the Deploy... button and select the `quartzdesk-web-x.y.z.war` file and provide the following values:

Type: Web Application

Context Root: quartzdesk

Virtual Servers: Select the virtual servers QuartzDesk Web Application will be served by.

Status: Uncheck. This causes GlassFish not to start QuartzDesk Web Application automatically upon its deployment. The web application will be started manually in the following step.



Deploy Applications or Modules

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

* Indicates required field

Location: ● Packaged File to Be Uploaded to the Server
quartzdesk-web-4.0.0.war

○ Local Packaged File or Directory That Is Accessible from GlassFish Server

Type: * Web Application

Context Root: quartzdesk
Path relative to server's base URL. If empty, takes the default context path of a web application.

Application Name: * quartzdesk-web-4.0.0

Virtual Servers: server
Associates an Internet domain name with a physical server.

Status: □
Allows users to access the application.

Implicit CDI: ☑
Implicit discovery of CDI beans

Precompile JSPs: □
Precompiles JSP pages during deployment.

Run Verifier: □
Verifies the syntax and semantics of the deployment descriptor. Verifier packages must be installed.

Force Redeploy: □
Forces redeployment even if this application has already been deployed or already exists.

Keep State: □
Retains web sessions, SFSB instances, and persistently created EJB timers between redeployments.

Deployment Order:
A number that determines the loading order of the application at server startup. Lower numbers are loaded first. The default is 100.

Libraries:
A comma-separated list of library JAR files. Specify the library JAR files by their relative or absolute paths. Specify relative paths relative to *instance-root*/lib/applibs. The libraries are made available to the application in the order specified.

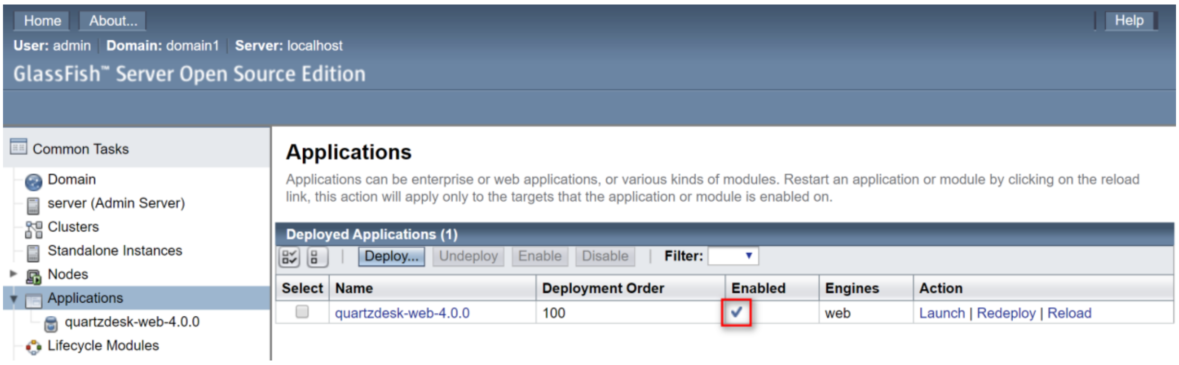
Description:

Click OK.

4.9 Start Application

In GAC go to Applications. Select QuartzDesk Web Application deployed in the previous step. Click the Enable button to start the application and wait for the startup procedure to complete.

Upon successful starting, the Enabled flag shown next to the QuartzDesk Web Application name in the Deployed Applications list, indicates that the application has been started.



Applications

Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload link, this action will apply only to the targets that the application or module is enabled on.

Deployed Applications (1)

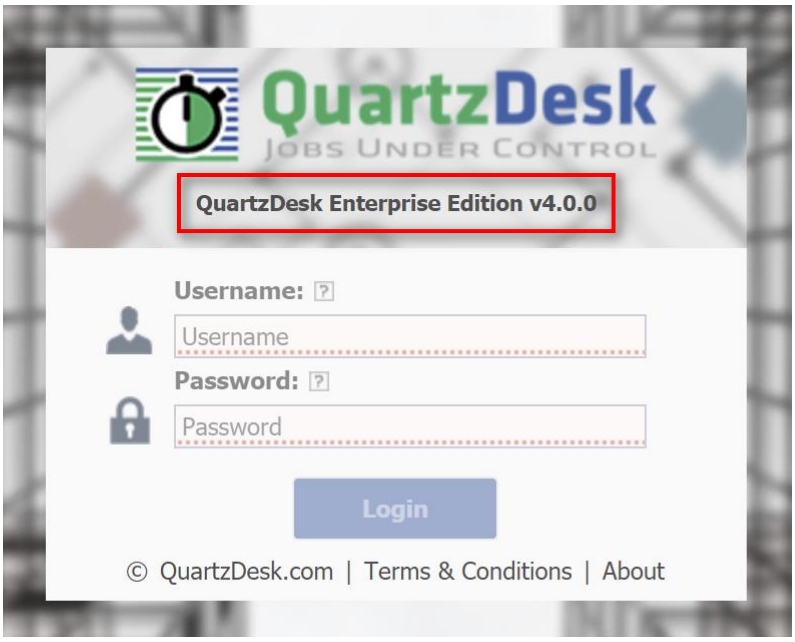
Select	Name	Deployment Order	Enabled	Engines	Action
□	quartzdesk-web-4.0.0	100	✓	web	Launch | Redeploy | Reload

Check the GAS `server.log` log file under GAS_`DOMAIN_DIR/logs` for errors.

Check the QuartzDesk Web Application logs (by default located in the `WORK_DIR/logs` directory) for errors.

If there are no errors, point your browser to `http://GAS_HTTP_HOST:GAS_HTTP_PORT/quartzdesk/` and verify that the QuartzDesk Web Application's GUI is accessible.

Check the version number of the deployed QuartzDesk Web Application.


QuartzDesk
Jobs Under Control
QuartzDesk Enterprise Edition v4.0.0
Username:
Username
Password:
Password
Login
© QuartzDesk.com | Terms & Conditions | About

To log in, use the default administrator login credentials:

Username: admin
Password: admin123

Once logged in, you can go to Settings > Users to manage users with access to the QuartzDesk Web Application's GUI. Users can be assigned different access permissions based on their intended roles.

In Settings > Groups, you can manage groups and assign access permissions to these groups. A group can contain users (members) who inherit access permissions of the group. A user can be a member of any number of groups.

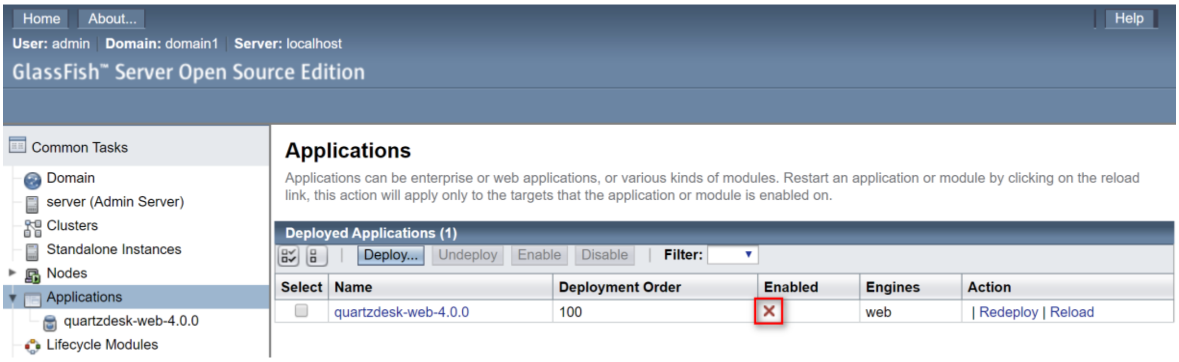
Effective access permissions of a user are permissions associated directly with the user plus access permissions of all groups the user is a member of.

5. Upgrading

5.1 Stop Existing Application

In GAC go to Applications. Select the checkbox next to QuartzDesk Web Application in the Applications list. Click the Disable button at the top of the list. Confirm this action in a dialog window that opens and wait for the action to complete.

Upon successful stopping, the Enabled flag shown next to the QuartzDesk Web Application name in the Deployed Applications list, indicates that the application has been stopped.



User: admin Domain: domain1 Server: localhost

GlassFish™ Server Open Source Edition

Applications

Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload link, this action will apply only to the targets that the application or module is enabled on.

Deployed Applications (1)

Select	Name	Deployment Order	Enabled	Engines	Action
☐	quartzdesk-web-4.0.0	100	✗	web	| Redeploy | Reload

5.2 Backup

Backup your QuartzDesk Web Application database. We recommend performing a **full database backup**.

Backup the contents of the QuartzDesk Web Application work directory.

Make sure you still have the WAR file of the existing QuartzDesk Web Application.

Store the backup files in a safe place so that you can restore the original QuartzDesk Web Application version if the need arises.

5.3 Remove Existing Application

In GAC go to Applications. Select the checkbox next to the existing QuartzDesk Web Application in the Applications list. Click the Undeploy button at the top of the list. Confirm this action in a dialog window that opens and wait for the action to complete.

Upon successful removal, the QuartzDesk Web Application’s name disappears from the Applications list.

5.4 Deploy New Application

Deploy the new version of QuartzDesk Web Application by following the deployment steps outlined in 4.8.

5.5 Start New Application

Start the new version of QuartzDesk Web Application by following the steps outlined in 4.9.

6. QuartzDesk 2.x to 3.x Migration Notes

To upgrade QuartzDesk Web Application 2.x to 3.x, follow the upgrade steps outlined in 5.

Before deploying the new QuartzDesk Web Application WAR file (quartzdesk-web-x.y.z.war), as outlined in 5.4, make sure you have implemented changes described in this chapter.

6.1 Minimum Required Java Version

QuartzDesk Web Application 3.x requires Java 7 or higher. Make sure GAS is configured to use Java 7 or higher.

6.2 Rename Configuration File

The name of the QuartzDesk Web Application 3.x configuration file has changed from `quartzdesk.properties` to `quartzdesk-web.properties`.

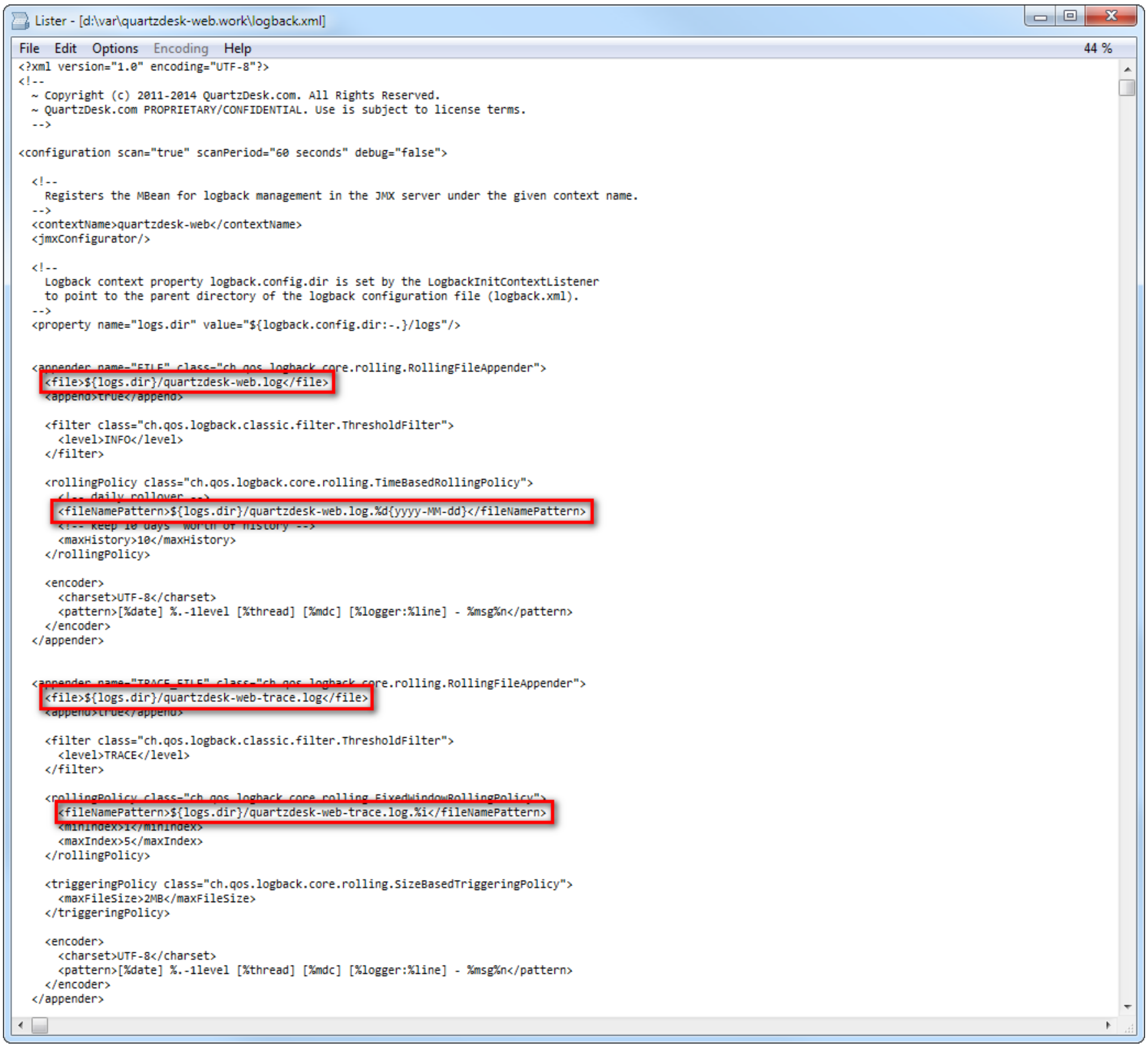
Rename the existing configuration file `quartzdesk.properties` located in the QuartzDesk Web Application work directory.

6.3 Rename Log Files

The names of QuartzDesk Web Application 3.x log files have changed.

Original Log File Name (2.x)	New Log File Name (3.x)
`quartzdesk.log`	`quartzdesk-web.log`
`quartzdesk-trace.log`	`quartzdesk-web-trace.log`

To use these new log file names, edit the QuartzDesk Web Application logging configuration file (`WORK_DIR/logback.xml`) and change the following lines:

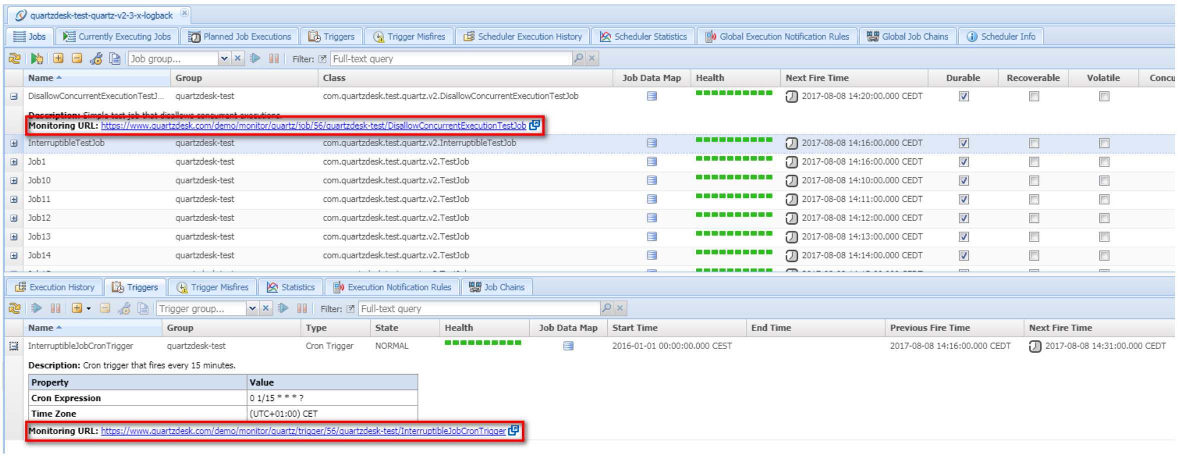


```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  ~ Copyright (c) 2011-2014 QuartzDesk.com. All Rights Reserved.
  ~ QuartzDesk.com PROPRIETARY/CONFIDENTIAL. Use is subject to license terms.
  -->

<configuration scan="true" scanPeriod="60 seconds" debug="false">

  <!--
    Registers the MBean for logback management in the JMX server under the given context name.
  -->
  <contextName>quartzdesk-web</contextName>
  <jmxConfigurator/>

  <!--
    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:-.}/logs"/>


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

    <filter class="ch.qos.logback.classic.filter.ThresholdFilter">
      <level>INFO</level>
    </filter>

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

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


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

    <filter class="ch.qos.logback.classic.filter.ThresholdFilter">
      <level>TRACE</level>
    </filter>

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

    <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
      <maxFileSize>2MB</maxFileSize>
    </triggeringPolicy>

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

Alternatively, extract the default `logback.xml` configuration file from the QuartzDesk Web Application 3.x WAR (`quartzdesk-web-x.y.z.war/extras/work/logback.xml`) and copy it to WORK_DIR.

6.4 Access to Monitoring URLs (REST API)

In QuartzDesk Web Application 2.x, the monitoring REST API URLs could be accessed by users with the QuartzDeskMonitor J2EE security role. In QuartzDesk Web Application 3.x, these monitoring URLs can be accessed by all authenticated users.



Name	Group	Class	Job Data Map	Health	Next Fire Time	Durable	Recoverable	Volatile	Concu
DisallowConcurrentExecutionTestJ...	quartzdesk-test	com.quartzdesk.test.quartz.v2.DisallowConcurrentExecutionTestJob			2017-08-08 14:20:00.000 CEDT	☑	☐	☐	
InterruptibleTestJob	quartzdesk-test	com.quartzdesk.test.quartz.v2.InterruptibleTestJob			2017-08-08 14:16:00.000 CEDT	☑	☐	☐	
Job1	quartzdesk-test	com.quartzdesk.test.quartz.v2.TestJob			2017-08-08 14:16:00.000 CEDT	☑	☐	☐	
Job10	quartzdesk-test	com.quartzdesk.test.quartz.v2.TestJob			2017-08-08 14:10:00.000 CEDT	☑	☐	☐	
Job11	quartzdesk-test	com.quartzdesk.test.quartz.v2.TestJob			2017-08-08 14:11:00.000 CEDT	☑	☐	☐	
Job12	quartzdesk-test	com.quartzdesk.test.quartz.v2.TestJob			2017-08-08 14:12:00.000 CEDT	☑	☐	☐	
Job13	quartzdesk-test	com.quartzdesk.test.quartz.v2.TestJob			2017-08-08 14:13:00.000 CEDT	☑	☐	☐	
Job14	quartzdesk-test	com.quartzdesk.test.quartz.v2.TestJob			2017-08-08 14:14:00.000 CEDT	☑	☐	☐	

Description: Simple test job that disallows concurrent executions.
Monitoring URL: https://www.quartzdesk.com/demo/monitor/quartz/job/56/quartzdesk-test/DisallowConcurrentExecutionTestJob

Name	Group	Type	State	Health	Job Data Map	Start Time	End Time	Previous Fire Time	Next Fire Time
InterruptibleJobCronTrigger	quartzdesk-test	Cron Trigger	NORMAL			2016-01-01 00:00:00.000 CEST		2017-08-08 14:16:00.000 CEDT	2017-08-08 14:31:00.000 CEDT

Description: Cron trigger that fires every 15 minutes.

Property	Value
Cron Expression	0 1/15 * * * ?
Time Zone	(UTC+01:00) CET

Monitoring URL: https://www.quartzdesk.com/demo/monitor/quartz/trigger/56/quartzdesk-test/InterruptibleJobCronTrigger

We recommend that you create a dedicated user account to access these monitoring URLs. The user account can be created in Settings → Users in the QuartzDesk Web Application's GUI.

All monitoring URLs in QuartzDesk Web Application 3.x support the HTTP Basic authentication scheme where the user's authentication credentials are passed in the `Authorization` HTTP header. Please note that the same authentication scheme was used by monitoring URLs in QuartzDesk Web Application 2.x.

6.5 Access to JAX-WS Endpoints

In QuartzDesk Web Application 2.x, all JAX-WS web service endpoints could be accessed by users with the QuartzDeskService J2EE security role. In QuartzDesk Web Application 3.x, these web service end points can only be accessed by authenticated users with particular access permissions.

The following table lists all JAX-WS web services and the security permissions that are required to access these web services.

JAX-WS Service	Required Permission
Connection Service	`WS_CONNECTION`
Security Service	`WS_SECURITY`
Quartz Service	`WS_QUARTZ`
Quartz Execution History Service	`WS_QUARTZ_EXEC_HISTORY`
Quartz Execution Notification Rule Service	`WS_QUARTZ_EXEC_NOTIF_RULE`
Quartz Job Chain Service	`WS_QUARTZ_JOB_CHAIN`

We recommend that you create a dedicated user account to access these JAX-WS endpoints. The user account can be created in Settings → Users in the QuartzDesk Web Application's GUI. Do not forget to assign the user the relevant permission(s).

All JAX-WS web service endpoints in QuartzDesk Web Application 3.x support the HTTP Basic authentication scheme where the user's authentication credentials are passed in the `Authorization` HTTP header. Please note that the same authentication scheme was used by JAX-WS endpoints in QuartzDesk Web Application 2.x.

6.6 Remove Unused Users

In GAC go to Configurations → GAS_CONFIG → Security → Realms → [name of the realm containing QuartzDesk Web Application 2.x users].

Click Manage Users button.

Select all users with either of the following three security roles – QuartzDeskMonitor, QuartzDeskService and QuartzDeskUser.

Click Delete button and delete all users who were previously used to access QuartzDesk Web Application 2.x. QuartzDesk Web Application 3.x no longer requires any users managed by a GAS security realm.

7. QuartzDesk 3.x to 4.x Migration Notes

No configuration changes are required.

8. Cluster Deployment Notes

When deploying QuartzDesk Web Application to a GlassFish 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 for a cluster deployment.

8.1 HTTP Session Replication and Affinity

QuartzDesk Web Application makes use of HTTP sessions and to store some short-lived and user-specific data. To achieve high-availability (HA), it is necessary to make the session data available on all application cluster members so that when one cluster member becomes unavailable, the remaining cluster members can take over and handle user requests without the user noticing any service interruption. To make the session data available on all application cluster members, the HTTP session replication process must be enabled on the cluster.

ⓘ The amount of data stored by QuartzDesk Web Application in an HTTP session is kept at the absolute minimum to reduce the session replication overhead. The total size of data stored in the session does not exceed 1KB.

When configuring session replication, we recommend that you also enable session affinity (sticky-sessions) on the load-balancer so that all user requests are preferably passed to the GlassFish instance that handled the first user request that established the session.

Please refer to the GlassFish and load-balancer documentation for details on how to configure session replication and session affinity because the actual steps may vary depending on the GlassFish cluster topology and configuration.

8.2 Shared Work Directory

We recommend that you put the QuartzDesk Web Application work directory, described 4.3, on a shared drive and make this work directory available to all cluster members. Not only does this make application and configuration upgrading easier, it is actually requited by all “Save” (for example, Save Log, Save Chart etc.) actions provided by the QuartzDesk Web Application’s GUI. These actions trigger two subsequent HTTP requests where the first request prepares the data and stores it in the `WORK_DIR/tmp` directory and the second request downloads the data and makes the browser open the Save As dialog.

During a fail-over or if the session affinity is not enabled, it can easily happen that the first request is handled by cluster member A and the second request is handled by cluster member B. If A and B are not configured to use the same `WORK_DIR/tmp` directory, then B will fail to serve the data prepared by A during the preceding request because the data will not be found.

8.3 Logging Configuration

If you set up your cluster to use a shared QuartzDesk Web Application work directory, as described in the previous chapter, you will need to edit the QuartzDesk Web Application logging configuration file `WORK_DIR/logback.xml` and decide where QuartzDesk Web Application 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 Web Application uses two log files – `quartzdesk-web.log` and `quartzdesk-web-trace.log` that are stored in `WORK_DIR/logs` directory. The following chapters discuss these two options.

8.3.1 Using Shared Log Files

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

```
...

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


<appender name="TRACE_FILE"
class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-web-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="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <!-- daily rollover -->
    <fileNamePattern>${logs.dir}/quartzdesk-web.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 QuartzDesk Web Application's performance, we generally discourage using the prudent mode and shared log files.

8.3.2 Using Separate Log Files

In order to make individual QuartzDesk Web Application 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 be referred to from the `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 Web Application instances are stored:

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

```
...

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

  ...

  <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <!-- daily rollover -->
    <fileNamePattern>${logs.dir}/quartzdesk-web-
${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="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-web-${cluster.member.instanceId}-
trace.log</file>
  <append>true</append>

  ...

  <rollingPolicy
class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
    <fileNamePattern>${logs.dir}/quartzdesk-web-
${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"/>

...
```

8.4 Internal Quartz Scheduler

QuartzDesk Web Application ships with an embedded Quartz scheduler to periodically execute its internal jobs. When deploying QuartzDesk Web Application to a cluster, it is necessary to **assign unique instance IDs to Quartz scheduler instances** running in the clustered QuartzDesk Web Application instances.

For these purposes the QuartzDesk Web Application configuration (`quartzdesk-web.properties` file) provides the `scheduler.org.quartz.scheduler.instanceIdGenerator.class` configuration property. The value of this property must be a fully-qualified class name of a Java class that implements the `org.quartz.spi.InstanceIdGenerator` Quartz API interface. Quartz API provides two out of the box implementations suitable for clustered QuartzDesk Web Application deployments:

Implementation	Description
`org.quartz.simpl.HostnameInstanceIdGenerator`	This implementation is suitable for QuartzDesk Web Application deployments where individual clustered QuartzDesk Web Application instances run on distinct hosts and each of these hosts is assigned a unique hostname. This is the default implementation used by QuartzDesk Web Application. No configuration changes are necessary to use this instance ID generator.
`org.quartz.simpl.SystemPropertyInstanceIdGenerator`	This implementation is suitable for QuartzDesk Web Application deployments where some of the clustered QuartzDesk Web Application instances run on the same host. This implementation extracts the Quartz scheduler instance ID from the `org.quartz.scheduler.instanceId` JVM system property that must be explicitly set. Please refer to the GlassFish documentation for details on how to add a new JVM system property.

Please refer to the table above and optionally modify the value of the `scheduler.org.quartz.scheduler.instanceIdGenerator.class` configuration property according to the cluster configuration.