



QuartzDesk Web Application Installation and Upgrade Guide for IBM WebSphere AS 8.5 and 9.0

QuartzDesk Version: 4.x

March 3, 2020



Table of Contents

1.	PURPOSE	4
2.	DEFINITIONS	5
3.	REQUIREMENTS.....	6
3.1	SOFTWARE REQUIREMENTS	6
3.1.1	Browser	6
3.1.2	Operating System	6
3.1.3	JVM	6
3.1.4	Application Server.....	6
3.1.5	Database	6
3.1.6	Database JDBC Driver	6
3.1.7	QuartzDesk Web Application Archive.....	7
3.2	HARDWARE REQUIREMENTS.....	7
4.	INSTALLATION.....	8
4.1	DATABASE.....	8
4.2	JDBC DRIVER	8
4.3	JDBC PROVIDER	9
4.3.1	DB2	9
4.3.2	H2.....	11
4.3.3	Microsoft SQL Server.....	11
4.3.4	MySQL.....	13
4.3.5	Oracle.....	14
4.3.6	PostgreSQL.....	15
4.4	DATA SOURCE J2C AUTHENTICATION DATA	16
4.5	DATA SOURCE	16
4.5.1	DB2	17
4.5.2	H2.....	19
4.5.3	Microsoft SQL Server.....	21
4.5.4	MySQL.....	23
4.5.5	Oracle.....	25
4.5.6	PostgreSQL.....	28
4.6	APPLICATION WORK DIRECTORY	30
4.7	APPLICATION CONFIGURATION.....	31
4.8	DEPLOY APPLICATION.....	31
4.9	START APPLICATION	35
5.	UPGRADING	38
5.1	STOP EXISTING APPLICATION	38
5.2	BACKUP.....	38
5.3	REMOVE EXISTING APPLICATION	38
5.4	DEPLOY NEW APPLICATION.....	39
5.5	START NEW APPLICATION	39
6.	QUARTZDESK 2.X TO 3.X MIGRATION NOTES	40
6.1	MINIMUM REQUIRED JAVA VERSION.....	40
6.2	RENAME CONFIGURATION FILE	40
6.3	RENAME LOG FILES	40
6.4	ACCESS TO MONITORING URLS (REST API).....	41
6.5	ACCESS TO JAX-WS ENDPOINTS	42
7.	QUARTZDESK 3.X TO 4.X MIGRATION NOTES	43
7.1	CLASS LOADER ORDER	43

8.	CLUSTER DEPLOYMENT NOTES	44
8.1	HTTP SESSION REPLICATION AND AFFINITY	44
8.2	SHARED WORK DIRECTORY.....	44
8.3	LOGGING CONFIGURATION	44
	8.3.1 Using Shared Log Files.....	45
	8.3.2 Using Separate Log Files.....	46
8.4	INTERNAL QUARTZ SCHEDULER	47



1. Purpose

This document describes the installation and upgrade process for QuartzDesk Web Application 4.x on IBM WebSphere Application Server 8.5 and 9.0.

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 <code>.ear</code> extension.
JAR	Java Application Archive. A file with <code>.jar</code> extension.
JVM	Java Virtual Machine.
WAC	WebSphere Administrative Console.
WAR	Web Application Archive. A file with <code>.war</code> extension.
WAS	WebSphere Application Server.

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.
WAS_INSTALL_ROOT	/usr/local/was9	WebSphere Application Server installation directory.
WAS_SERVER_NAME	server1	WebSphere Application Server name.
WAS_SERVER_PROFILE	/usr/local/was9/profiles/server1	WebSphere Application Server profile directory.
WAS_HTTP_HOST	localhost	WebSphere HTTP listener host.
WAS_HTTP_PORT	9080	WebSphere 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

IBM JDK 8 bundled with the IBM WebSphere Application Server.

3.1.4 Application Server

IBM WebSphere Application Server 8.5.

IBM WebSphere Application Server 9.0.

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	<p>Microsoft JDBC driver 4.0 for SQL Server available at http://msdn.microsoft.com/en-us/sqlserver/aa937724.aspx.</p> <p>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 the 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.</p>
MySQL	Connector/J JDBC driver available at http://dev.mysql.com/downloads/connector/j/ .
Oracle	<p>Oracle JDBC driver available at http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html.</p> <p>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.</p>
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”.

Unless noted otherwise, all WebSphere variables and objects created in WAC and described in this document are created in the node scope.

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 `quartzdesk1` (`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 the 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.

In WAC (Environment → WebSphere variables) create a new variable pointing to the directory with the JDBC driver JAR file for the QuartzDesk Web Application database. Make sure the JDBC driver JAR file is readable by the user the WAS process runs under.

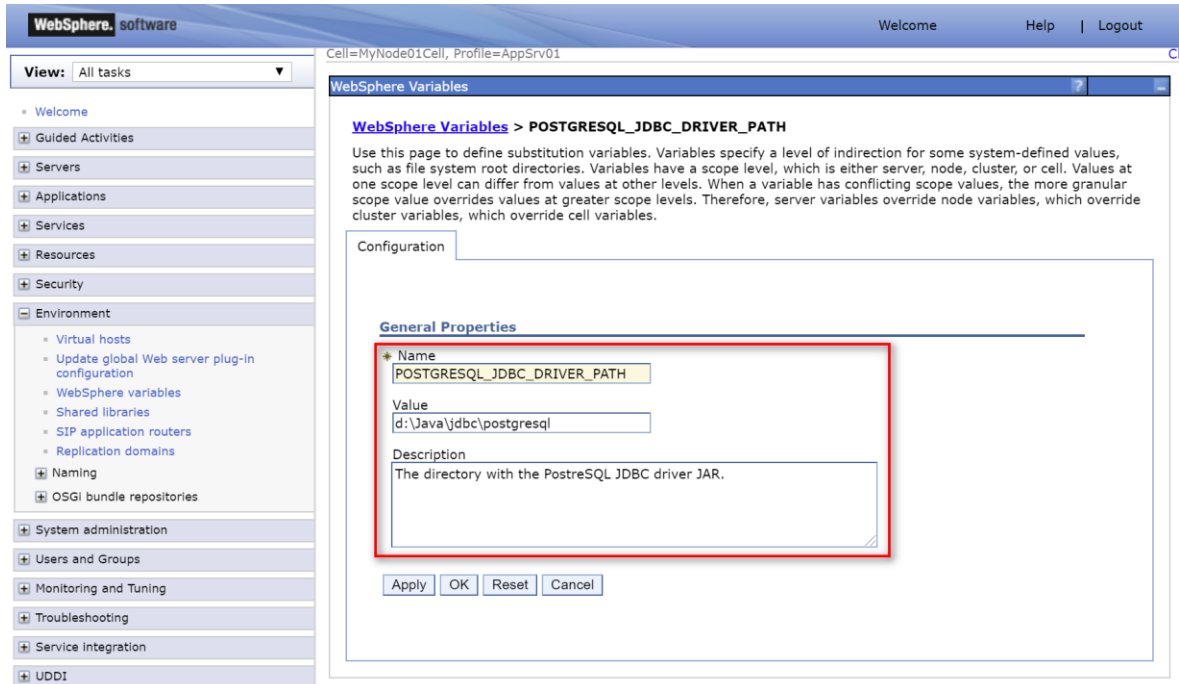
We suggest that you use the following variable names. Please note that some of these variables may already be defined. If the variable is already defined and points to a directory with a JDBC driver that is supported by QuartzDesk Web Application, then leave the variable as is. Otherwise, add, or adjust the variable value accordingly.

Database	WebSphere Variable Name	Default Value In WAC
DB2	<code>DB2_JCC_DRIVER_PATH</code>	Defined with empty value.
H2	<code>H2_JDBC_DRIVER_PATH</code>	Not defined.
Microsoft SQL Server	<code>MICROSOFT_JDBC_DRIVER_PATH</code>	Defined with empty value.

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`).

MySQL	MYSQL_JDBC_DRIVER_PATH	Not defined.
Oracle	ORACLE_JDBC_DRIVER_PATH	Defined with empty value.
PostgreSQL	POSTGRESQL_JDBC_DRIVER_PATH	Not defined.

The following is an example of creating a new variable named POSTGRESQL_JDBC_DRIVER_PATH pointing to the directory that contains the PostgreSQL JDBC driver JAR file.



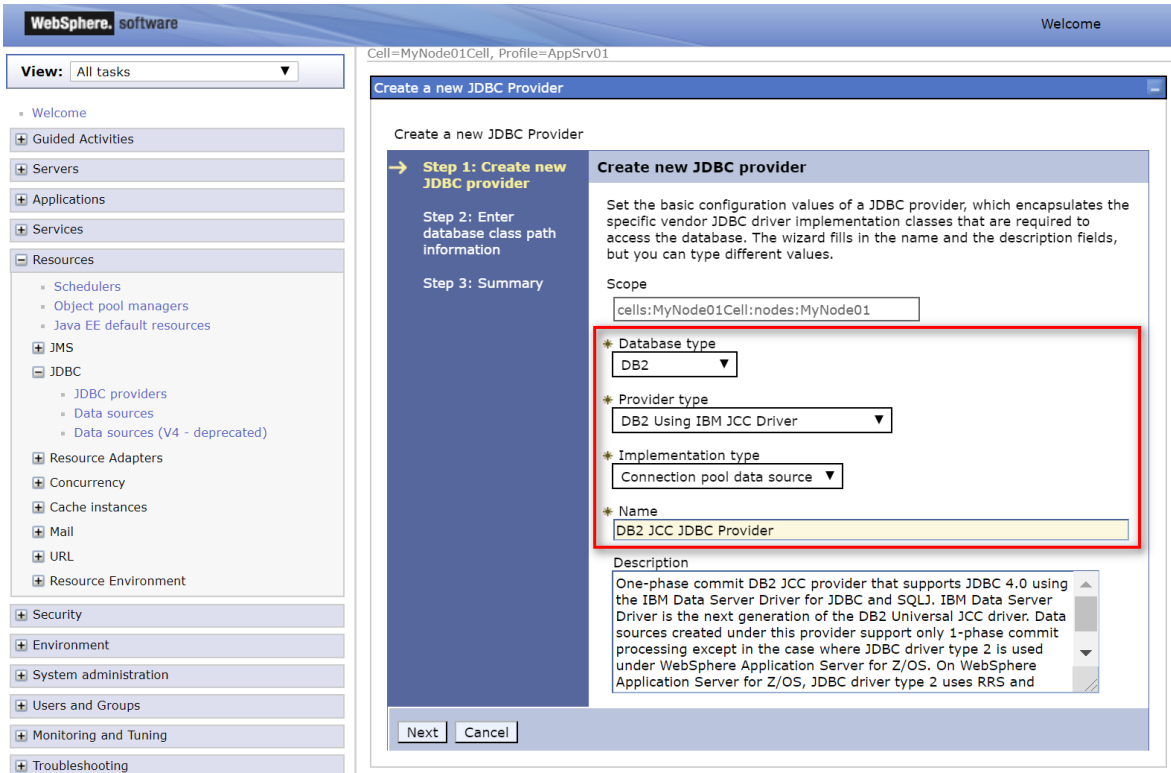
4.3 JDBC Provider

In WAC (Resources → JDBC → JDBC providers), register a JDBC provider for the QuartzDesk Web Application data-source.

4.3.1 DB2

Database type: DB2
Provider type: DB2 Using IBM JCC Driver
Implementation type: Connection pool data source
Name: DB2 JCC JDBC Provider





WebSphere. software Welcome

Cell=MyNode01Cell, Profile=AppSrv01

Create a new JDBC Provider

Create a new JDBC Provider

Step 1: Create new JDBC provider

Step 2: Enter database class path information

Step 3: Summary

Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.

Scope
cells:MyNode01Cell:nodes:MyNode01

* Database type
DB2

* Provider type
DB2 Using IBM JCC Driver

* Implementation type
Connection pool data source

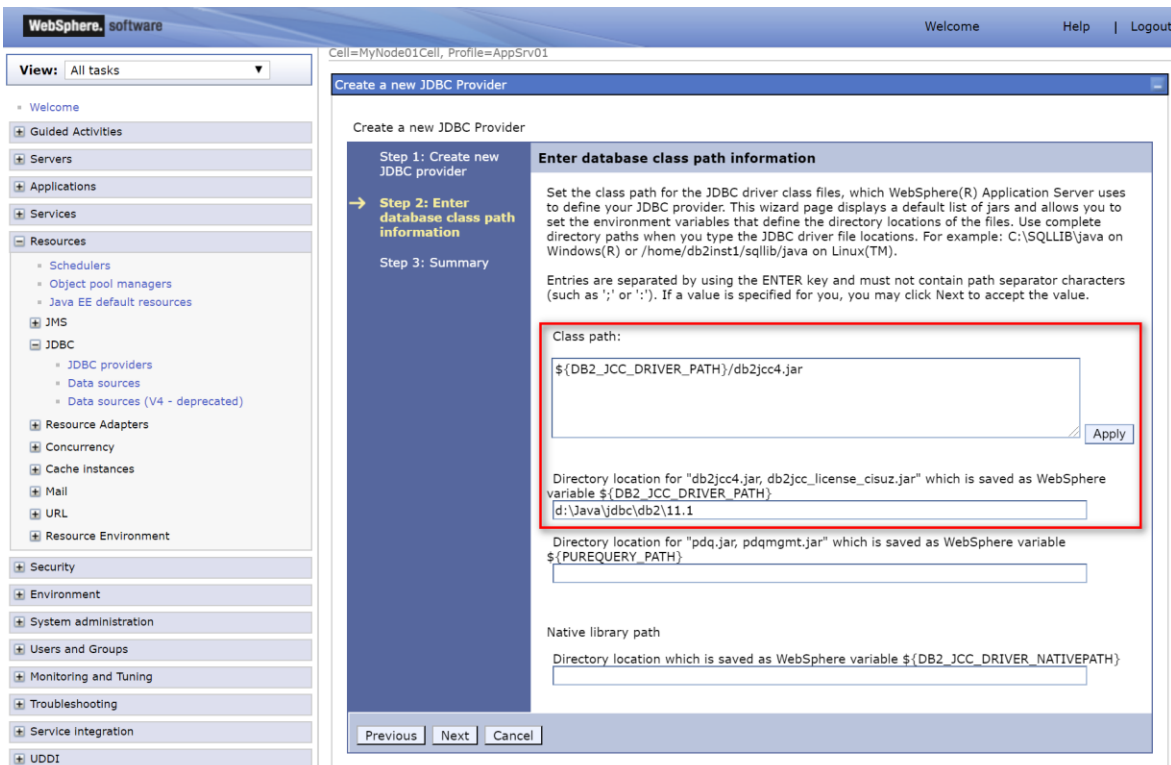
* Name
DB2 JCC JDBC Provider

Description
One-phase commit DB2 JCC provider that supports JDBC 4.0 using the IBM Data Server Driver for JDBC and SQLJ. IBM Data Server Driver is the next generation of the DB2 Universal JCC driver. Data sources created under this provider support only 1-phase commit processing except in the case where JDBC driver type 2 is used under WebSphere Application Server for Z/OS. On WebSphere Application Server for Z/OS, JDBC driver type 2 uses RRS and

Next Cancel

Click Next.

Class path: `${DB2_JCC_DRIVER_PATH}/db2jcc.jar`



WebSphere. software Welcome Help Logout

Cell=MyNode01Cell, Profile=AppSrv01

Create a new JDBC Provider

Create a new JDBC Provider

Step 1: Create new JDBC provider

Step 2: Enter database class path information

Step 3: Summary

Set the class path for the JDBC driver class files, which WebSphere(R) Application Server uses to define your JDBC provider. This wizard page displays a default list of jars and allows you to set the environment variables that define the directory locations of the files. Use complete directory paths when you type the JDBC driver file locations. For example: C:\SQLLIB\java on Windows(R) or /home/db2inst1/sqllib/java on Linux(TM).

Entries are separated by using the ENTER key and must not contain path separator characters (such as ';' or ':'). If a value is specified for you, you may click Next to accept the value.

Class path:
\${DB2_JCC_DRIVER_PATH}/db2jcc4.jar

Apply

Directory location for "db2jcc4.jar, db2jcc_license_cisuz.jar" which is saved as WebSphere variable `${DB2_JCC_DRIVER_PATH}`
d:\Java\jdbc\db2\11.1

Directory location for "pdq.jar, pdqgmt.jar" which is saved as WebSphere variable `${PUREQUERY_PATH}`

Native library path
Directory location which is saved as WebSphere variable `${DB2_JCC_DRIVER_NATIVEPATH}`

Previous Next Cancel

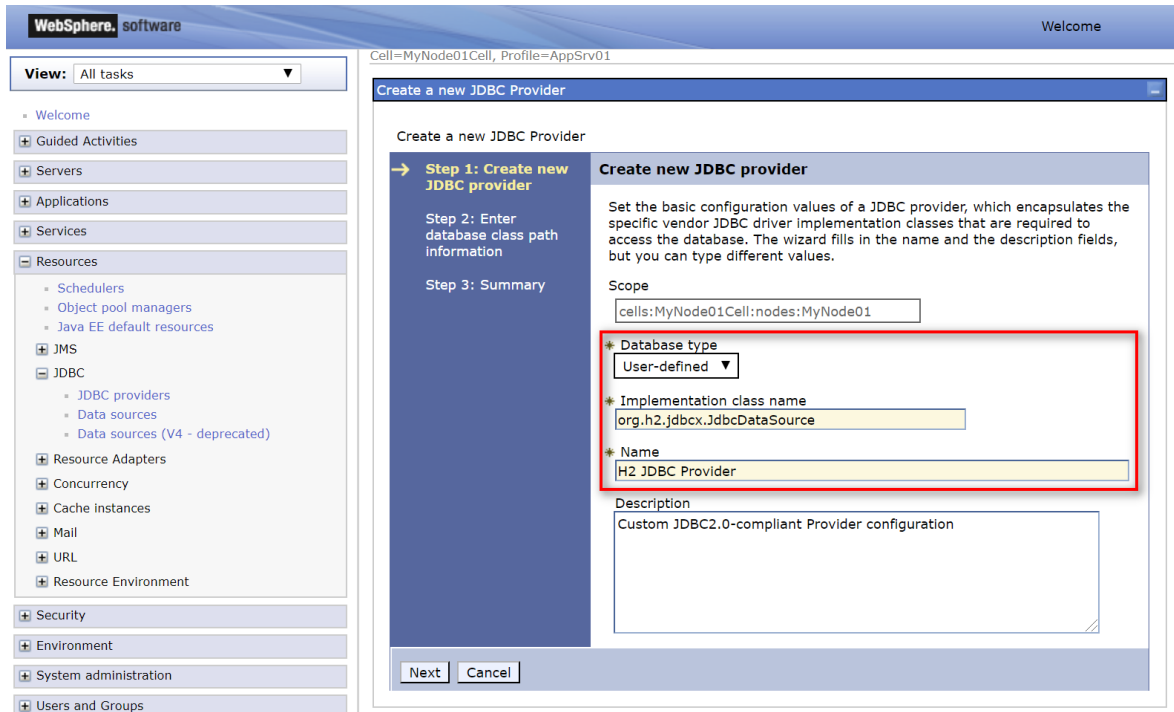
Click Next and then Finish. Save changes.

4.3.2 H2

Database type: User-defined

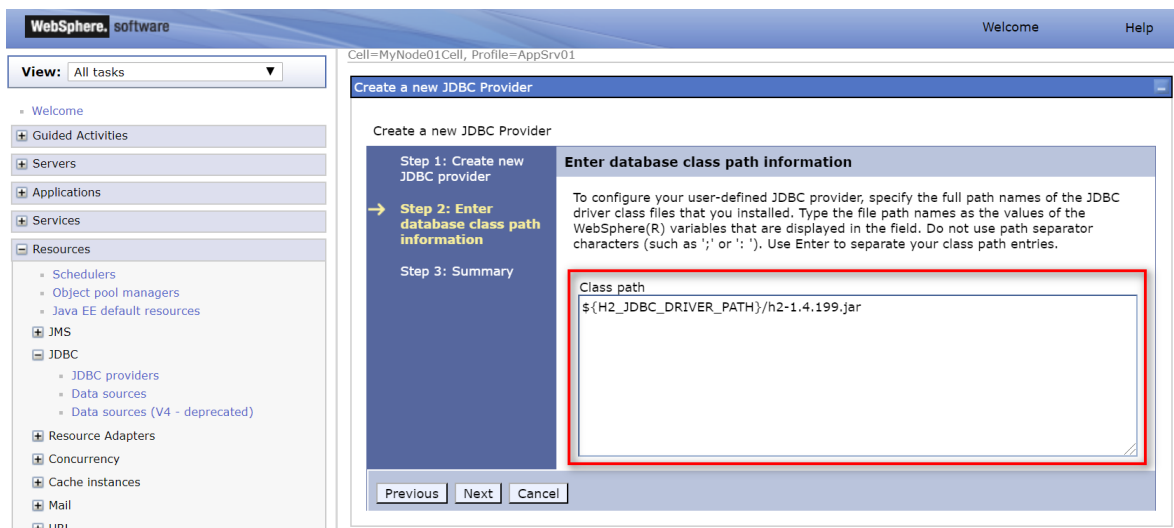
Implementation class name: `org.h2.jdbcx.JdbcDataSource`

Name: H2 JDBC Provider



Click Next.

Class path: `${H2_JDBC_DRIVER_PATH}/h2-x.y.z.jar`



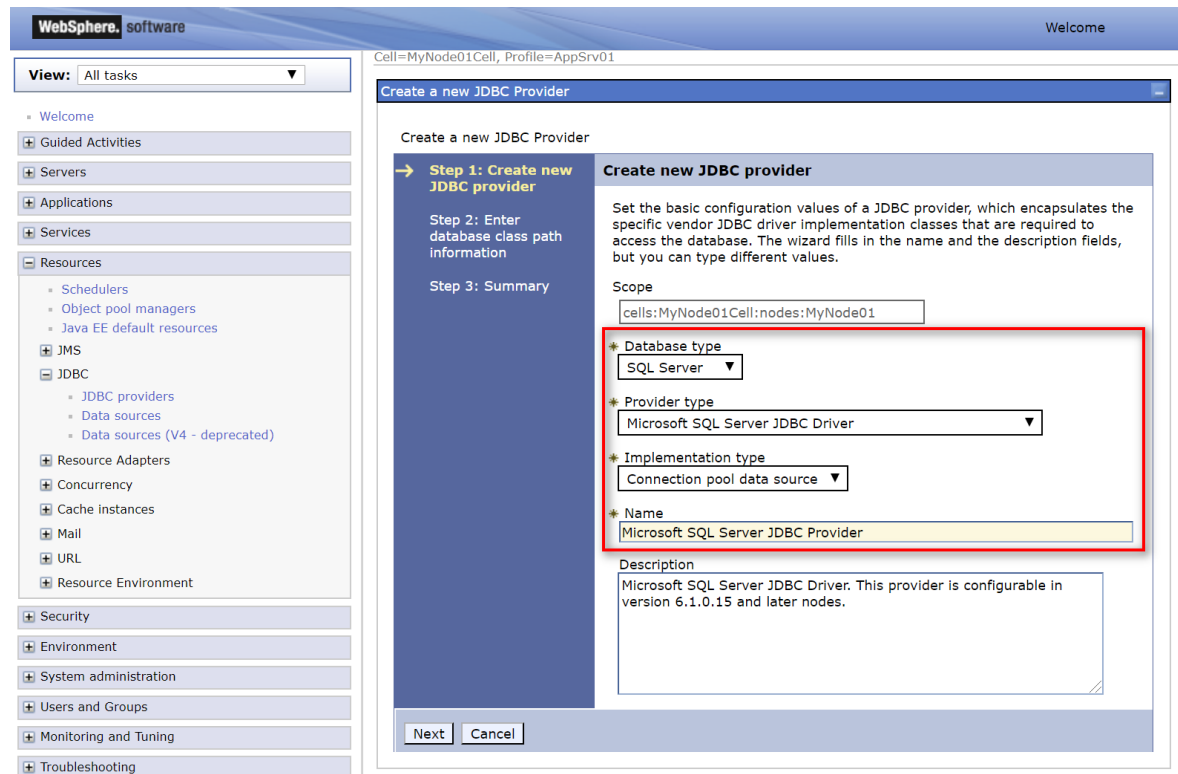
Click Next and then Finish. Save changes.

4.3.3 Microsoft SQL Server

Database type: SQL Server

Provider type: Microsoft SQL Server JDBC Driver

Implementation type: Connection pool data source
Name: Microsoft SQL Server JDBC Provider



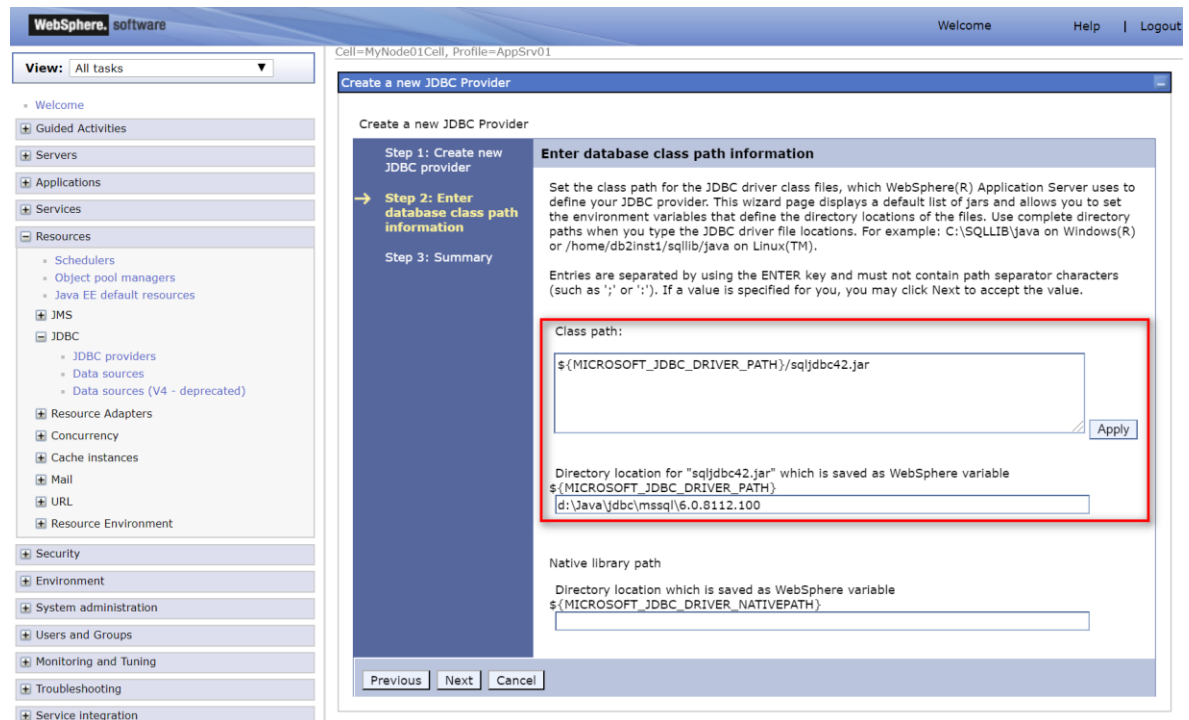
The screenshot shows the 'Create a new JDBC Provider' wizard in the WebSphere Admin Console. The left sidebar shows the navigation tree with 'JDBC' expanded. The main content area is titled 'Create a new JDBC provider' and includes the following fields:

- Scope: cells:MyNode01Cell:nodes:MyNode01
- Database type: SQL Server
- Provider type: Microsoft SQL Server JDBC Driver
- Implementation type: Connection pool data source
- Name: Microsoft SQL Server JDBC Provider
- Description: Microsoft SQL Server JDBC Driver. This provider is configurable in version 6.1.0.15 and later nodes.

Buttons for 'Next' and 'Cancel' are visible at the bottom.

Click Next.

Class path: `${MICROSOFT_JDBC_DRIVER_PATH}/sqljdbc4.jar`



The screenshot shows the 'Create a new JDBC Provider' wizard in the WebSphere Admin Console, Step 2: Enter database class path information. The main content area includes the following fields:

- Class path: `${MICROSOFT_JDBC_DRIVER_PATH}/sqljdbc4.jar`
- Directory location for "sqljdbc4.jar" which is saved as WebSphere variable `${MICROSOFT_JDBC_DRIVER_PATH}`: `d:\Java\jdbc\mssql\6.0.8112.100`
- Native library path: `${MICROSOFT_JDBC_DRIVER_NATIVEPATH}`

Buttons for 'Previous', 'Next', and 'Cancel' are visible at the bottom.

Click Next and then Finish. Save changes.

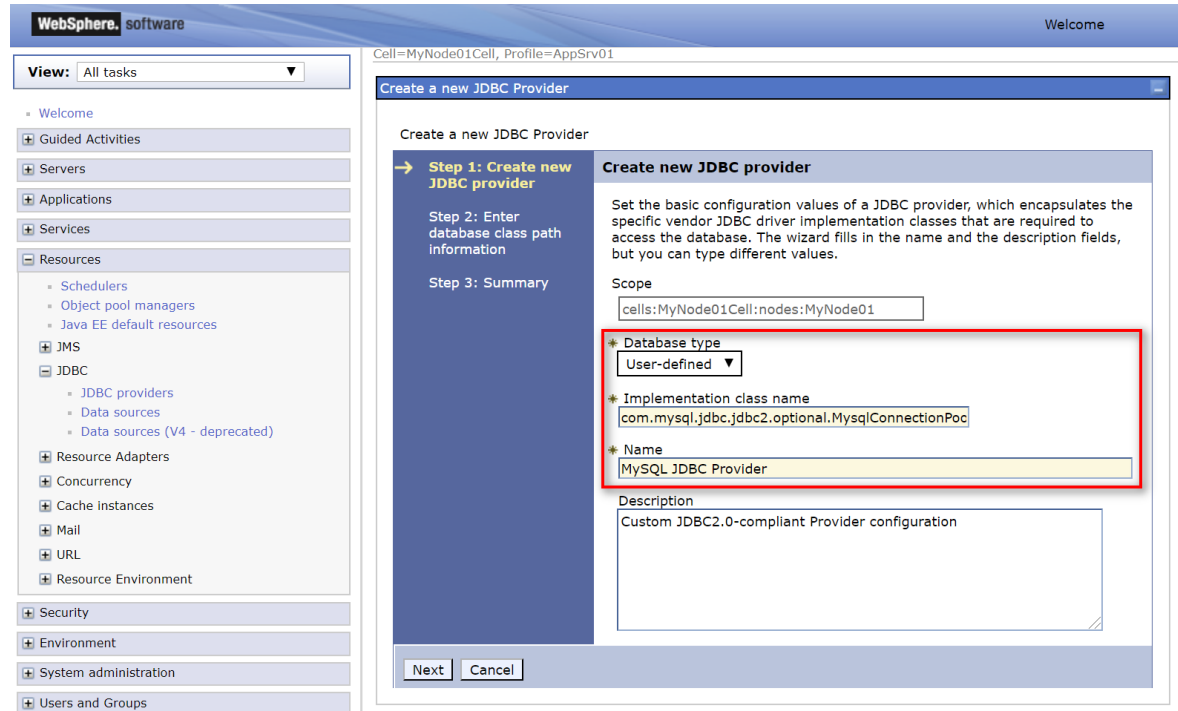
4.3.4 MySQL

Database type: User-defined

Implementation class name:

`com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSource`

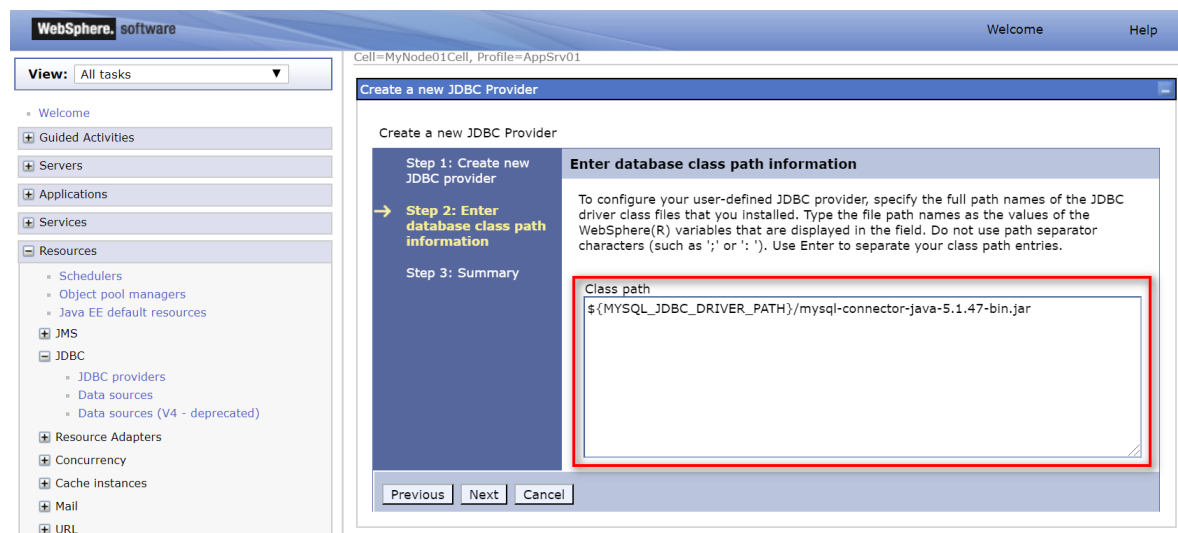
Name: MySQL JDBC Provider



The screenshot shows the 'Create a new JDBC Provider' wizard in the WebSphere Admin Console. The left sidebar shows the navigation tree with 'JDBC' expanded. The main window displays the 'Create new JDBC provider' step. The 'Database type' is set to 'User-defined'. The 'Implementation class name' is `com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoc`. The 'Name' is 'MySQL JDBC Provider'. The 'Description' is 'Custom JDBC2.0-compliant Provider configuration'. The 'Scope' is `cells:MyNode01Cell:nodes:MyNode01`. The 'Next' button is highlighted.

Click Next.

Class path: `${MYSQL_JDBC_DRIVER_PATH}/mysql-connector-java-x.y.z-bin.jar`



The screenshot shows the 'Create a new JDBC Provider' wizard in the WebSphere Admin Console, Step 2: Enter database class path information. The 'Class path' field contains `${MYSQL_JDBC_DRIVER_PATH}/mysql-connector-java-5.1.47-bin.jar`. The 'Previous', 'Next', and 'Cancel' buttons are visible at the bottom.

Click Next and then Finish. Save changes.

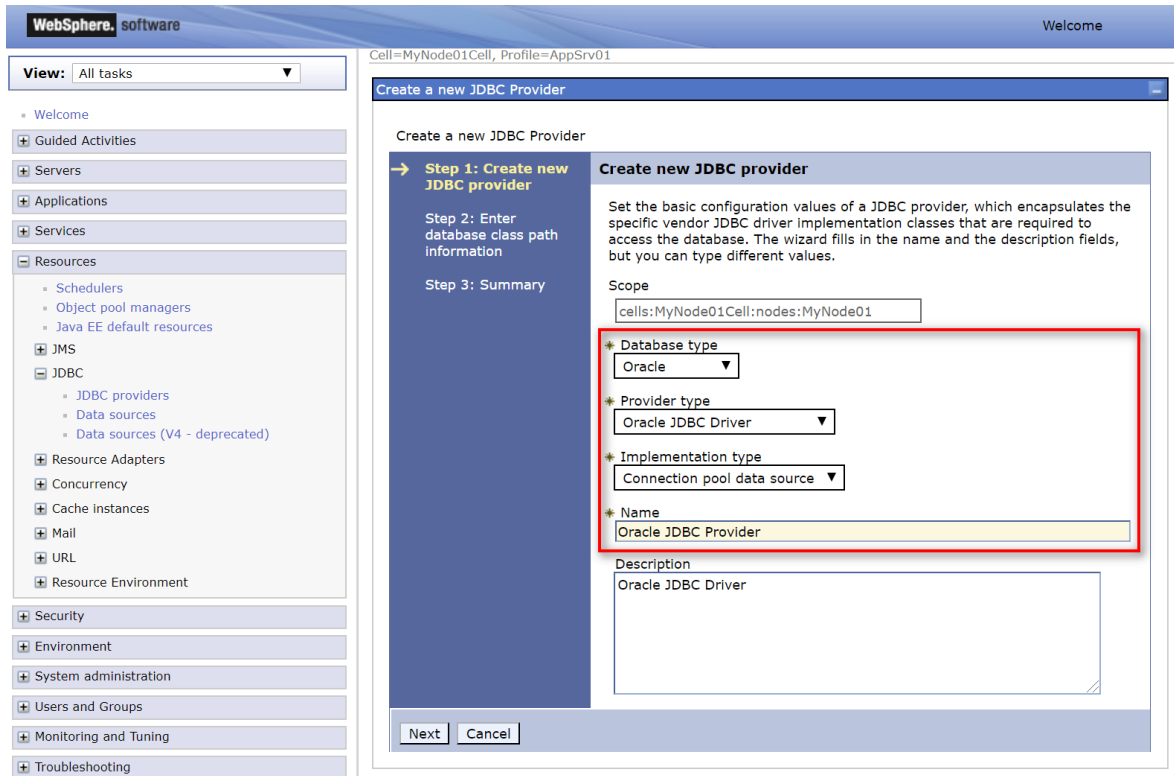
4.3.5 Oracle

Database type: Oracle

Provider type: Oracle JDBC Driver

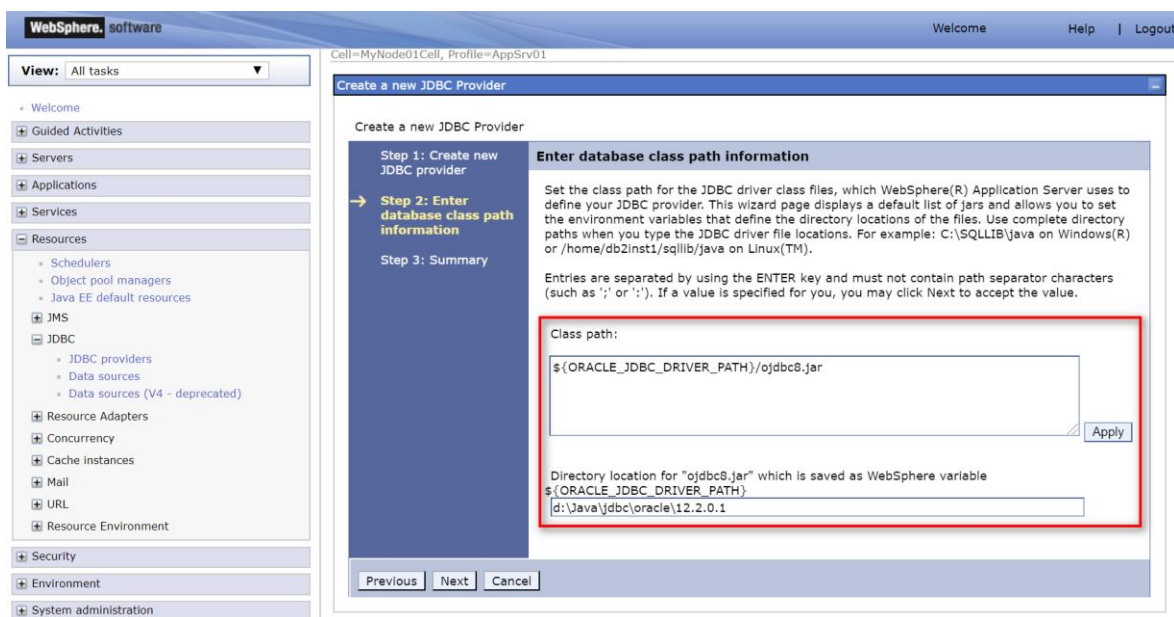
Implementation type: Connection pool data source

Name: Oracle JDBC Provider



Click Next.

Class path: `${ORACLE_JDBC_DRIVER_PATH}/ojdbc8.jar`



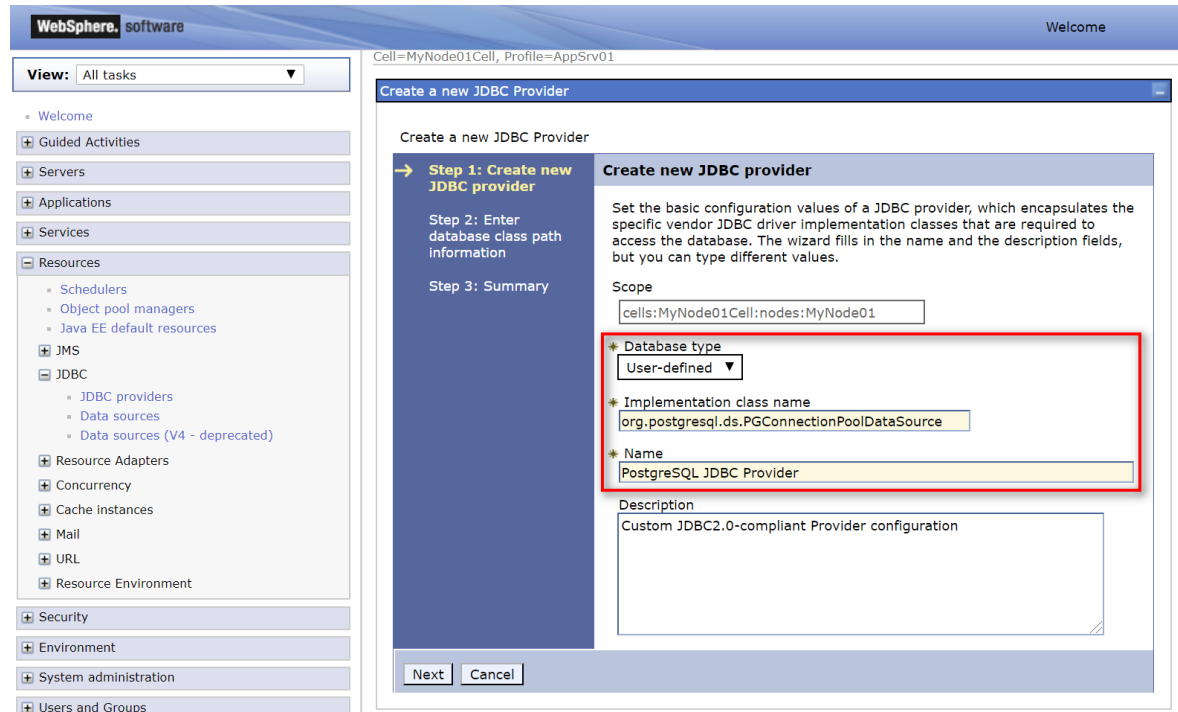
Click Next and then Finish. Save changes.

4.3.6 PostgreSQL

Database type: User-defined

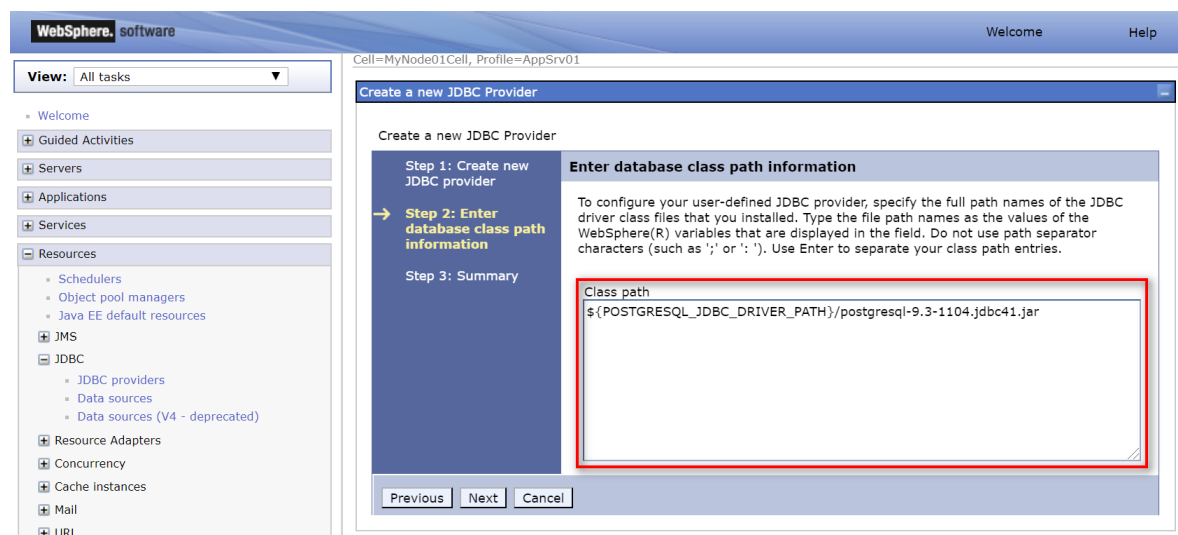
Implementation class name: `org.postgresql.ds.PGConnectionPoolDataSource`

Name: PostgreSQL JDBC Provider



Click Next.

Class path: `${POSTGRESQL_JDBC_DRIVER_PATH}/postgresql-x-y-z.jar`



Click Next and then Finish. Save changes.

4.4 Data Source J2C Authentication Data

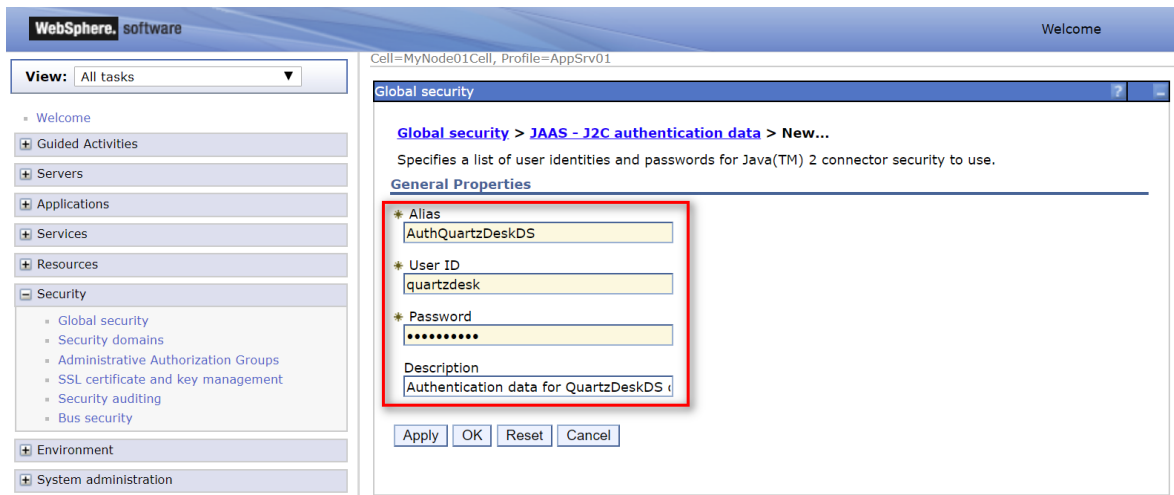
In WAC (Security → Global Security → Java Authentication and Authorization Service → J2C Authentication Data), create a new authentication entry for the QuartzDesk Web Application data source.

Alias: AuthQuartzDeskDS

User ID: DB_USER

Password: DB_PASSWORD

Description: Authentication data for QuartzDeskDS data source.



Click OK. Save changes.

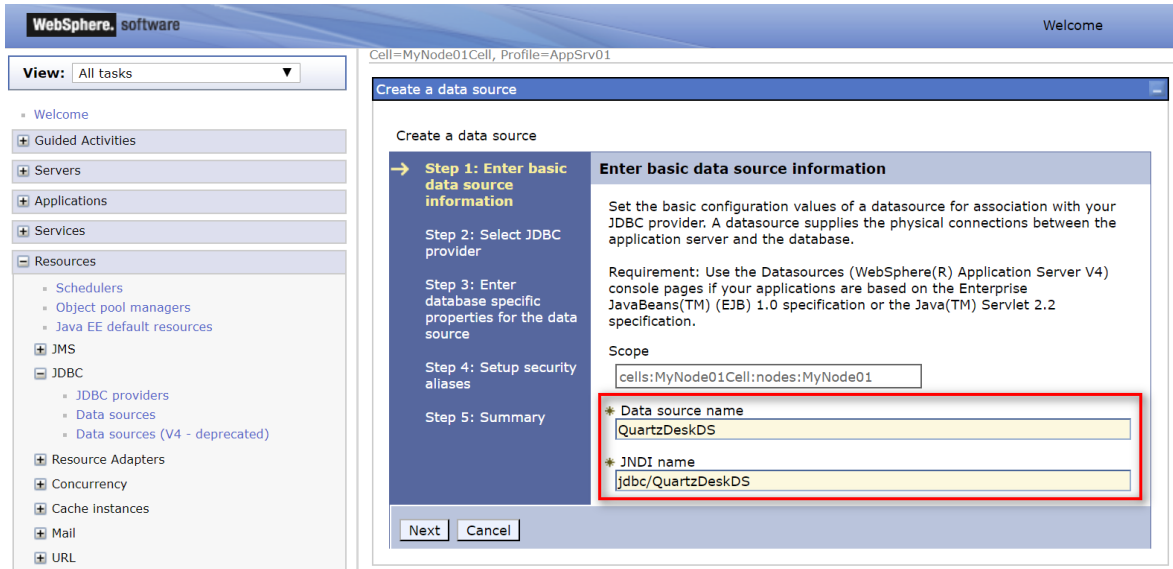
4.5 Data Source

In WAC (Resources → JDBC → Data sources), create a new data source for the QuartzDesk Web Application database.

In Step 1, provide the data source name and JNDI name.

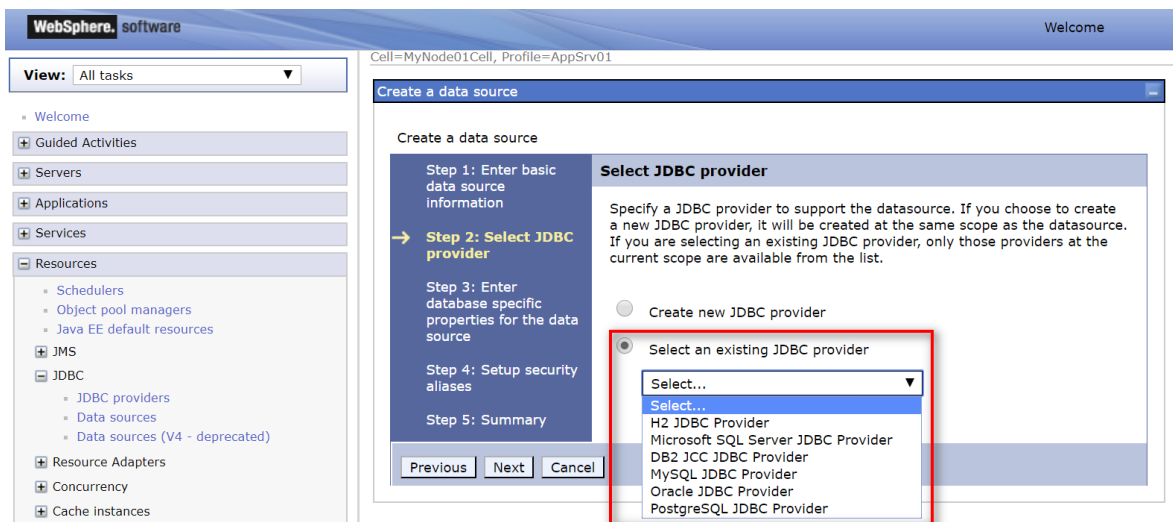
Data source name: QuartzDeskDS

JNDI name: jdbc/QuartzDeskDS



Click Next.

In Step 2, select the JDBC provider created in 4.3.



Click Next.

The following steps depend on the selected JDBC provider.

4.5.1 DB2

In Step 3, provide these values:

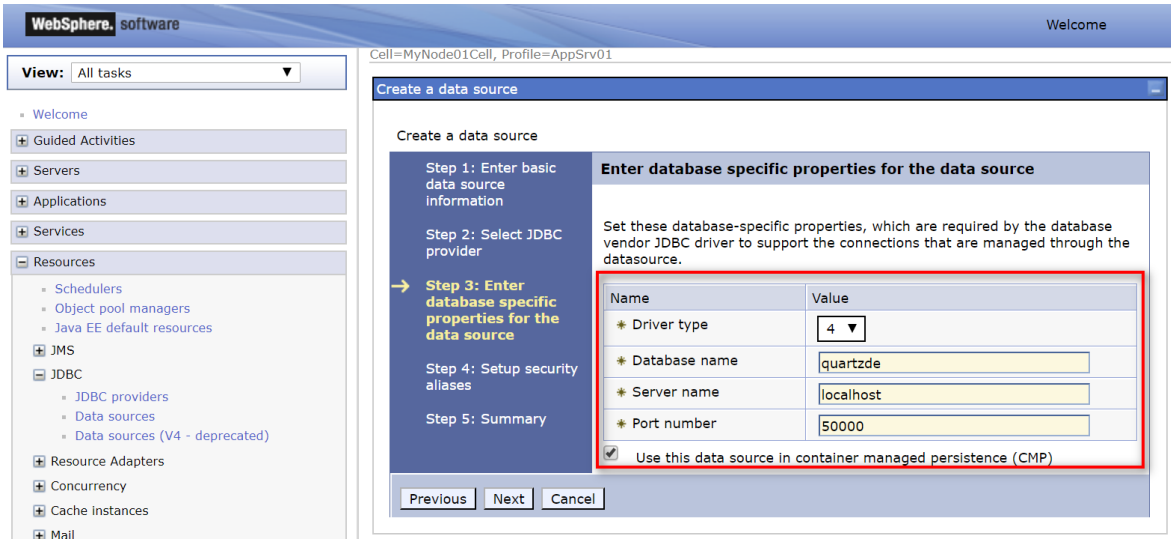
Driver type: 4

Database name: DB_NAME

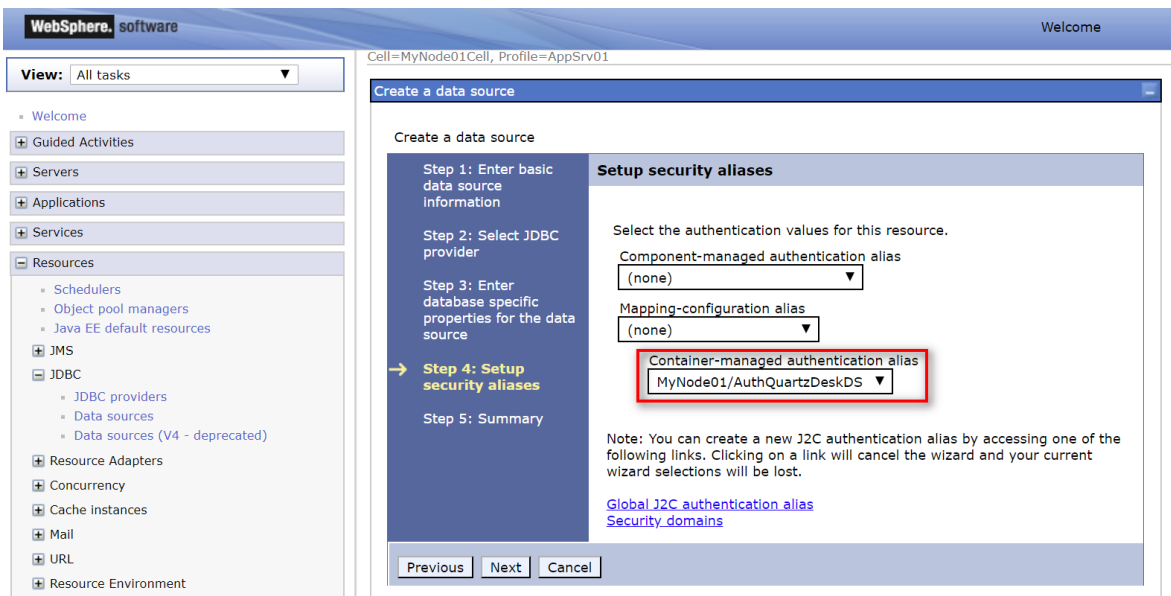
Server name: DB_HOST

Port number: DB_PORT

Leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the created data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:
Statement cache size: 100

Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select "Validation by SQL query" with the following query:

```
select 1 from sysibm.sysdummy1
```

Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

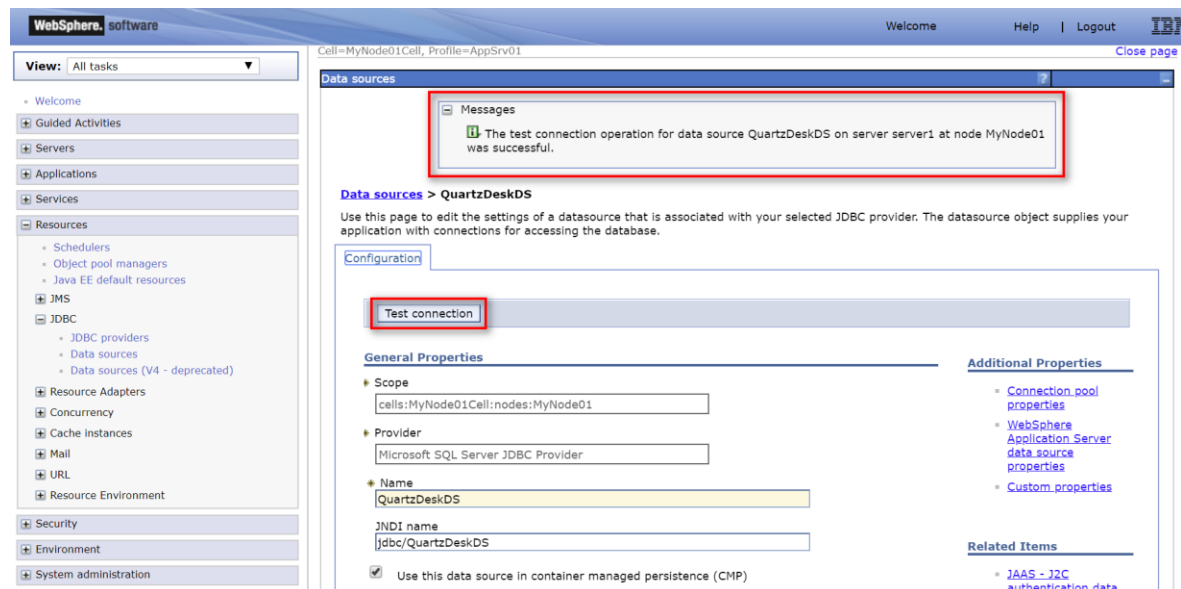
Set the following properties:

Name: clientApplicationInformation

Value: QuartzDesk Web Application

Apply and Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



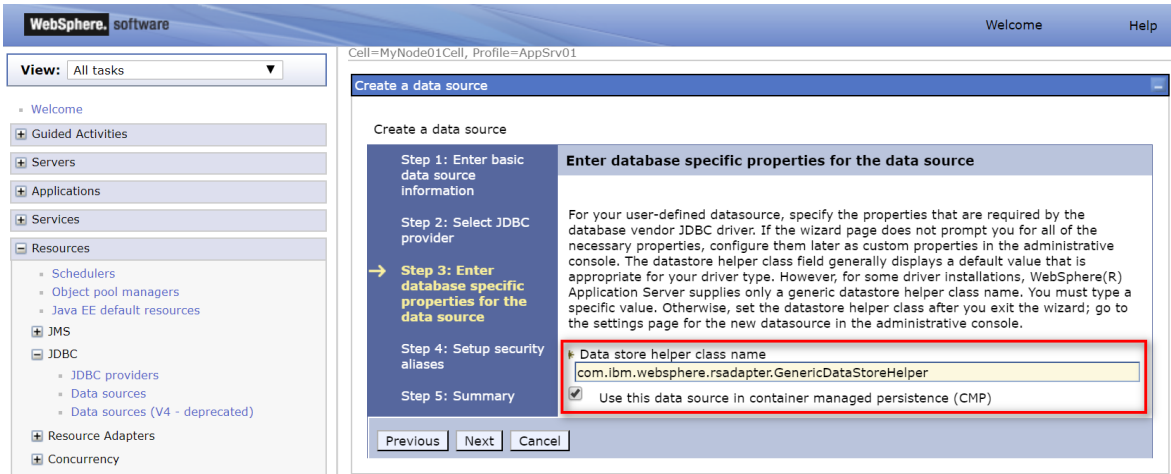
Check there are no errors displayed.

4.5.2 H2

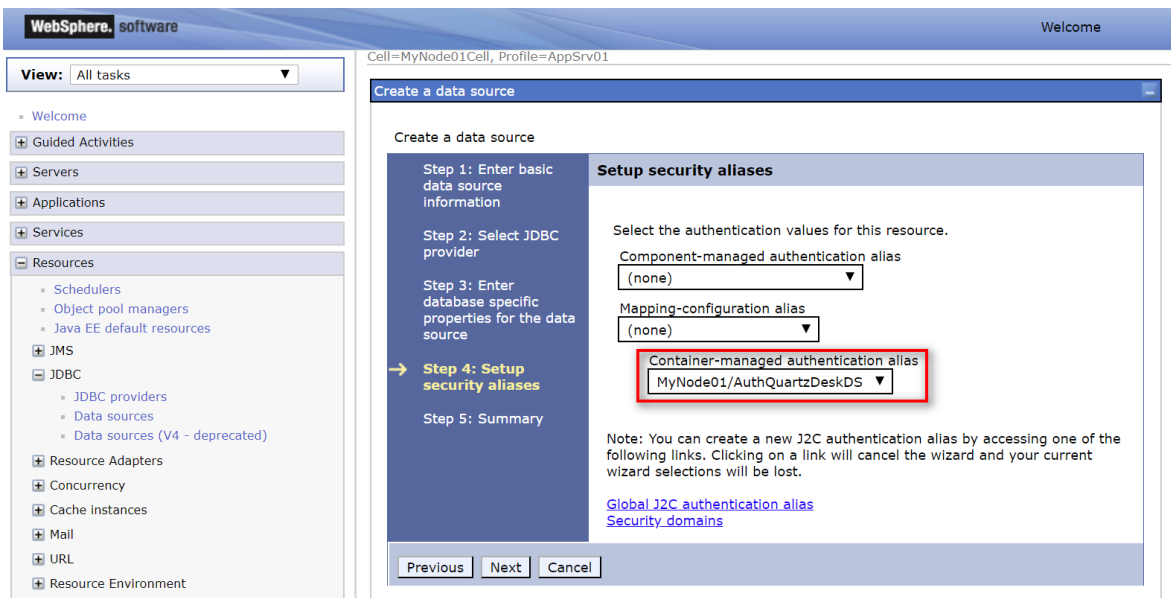


H2 is a light-weight Java database with limited fault tolerance and recovery functionality. We recommend using H2 for evaluation and experimental purposes only.

In Step 3, leave the “Data store helper class name” value as is and “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:
Statement cache size: 100

Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select "Validation by SQL query" with the following query:
select 1

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

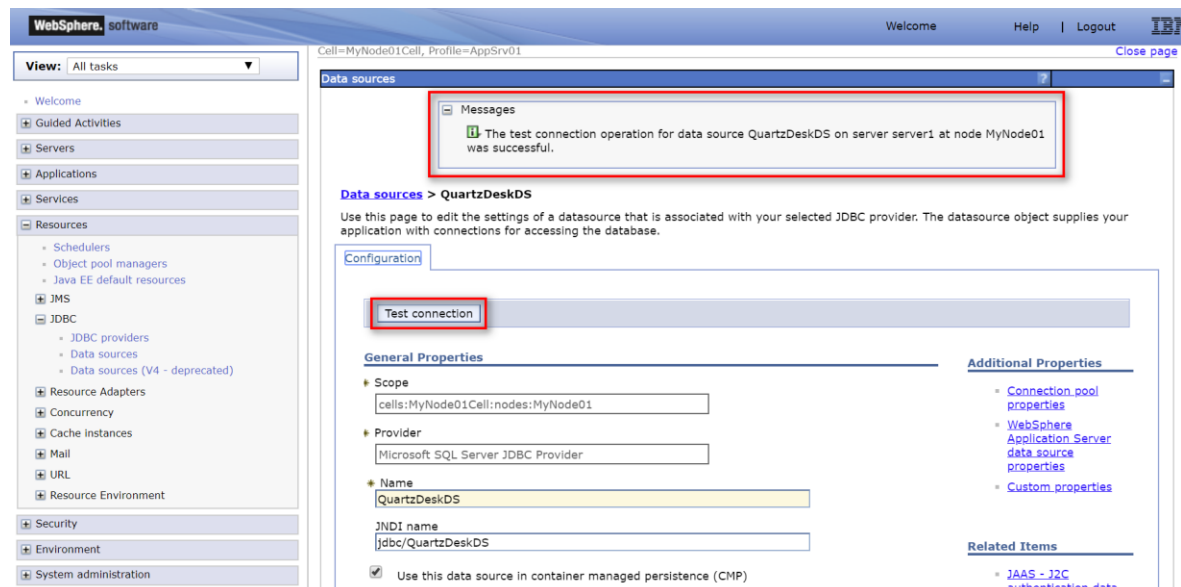
Name: URL

Value: jdbc:h2:file:<H2_DB_FILE_PATH>

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.

Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



Check there are no errors displayed.



When testing connection of a data source that uses a JDBC provider with the “Database type” set to “User type”, there may be a warning message displayed. In the JVM logs, the following message is logged:

```
DSRA0174W: Warning: GenericDataStoreHelper is being used.
```

You can safely ignore this warning.

4.5.3 Microsoft SQL Server

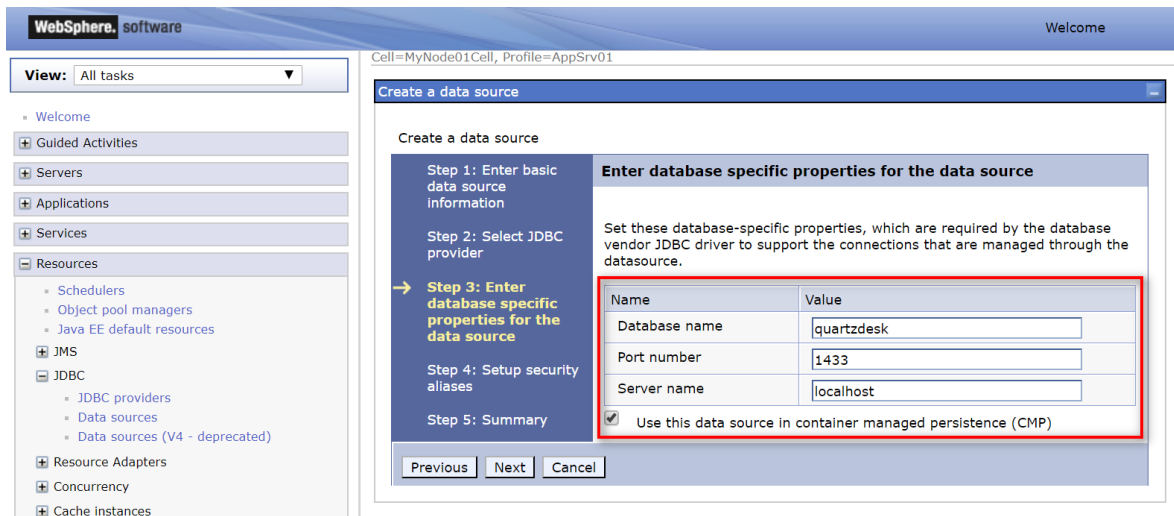
In Step 3, provide these values:

Database name: DB_NAME

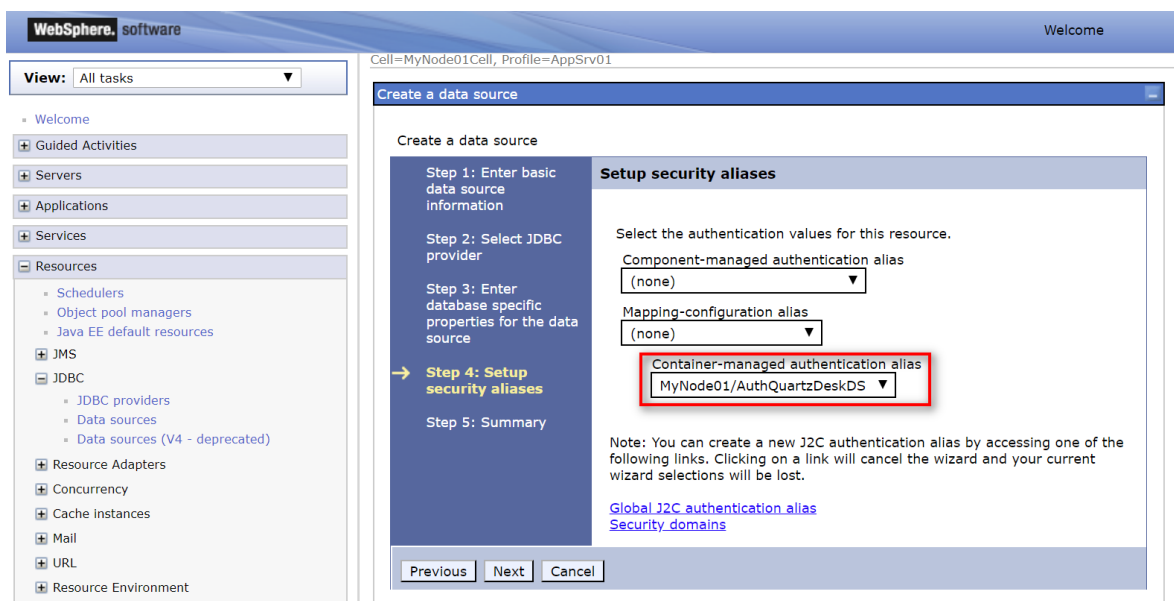
Port number: DB_PORT

Server name: DB_HOST

Leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the created data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:

Statement cache size: 100

Validate new connectons: checked

Number of retries: 100

Retry Interval: 3

Validate existing pooled connections: checked

Retry interval: 0

Select “Validation by SQL query” with the following query:

```
select 1
```

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

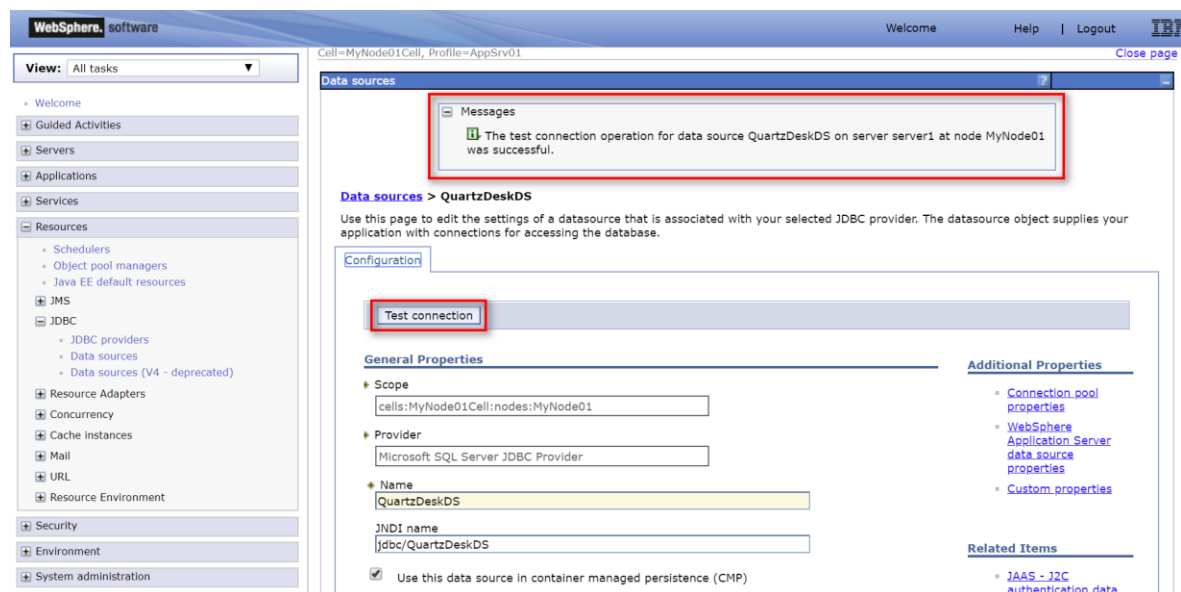
Name: applicationName

Value: QuartzDesk Web Application

Depending on your Microsoft SQL Server configuraton, you may need to set the value of the instanceName property. Read the property description for details.

Save changes.

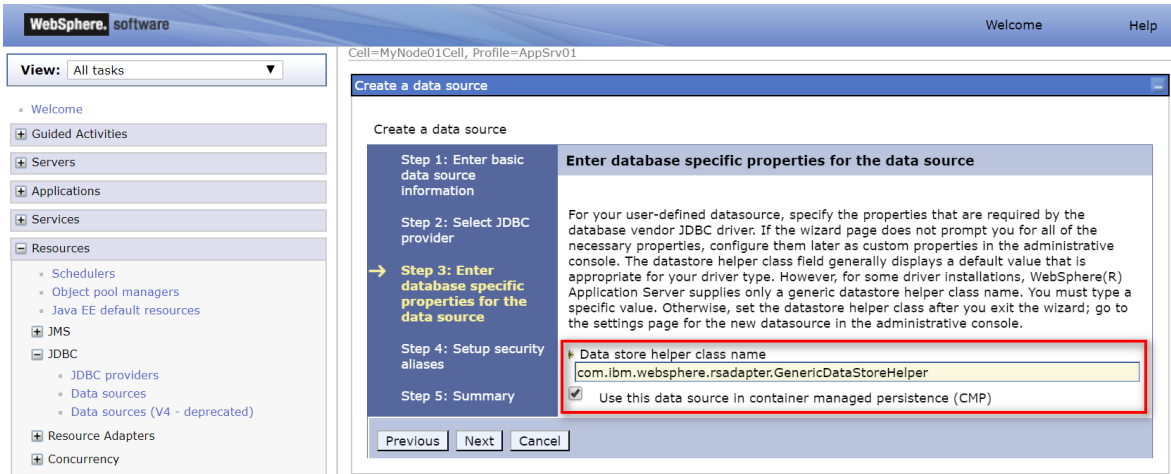
Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



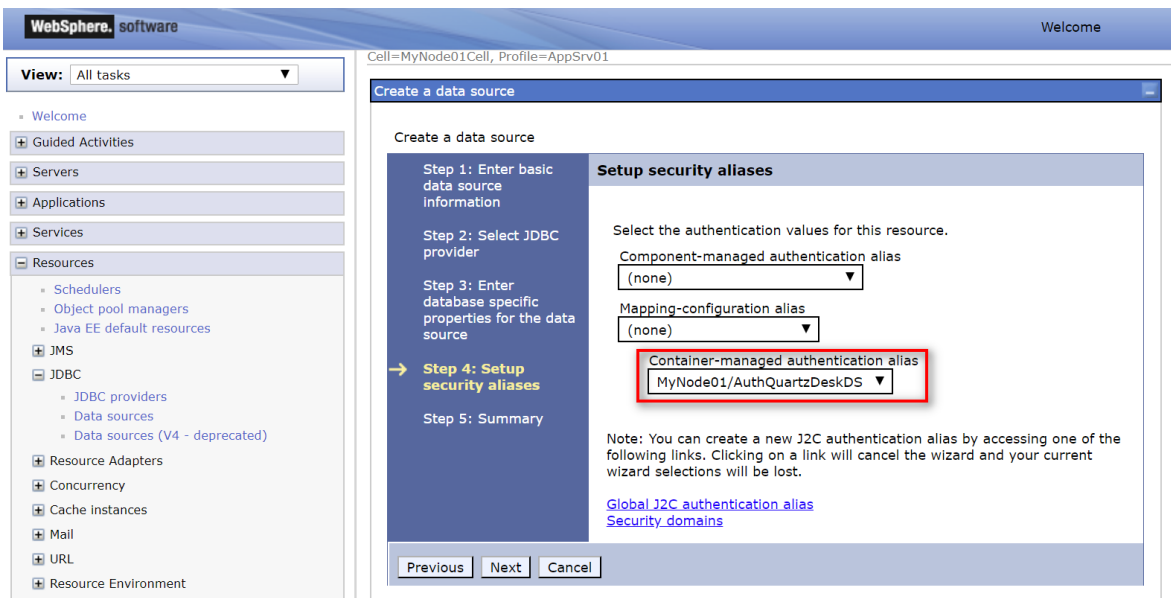
Check there are no errors displayed.

4.5.4 MySQL

In Step 3, leave the “Data store helper class name” value as is and “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:
Statement cache size: 100

Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select "Validation by SQL query" with the following query:
select 1

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

Name: databaseName

Value: DB_NAME

Name: serverName

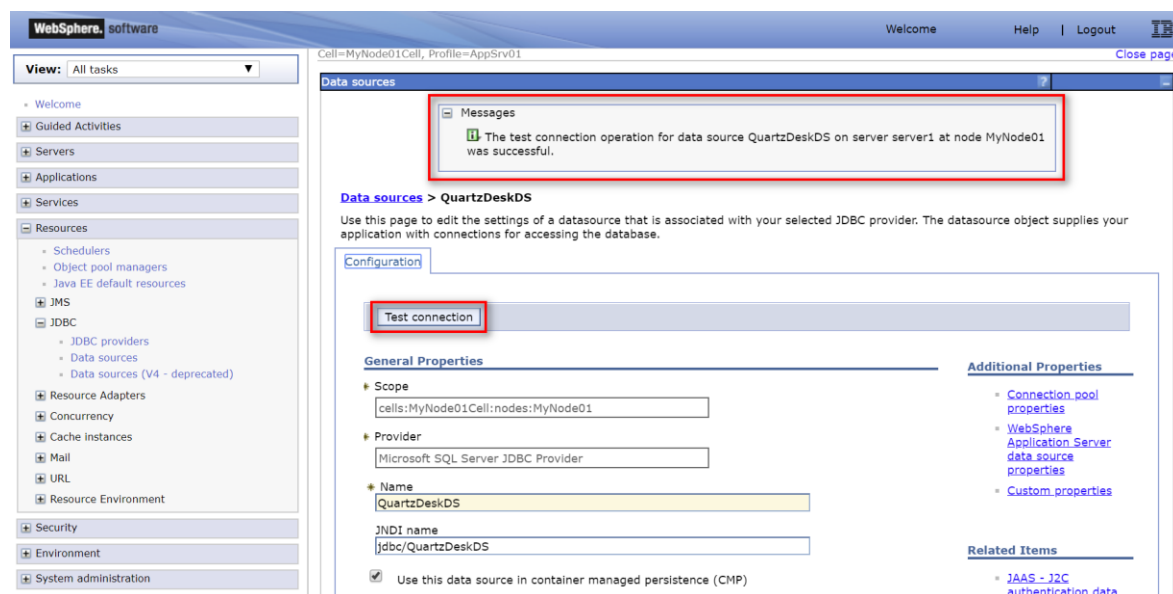
Value: DB_HOST

Name: portNumber

Value: DB_PORT

Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



Check there are no errors displayed.



When testing connection of a data source that uses a JDBC provider with the “Database type” set to “User type”, there may be a warning message displayed. In the JVM logs, the following message is logged:

DSRA0174W: Warning: GenericDataStoreHelper is being used.

You can safely ignore this warning.

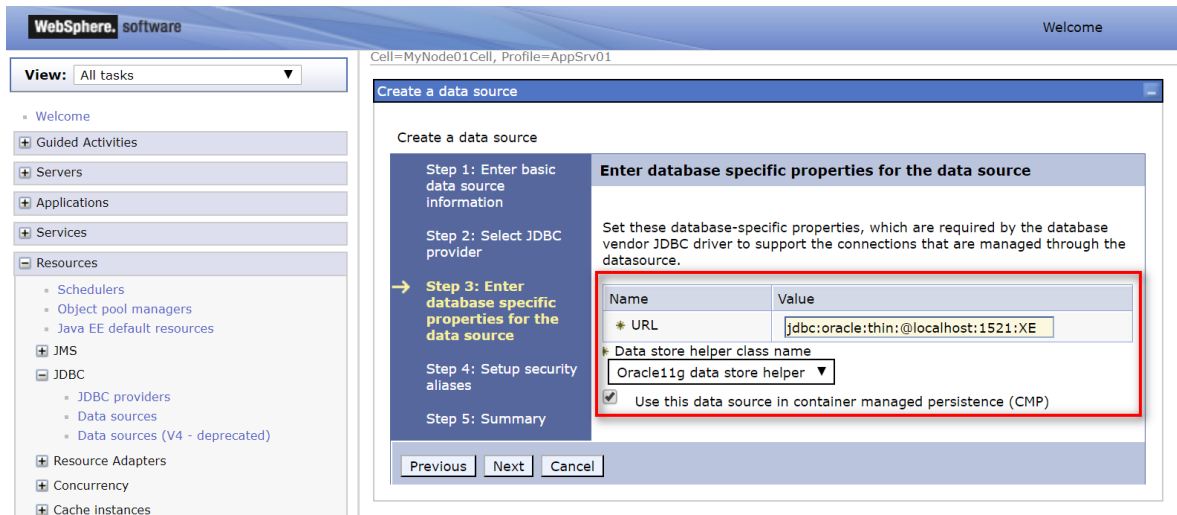
4.5.5 Oracle

In Step 3, provide the URL value:

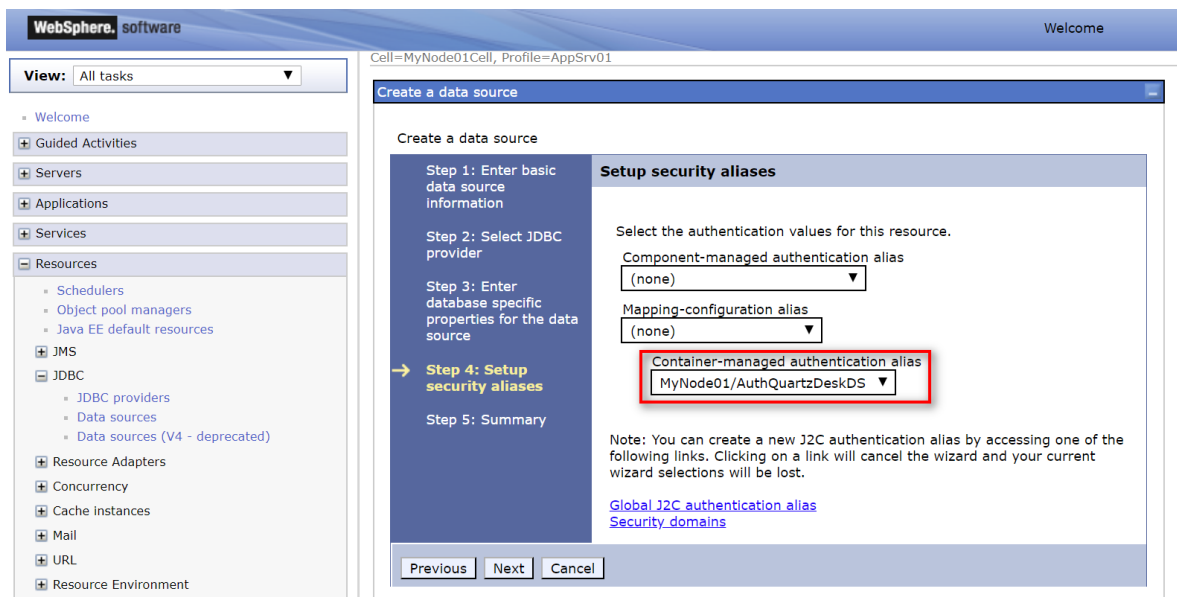
URL: jdbc:oracle:thin:@DB_HOST:DB_PORT:DB_NAME

Select the “Data store helper class name” based on your Oracle database version.

Leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the created data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:
Statement cache size: 100

Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select “Validation by SQL query” with the following query:
`select 1 from dual`

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

Name: driverType
Value: thin

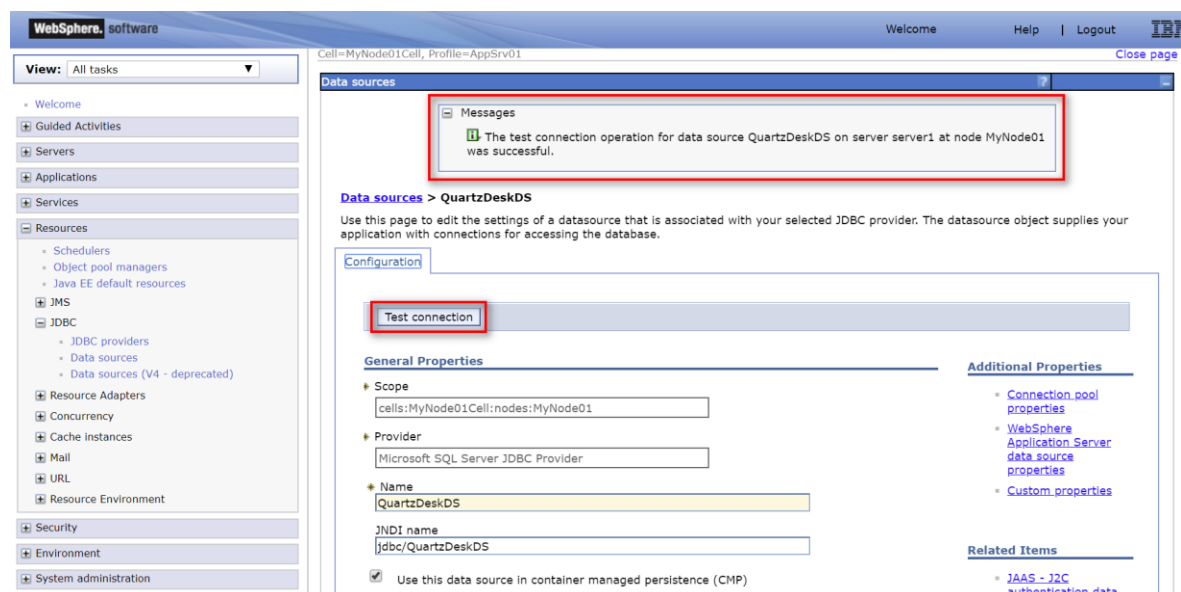
Name: databaseName
Value: DB_NAME

Name: serverName
Value: DB_HOST

Name: portNumber
Value: DB_PORT

Save changes.

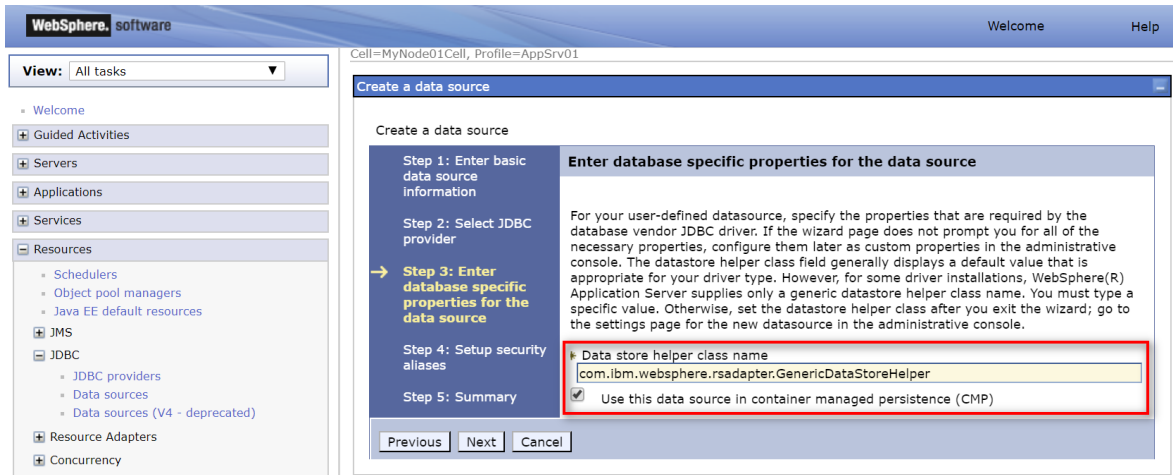
Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



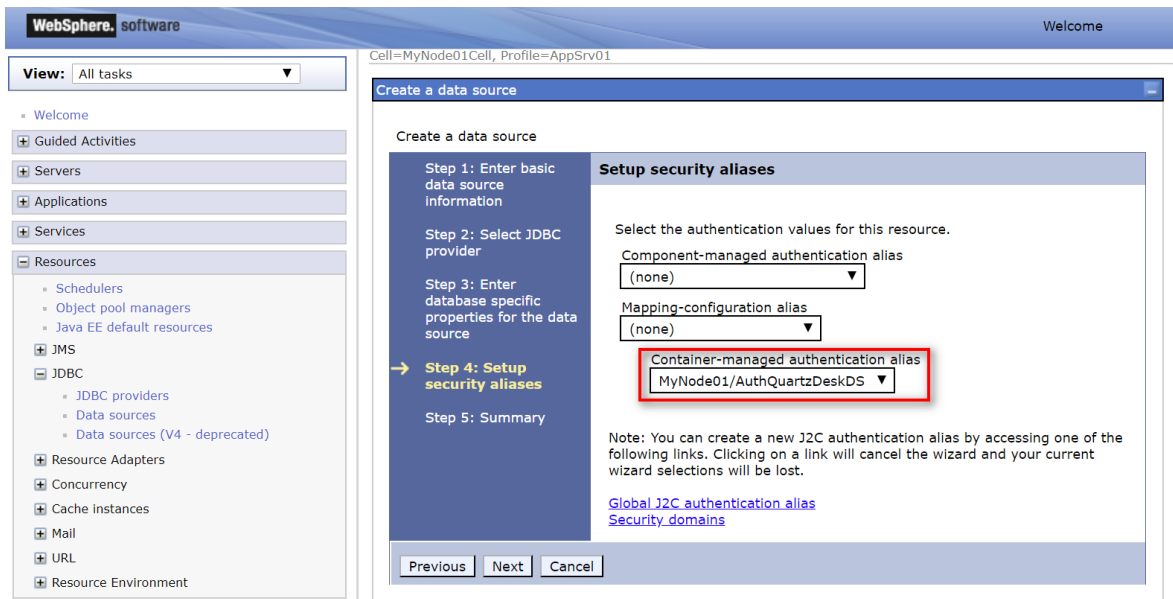
Check there are no errors displayed.

4.5.6 PostgreSQL

In Step 3, leave the “Data store helper class name” value as is and “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties.

Statement cache size: 100

Validate new connections: checked

Number of retries: 100

Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select “Validation by SQL query” with the following query:
select 1

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

Name: applicationName
Value: QuartzDesk Web Application

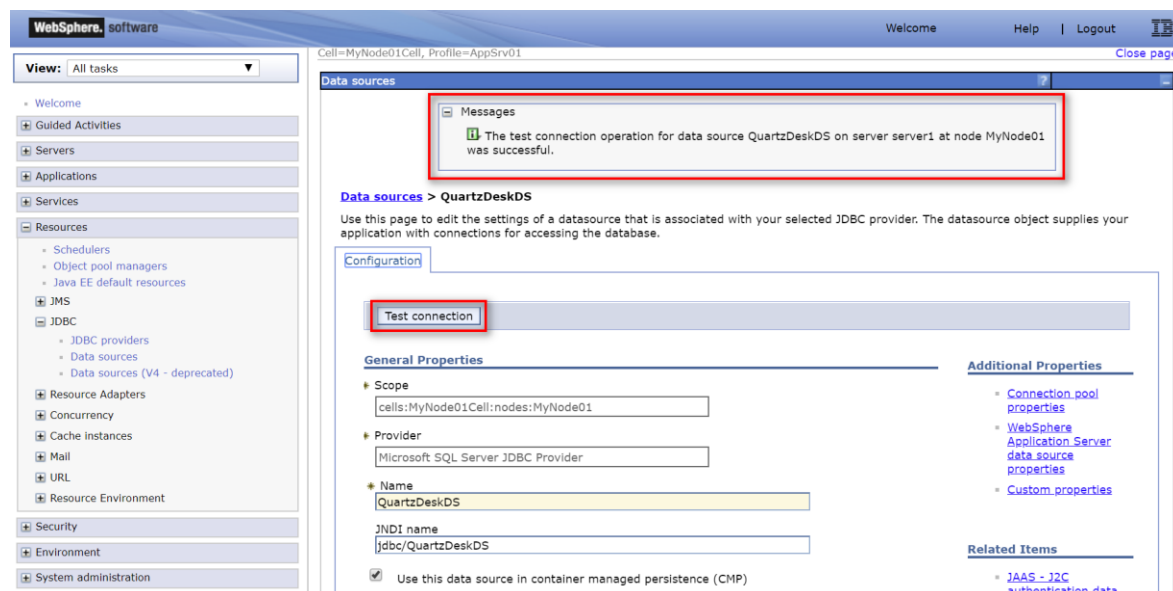
Name: databaseName
Value: DB_NAME

Name: serverName
Value: DB_HOST

Name: portNumber
Value: DB_PORT

Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



The screenshot shows the WebSphere Administration Console interface. On the left is a navigation tree with 'Resources' expanded to 'JDBC'. The main content area displays the configuration for 'QuartzDeskDS'. A message box at the top indicates a successful test connection. Below it, a 'Test connection' button is highlighted. The configuration page includes sections for 'General Properties' and 'Additional Properties'. The 'General Properties' section contains fields for Scope, Provider, Name, and JNDI name. The 'Additional Properties' section lists links for 'Connection pool properties', 'WebSphere Application Server data source properties', and 'Custom properties'. A checkbox at the bottom is checked for 'Use this data source in container managed persistence (CMP)'.

Check there are no errors displayed.



When testing connection of a data source that uses a JDBC provider with the “Database type” set to “User type”, there may be a warning message displayed. In the JVM logs, the following message is logged:

DSRA0174W: Warning: GenericDataStoreHelper is being used.

You can safely ignore this warning.

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 writable by the user the WAS process runs under.

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



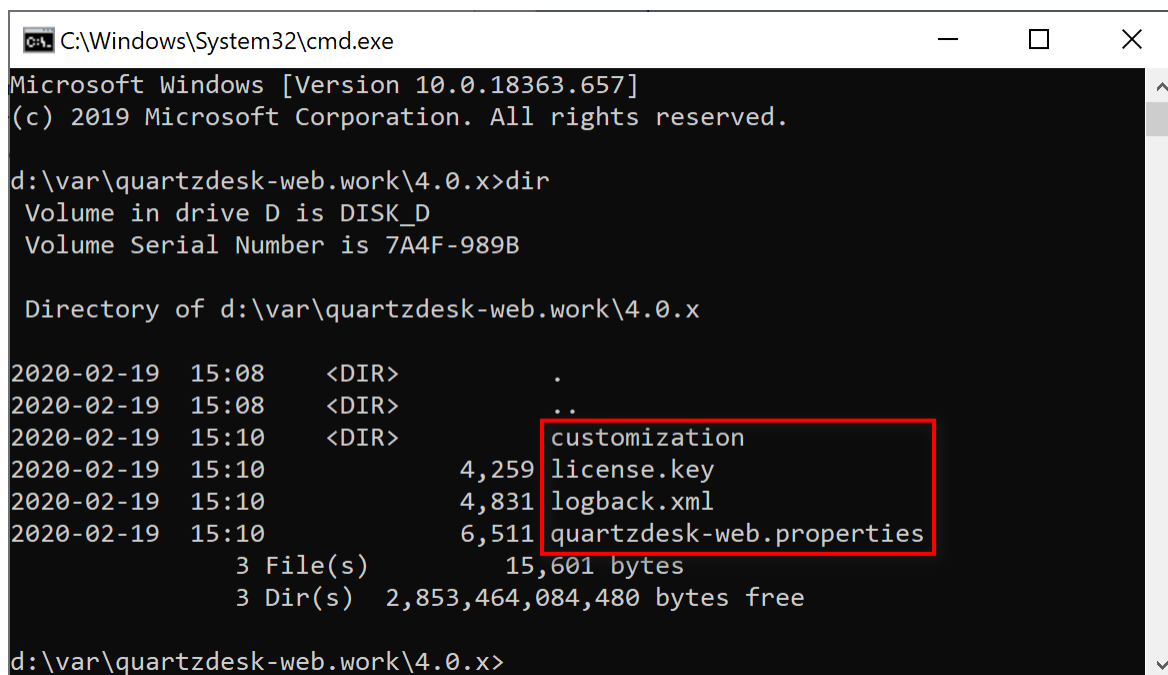
You can obtain a free 30-day trial license key at www.quartzdesk.com (go to Try / Purchase > Get Trial License Key).

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.



```
C:\Windows\System32\cmd.exe
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

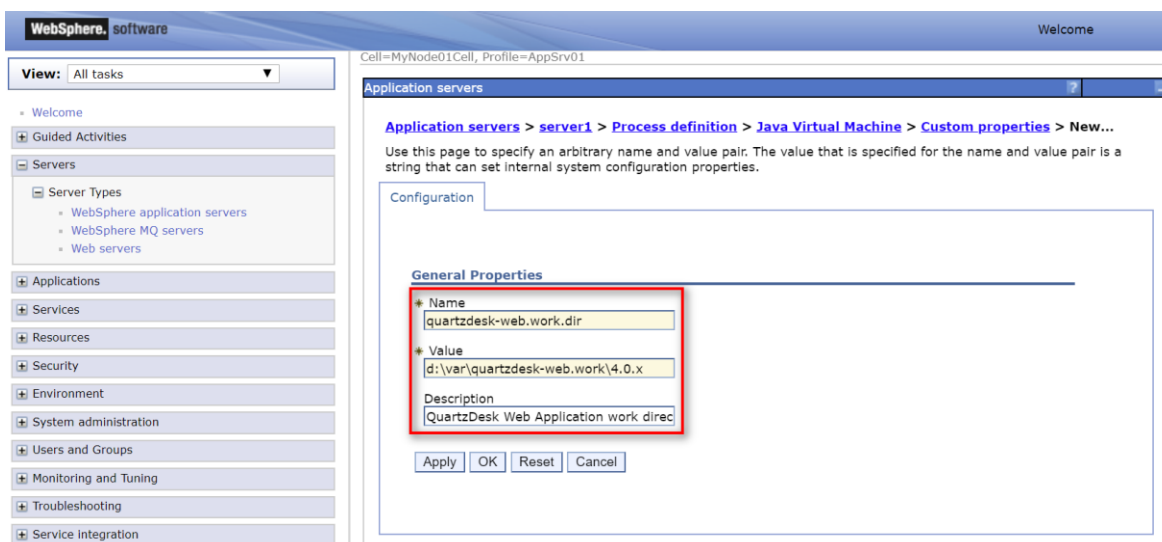
d:\var\quartzdesk-web.work\4.0.x>
```

In WAC open Servers → Server Types → WebSphere application servers → WAS_SERVER_NAME → Java and Process Management → Process Definition → Java Virtual Machine → Custom Properties. Add a new JVM system property:

Name: quartzdesk-web.work.dir

Value: WORK_DIR

Description: QuartzDesk Web Application work directory.



Apply and Save changes.

Restart WAS for the changes to take effect.

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 application starts. Please refer to the [Logback Manual](#) for Logback configuration details.

4.8 Deploy Application

In WAC go to Applications → Application Types → WebSphere enterprise applications.

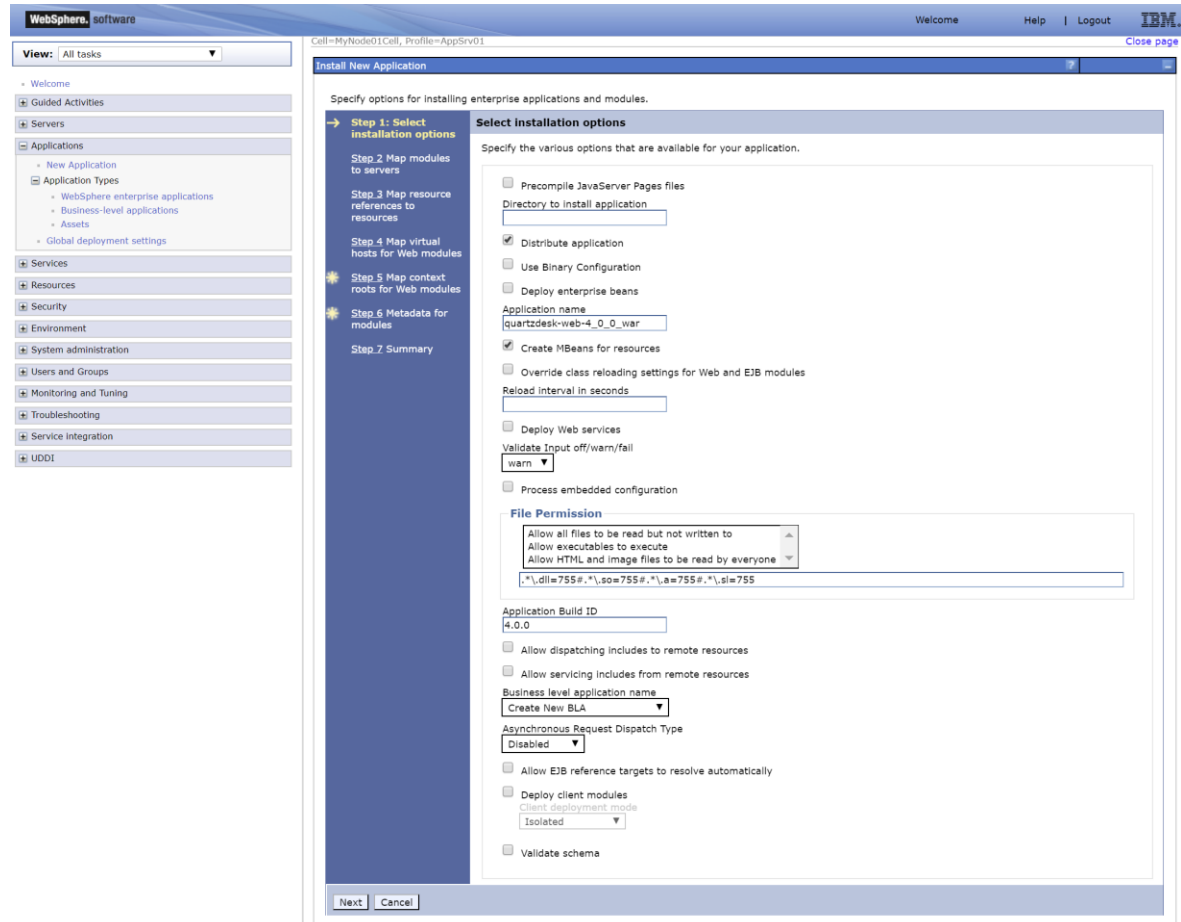
Click the Install button and select the `quartzdesk-web-x.y.z.war` file.

Click Next.

Leave the “Fast Path” radio button selected. Click Next.

Step 1 – Select installation options

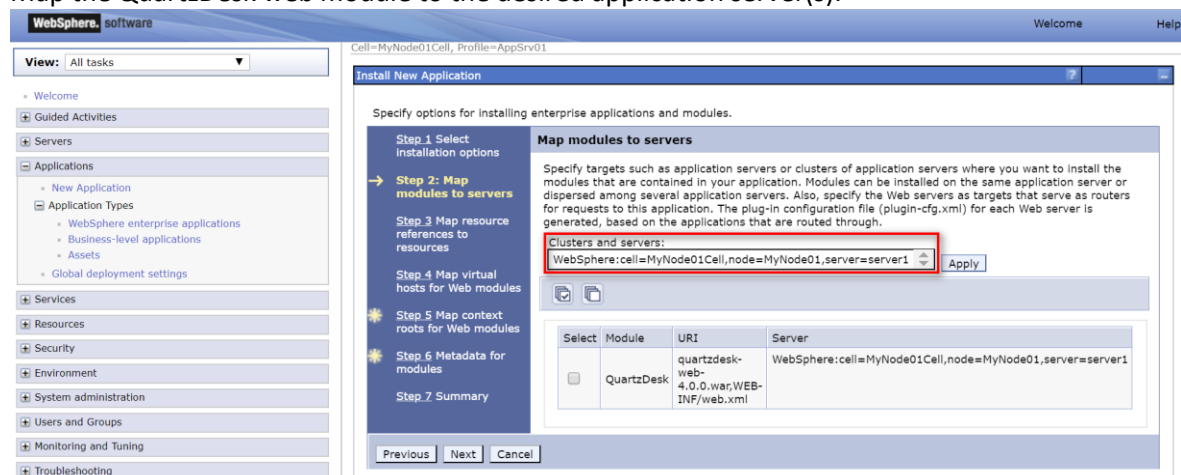
Click Next.



The screenshot shows the 'Install New Application' wizard in the WebSphere Admin Console. The left sidebar contains a navigation tree with 'Applications' selected. The main panel is titled 'Select installation options' and contains various checkboxes and input fields for configuring the application. The 'Fast Path' radio button is selected. The 'File Permission' section shows a dropdown menu set to 'Allow all files to be read but not written to'. The 'Application Build ID' is set to '4.0.0'. The 'Business level application name' is set to 'Create New BLA'. The 'Asynchronous Request Dispatch Type' is set to 'Disabled'. The 'Deploy client modules' dropdown is set to 'Isolated'. The 'Validate schema' checkbox is checked. The 'Next' button is visible at the bottom.

Step 2 – Map modules to servers

Map the QuartzDesk web module to the desired application server(s).



The screenshot shows the 'Install New Application' wizard in the WebSphere Admin Console, Step 2: Map modules to servers. The left sidebar is the same as in Step 1. The main panel is titled 'Map modules to servers' and contains a table for mapping modules to servers. The 'Clusters and servers' dropdown is set to 'WebSphere:cell=MyNode01Cell,node=MyNode01,server=server1'. The table below shows the mapping for the 'QuartzDesk' module.

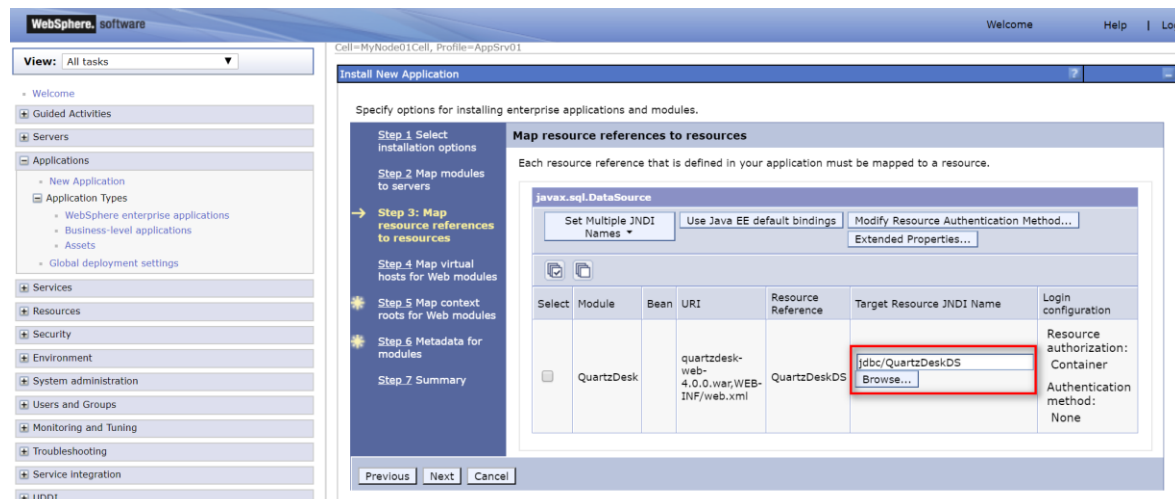
Select	Module	URI	Server
<input type="checkbox"/>	QuartzDesk	quartzdesk-web-4.0.0.war;WEB-INF/web.xml	WebSphere:cell=MyNode01Cell,node=MyNode01,server=server1

The 'Apply' button is visible next to the 'Clusters and servers' dropdown. The 'Previous', 'Next', and 'Cancel' buttons are visible at the bottom.

Click Next.

Step 3 – Map resource references to resources

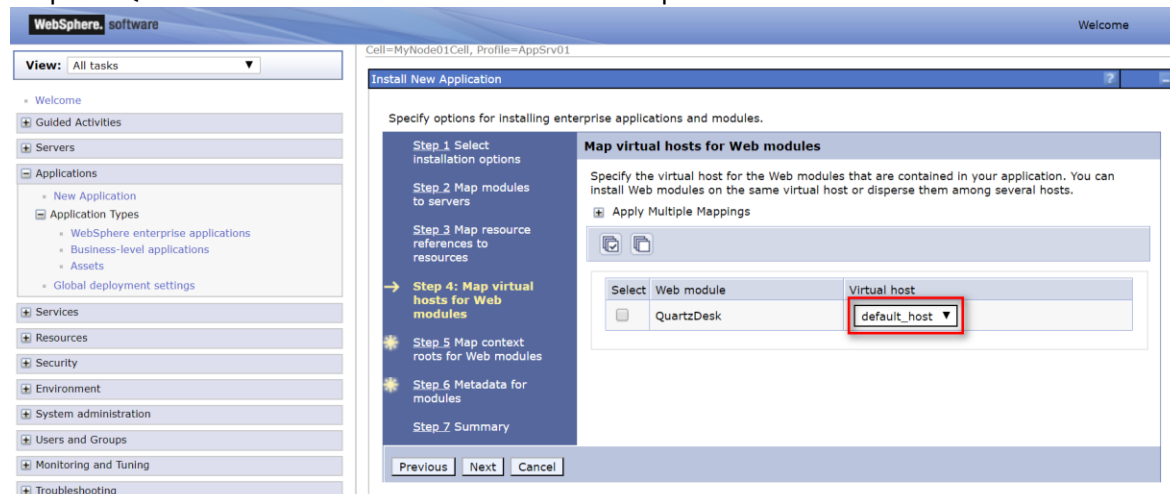
Map the QuartzDeskDS data source reference to the JNDI name of the QuartzDesk data source created in 4.5.



Click Next.

Step 4 – Map virtual hosts for Web modules

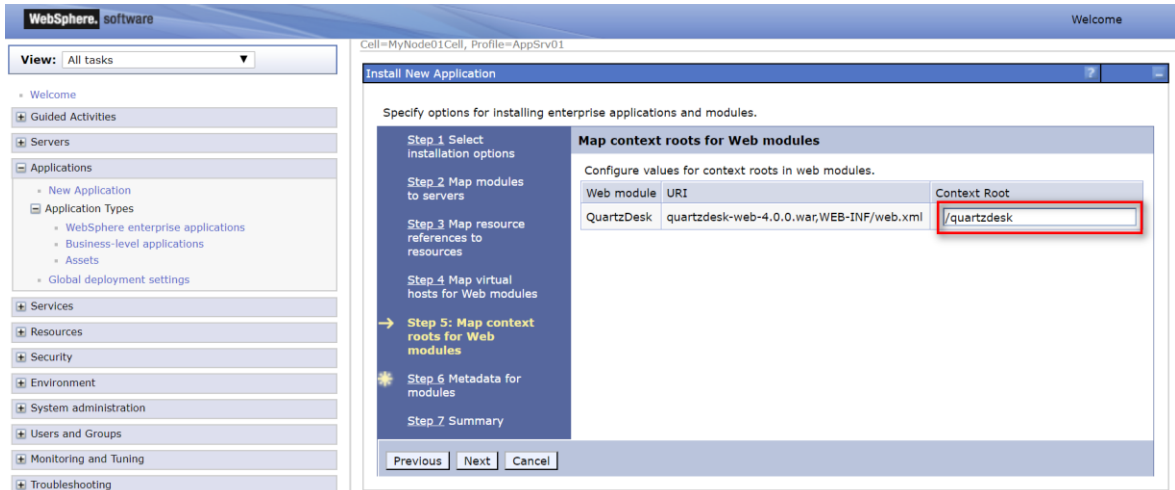
Map the QuartzDesk web module to the desired WebSphere virtual host.



Click Next.

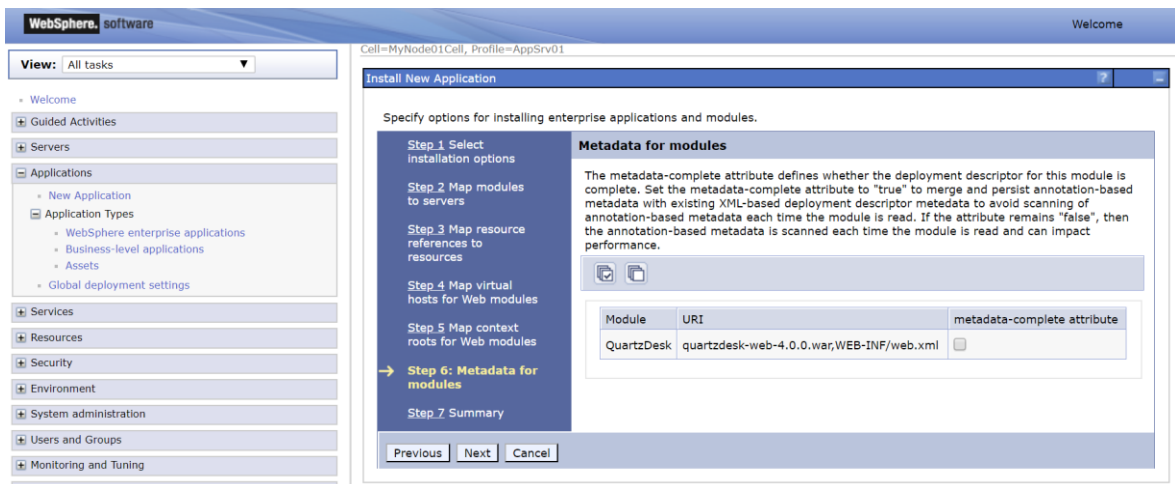
Step 5 – Map context roots for Web modules

Provide the web servlet context root for the QuartzDesk Web Application. We recommend using "/quartzdesk" (without quotes).



Click Next.

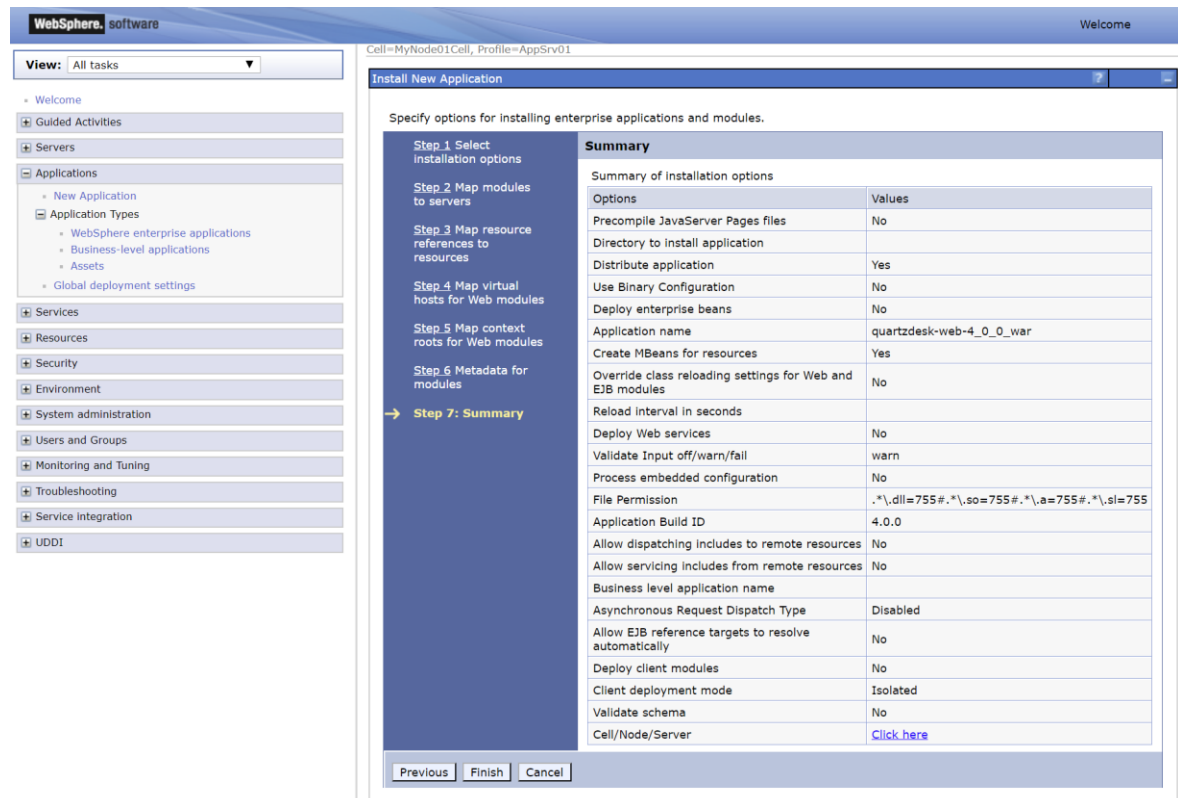
Step 6 – Metadata for modules



Click Next.



Step 7 – Summary



WebSphere, software | Welcome

Cell=MyNode01Cell, Profile=AppSrv01

Install New Application

Specify options for installing enterprise applications and modules.

Step 1. Select installation options
Step 2. Map modules to servers
Step 3. Map resource references to resources
Step 4. Map virtual hosts for Web modules
Step 5. Map context roots for Web modules
Step 6. Metadata for modules
→ Step 7: Summary

Summary

Summary of installation options

Options	Values
Precompile JavaServer Pages files	No
Directory to install application	
Distribute application	Yes
Use Binary Configuration	No
Deploy enterprise beans	No
Application name	quartzdesk-web-4_0_0_war
Create MBeans for resources	Yes
Override class reloading settings for Web and EJB modules	No
Reload interval in seconds	
Deploy Web services	No
Validate Input off/warn/fail	warn
Process embedded configuration	No
File Permission	.*\,dl=755#.*\,so=755#.*\,a=755#.*\,sl=755
Application Build ID	4.0.0
Allow dispatching includes to remote resources	No
Allow servicing includes from remote resources	No
Business level application name	
Asynchronous Request Dispatch Type	Disabled
Allow EJB reference targets to resolve automatically	No
Deploy client modules	No
Client deployment mode	Isolated
Validate schema	No
Cell/Node/Server	Click here

Previous Finish Cancel

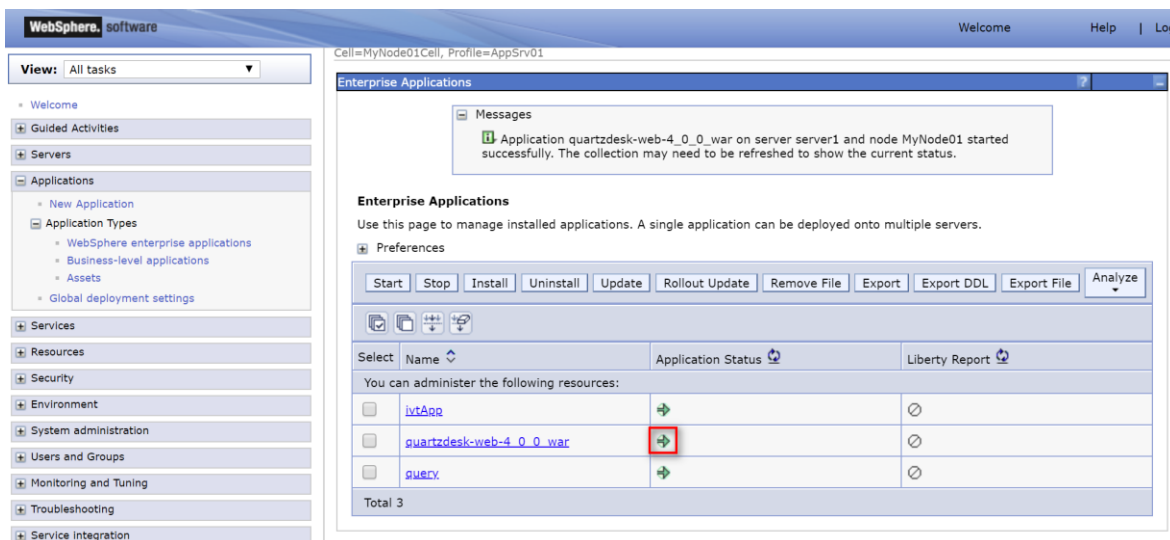
Click Finish.

When the installation completes, Save changes.

4.9 Start Application

In WAC go to Applications → Application Types → WebSphere enterprise applications. Select the checkbox next to the QuartzDesk Web Application in the Enterprise Applications list. Click the Start button and wait for the startup procedure to complete.

Upon successful starting, the Application Status flag, shown next the QuartzDesk Web Application's name in the Enterprise Applications list, turns green.



Check WAS logs under `WAS_SERVER_PROFILE/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://WAS_HTTP_HOST:WAS_HTTP_PORT/quartzdesk/ and verify that the QuartzDesk Web Application's GUI is accessible.

Check the version number of deployed QuartzDesk Web Application.



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.

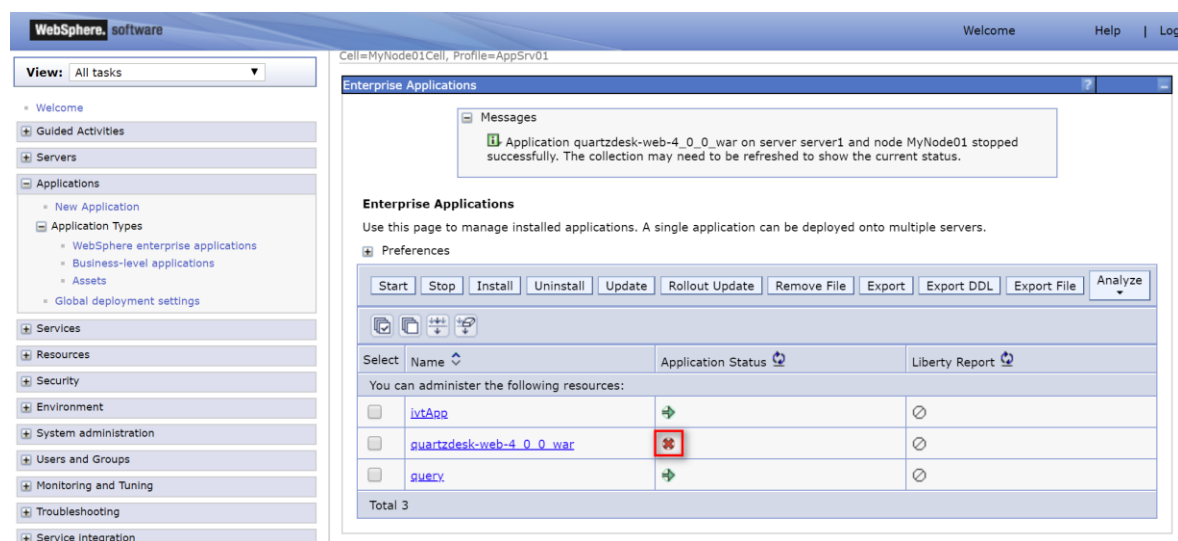


5. Upgrading

5.1 Stop Existing Application

In WAC go to Applications → Application Types → WebSphere enterprise applications. Select the checkbox next to the QuartzDesk Web Application in the Enterprise Applications list. Click the Stop button at the top of the list. Wait for the action to complete.

Upon successful stopping, the Application Status flag, shown next the existing QuartzDesk Web Application in the Enterprise Applications list, turns red.



The screenshot shows the WebSphere Enterprise Applications management console. The 'Enterprise Applications' tab is active, displaying a table of installed applications. The application 'quartzdesk-web-4_0_0_war' is selected, and its status is shown as a red flag with a star, indicating it is stopped. A message box above the table states: 'Application quartzdesk-web-4_0_0_war on server server1 and node MyNode01 stopped successfully. The collection may need to be refreshed to show the current status.'

Select	Name	Application Status	Liberty Report
<input type="checkbox"/>	lvApp	➔	⊗
<input checked="" type="checkbox"/>	quartzdesk-web-4_0_0_war	🚩	⊗
<input type="checkbox"/>	quark	➔	⊗

Total 3

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.

Backup the QuartzDesk Web Application in WAC by going to Applications → Application Types → WebSphere enterprise applications. Select the QuartzDesk Web Application by selecting the checkbox on the line. Click the Export button and wait for the export to complete. Download the exported WAR file.

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 WAC go to Applications → Application Types → WebSphere enterprise applications. Select the checkbox next to the QuartzDesk Web Application in the Enterprise Applications list. Click the Uninstall button at the top of the list. Wait for the action to complete and Save the changes when prompted.

Upon successful removal, the QuartzDesk Web Application disappears from the Enterprise Applications list.

5.4 Deploy New Application

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

5.5 Start New Application

Start the new version of the 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 WAS is configured to use Java 7 or higher.

6.2 Rename Configuration File

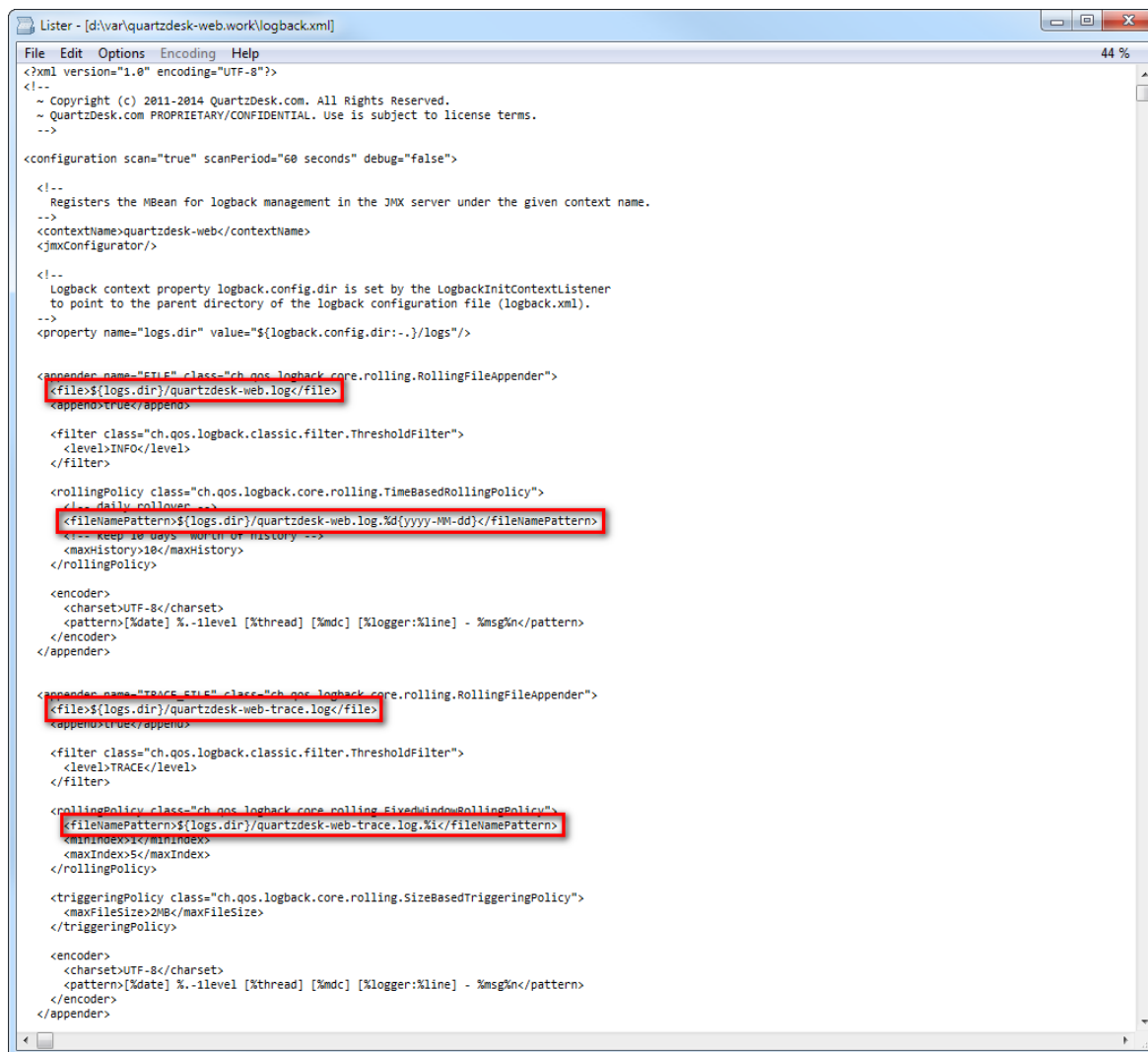
The name of the QuartzDesk Web Application 3.x configuration file has changed from `quartzdesk.properties` to `quartzdesk-web.properties`. Rename the file located in the QuartzDesk Web Application work directory.

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)
<code>quartzdesk.log</code>	<code>quartzdesk-web.log</code>
<code>quartzdesk-trace.log</code>	<code>quartzdesk-web-trace.log</code>

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



```

Listner - [d:\var\quartzdesk-web.work\logback.xml]
File Edit Options Encoding Help
<?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] %.-level [%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>

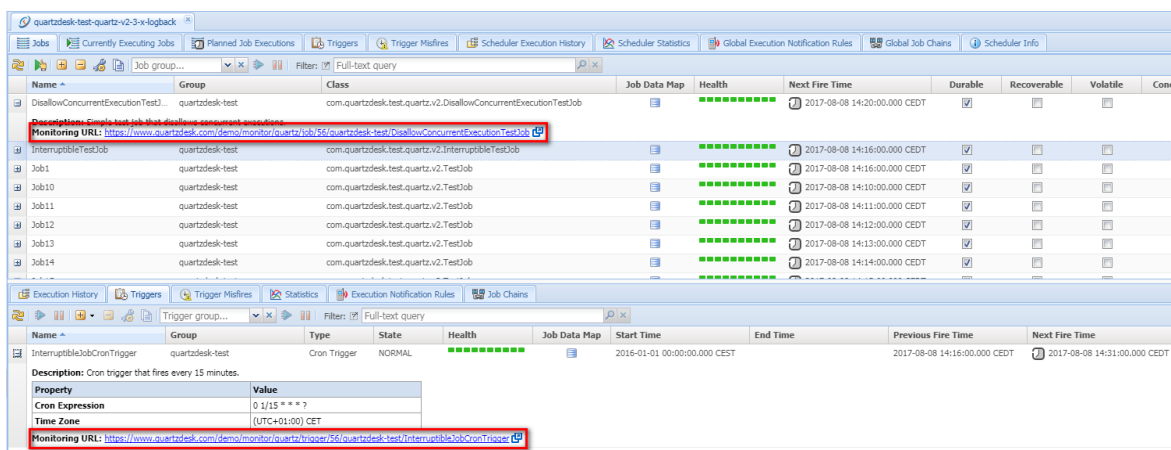
    <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
      <maxFileSize>2MB</maxFileSize>
    </triggeringPolicy>

    <encoder>
      <charset>UTF-8</charset>
      <pattern>[%date] %.-level [%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 use it to overwrite the existing configuration file in `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.



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).



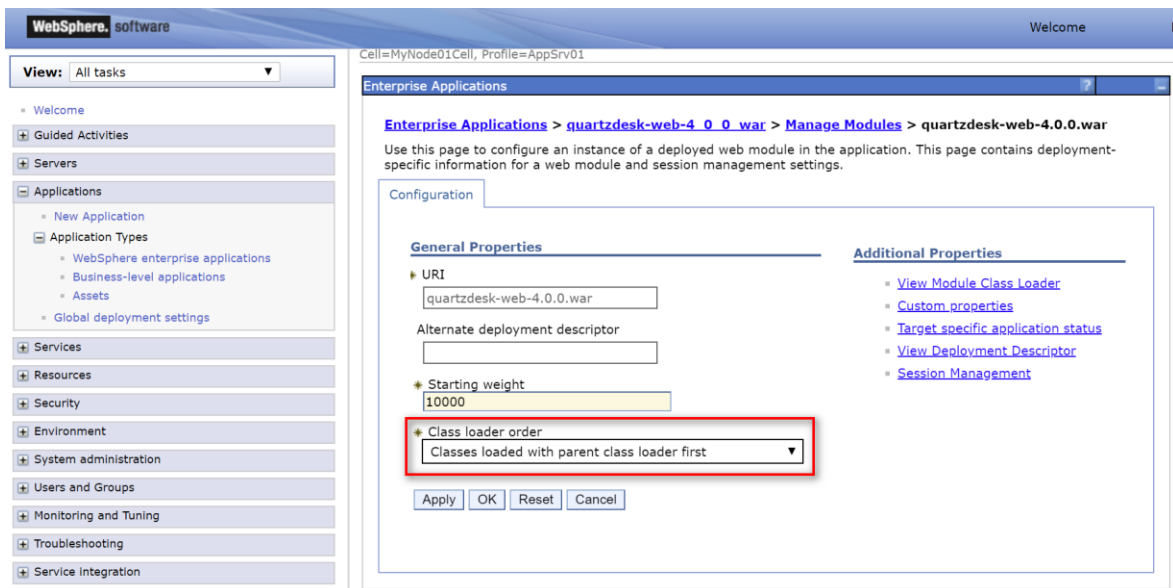
All JAX-WS web service endpoints in QuartzDesk Web Application 3.x support the HTTP Basic authentication scheme where the user’s authentication credentials are passed in the `Authorization` HTTP header. Please note that the same authentication scheme was used by JAX-WS endpoints in QuartzDesk Web Application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

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

7.1 Class Loader Order

QuartzDesk Web Application 4.x requires the class loader order to be set to “parent-first”. In WAS, this is the default class loader order when a new web application is deployed.

In WAC go to Applications → Application Types → WebSphere enterprise applications → quartzdesk-web_x_y_z_war → Manage Modules → QuartzDesk. In Class loader order box, select “Classes loaded with parent class loader first” option.



The screenshot shows the IBM WebSphere Administration Console interface. The left-hand navigation pane is expanded to 'Applications' > 'Application Types' > 'WebSphere enterprise applications'. The main content area displays the configuration for the web module 'quartzdesk-web-4.0.0.war'. Under the 'Configuration' tab, the 'Class loader order' dropdown menu is highlighted with a red box and set to 'Classes loaded with parent class loader first'. Other visible settings include URI, Alternate deployment descriptor, and Starting weight.

Click OK and Save changes.

8. Cluster Deployment Notes

When deploying QuartzDesk Web Application to a WebSphere 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 WebSphere instance that handled the first user request that established the session.

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

8.2 Shared Work Directory

We recommend that you put the QuartzDesk Web Application work directory, described in 4.4, 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 required 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 the 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 the 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
<code>org.quartz.simpl.HostnameInstanceIdGenerator</code>	<p>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.</p> <p>This is the default implementation used by the QuartzDesk Web Application. No configuration changes are necessary to use this instance ID generator.</p>
<code>org.quartz.simpl.SystemPropertyInstanceIdGenerator</code>	<p>This implementation is suitable for QuartzDesk Web Application deployments where some of the clustered QuartzDesk Web Application instances run on the same host.</p> <p>This implementation extracts the Quartz scheduler instance ID from the <code>org.quartz.scheduler.instanceId</code> JVM system property that must be explicitly set.</p> <p>Please refer to the WebSphere documentation for details on how to add a new JVM system property.</p>

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.

