PYMMA

OpenESB Standalone Edition

Database connection with JNDI

V1.1

PYMMA

Document identifier:

Pymma document: 770-005

Location:

www.pymma.com

Editor:

Pymma Services: contact@pymma.com

Abstract:

This document provides a guide to connect OpenESB to databases with JNDI

Status:

This document is in version 1.1

ABOUT PYMMA CONSULTING

Pymma Services is a technical architect bureau founded in 1999 and headquartered in London, United Kingdom. It provides expertise in service oriented integration systems design and implementation. Leader of OpenESB project, Pymma is recognized as one of the main actors in the integration landscape. It deeply invests in open source projects such as Drools rules engine. Pymma is a European company based in London with regional offices in France, Belgium and Canada. (contact@pymma.com or visit our website at www.pymma.com)

Copyright

Disclaimer

Information in this document is subject to change without notice and does not represent a commitment on the part of Pymma Services LTD. This manual is provided “as is” and Pymma is not responsible for disclaims all warranties of any kind with respect to third-party content, products, and services. Pymma will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Trademark Notice

Pymma is a registered of Pymma Engineering LTD. Java is a registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Contents

1 Introduction

Some OpenESB components require a JNDI context to access to external resources such as databases or Message Brokers. OpenESB legacy editions embedded in a JEE container benefited from a JNDI context provided by the application server. OpenESB standalone does not require any additional container to run and uses a different mechanism to retrieve a JNDI context. The aim of this document is to explain how to set up a JNDI context for OpenESB SE.

Our implementation is for a main part based on Apache libraries (Apache Tomcat Naming / DS pooling). We associate Apache naming libraries, the new Apache pool and an advanced introspection development to provide flexible and efficient JNDI features, and databases pooling.

In this document, ${OESE_HOME} is the directory where the OpenESB Standalone edition is installed. On Windows, if ${OESE_HOME} = “F:\OpenESB-SE-3.0” Then components will be located in ${OESE_HOME}\components or “F:\OpenESB-SE-3.0”\component. OpenESB SE has three main subdirectories:

1. ${OESE_HOME}\components = where the component are located
2. ${OESE_HOME}\instance = where the instance of OE SE is located
3. ${OESE_HOME}\studio = where OpenESB Studio is located

2 JNDI configuration file

OE Standalone Edition JNDI Configuration is defined in the file ${OESE-HOME}\OE-Instance\config\context.xml.

Context file can be changed. To do it, modify the content of ${OESE-HOME}\OE-Instance\config\openesb.yaml.

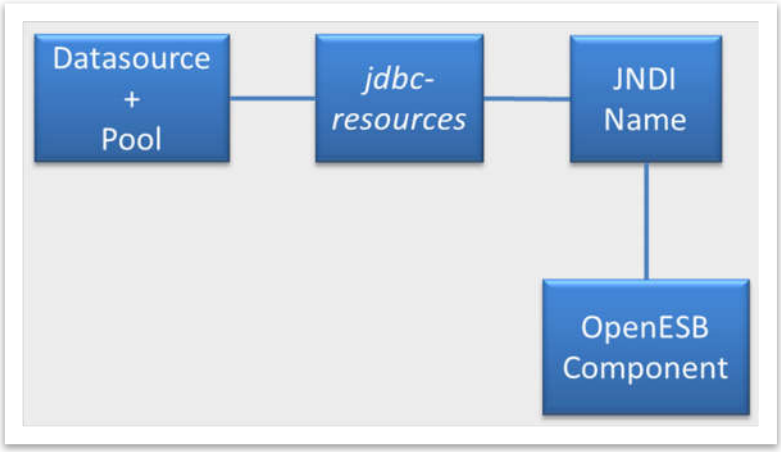
```
22  ################################### JNDI ######################################
23  # Specify the context file to used for JNDI
24  # jndi.context: ${openesb.home}/config/context.xml
25
```

Contrary to legacy contexts, this XML document is defined by an XML schema named context. xsd. The schema can be found in the same directory than context.xml

Context.xsd defines two main elements: Datasource-pool-properties and JDBC resources.

Datasource-pool-properties elements contain both Datasources properties and Connection pool configurations.

Jdbc-resource links a Datasource-pool-properties element with a JNDI name. OpenESB components access to a Datasource through a JNDI name.



3 Configure a JNDI context

A six steps process is required to set up OpenESB context.

1. Select a native datasource implementation
2. Choose the datasource properties
3. Create a Datasource-pool-properties
4. Set up native datasource properties
5. Set up pool properties
6. Create a jdbc-resource

3.1 Select the native datasource implementation

Database editors provide Java Datasource implementations to create connections to their databases. JDBC specifications define two classes to connect Java applications to databases.

DriverManager: This fully implemented class connects an application to a data source, which is specified by a database URL. When this class first attempts to establish a connection, it automatically loads any JDBC 4.0 drivers found within the classpath.

DataSource: This interface is preferred over DriverManager because it allows details about the underlying data source to be transparent to your application. A DataSource object's properties are set so that it represents a particular data source.

Using Java driver is more straightforward but less sophisticated than using Java datasource, so we decided to use datasource objects to connect OpenESB components to relational databases.

To configure OpenESB context.xml, you must find out the native implementation of javax.sql.Datasource provided by the editor of your database. If you want to use XA transaction, search for an implementation of javax.sql.XADatasource.

The table below provides Datasource and XADatasource names for the most common databases

Database	Datasource	XADatasource
Java DB	org.apache.derby.jdbc.ClientDataSource org.apache.derby.jdbc.ClientDataSource40	org.apache.derby.jdbc.ClientXADataSource org.apache.derby.jdbc.ClientXADataSource40
MySQL	com.mysql.jdbc.jdbc2.optional.MysqlDataSource	com.mysql.jdbc.jdbc2.optional. MysqlXADataSource
DB2	com.ibm.db2.jdbc.db2DataSource	com.ibm.db2.jdbc.db2XADataSource
Postgres	org.postgresql.ds.PGSimpleDataSource	org.postgresql.xa.PGXADataSource
Oracle	oracle.jdbc.pool.OracleDataSource	oracle.jdbc.xa.client OracleXADataSource

Some Datasource implementations offer you to use pooled datasources. We recommend you not to use pooled datasources with OpenESB since we have already embedded a powerful pool in our implementation. Pools of pools often generate unexpected behaviours.

3.1.1 Jar files and OpenESB classpaths

The jar file containing your datasources drivers must be declared either in the OpenESB classpath or the component classpath that will use access to a database.

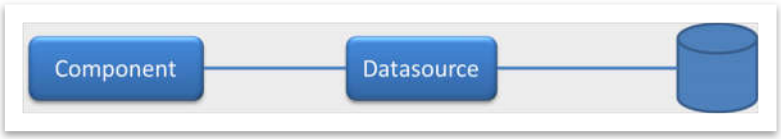
If you want to add an external library to the OpenESB classpath, copy the jar file to ${OESE_HOME}\OE-Instance\lib\ext.

If the library is just used by one component, copy the jar file to ${OESE_HOME}\OE-Instance\lib\ext***ComponentName.*** For example, if you want to add the jar file to sun-jms-binding classpath, copy it to ${OESE_HOME}\OE-Instance\libext\ sun-jms-binding.

You need to restart OpenESB or the component to take into account the external library. Before restarting, OpenESB context must be setup correctly.

3.2 Choose the datasource properties you want to set up

To set up the connection between an OpenESB component and a database, you have to set up datasource properties.



Each datasource has its own properties. Some are well known such as user, password, database names… but, some datasources can have more than one hundred properties to set up and it is not possible for OpenESB to provide an exhaustive list of the properties to set up for each database on the market.

In order to allow users to benefit from the complete set of properties available for a datasource, OpenESB uses introspection and set up datasource properties with the values defined in context.xml.

OpenESB setups property values with simple types such as int, boolean or String and not with complex object.

Read your database documentation to list the mandatory properties to set up and find out the optimised settings that could improve your connection.

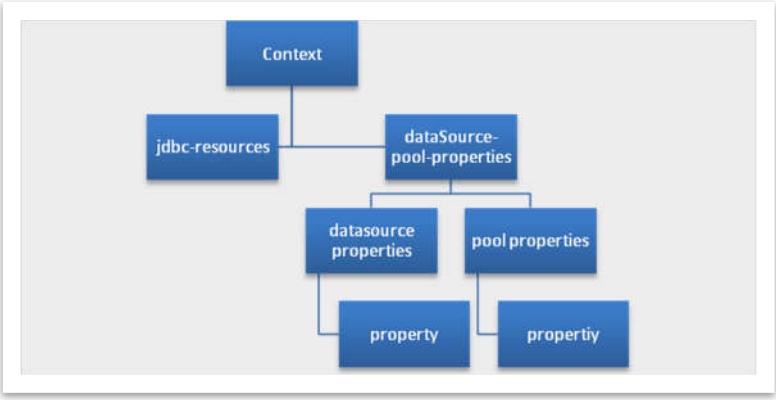
3.3 Create a Datasource-pool-properties in context.xml

Datasource and pool properties are defined in context.xml. Context.xml must conform with context.xsd.

Context.xml defines the following element:

XML element	Comment

Context	Root of the document
dataSource-pool-properties	Datasource and pool father
dataSource-properties	Where datasource properties are defined and set up
pool-properties	Where pool properties are defined and set up
jdbc-resources	Where Datasource JNDI Names are defined



3.4 Set up native datasource properties

3.4.1 Datasource-pool-properties element

“Datasource-pool-properties” is the root element where datasource and pool properties are defined. Datasource-pool-properties elements are listed in the table below. Elements in orange rows are mandatory.

Element name	Description	Comment
dbConnector-name	Name given to the dbConnector. This element identifies a connection pool (Datasource-pool-properties)	MYSQL Connector
datasource-classname	Class name of the native datasource	com.mysql.jdbc.jdbc2.optional.MysqlDataSource
resource-type	Type of the class: Datasource or XADatasource	XADatasource support XA transaction
database-name	Database name (Required for information only)	MySQL
database-vendor	Database vendor (Required for information only)	Oracle
database-version	Database version (Required for information only)	10.3
dbconnector-description	Database description (Required for information only)	DBConnector for MySQL

3.4.2 DataSource properties element

"Datasource-properties" element is a set of properties. Each property contains 3 elements: name, value and description.

Elements in orange rows are mandatory.

Element name	Description	Comment
name	Datasource property name	Must be exactly the same name than the name of the field you want to set up in the java class. If the name is incorrect or does not exist, and exception will be logged, but it does not prevent OpenESB from starting up. Example : user
value	Property values	Ex: root
description	Property description (Required for information only)	Example: user name

4 Set up pool properties

4.1 Pool properties element

Pool properties element is a set of properties used to configure the connection pool. The pool used in OpenESB SE is an Apache library that implements a JDBC Connection pool (http://commons.apache.org/proper/commons-dbcp/configuration.html).

Pool configuration could be found here: http://commons.apache.org/proper/commons-dbcp/configuration.html.

You certainly notice that JDBC Connection pool contains both JDBC properties and Pools properties. In order to provide more flexibility to OpenESB DB connection configurations, we decided to set up JDBC properties at the native driver level. Consequently: **It is useless toset any information related to JDBC mechanism at the JDBC Connection pool level**.

Pool properties are used to set up pool properties only.

The table below lists pools properties. Elements in orange rows are mandatory.

Parameter	Default	Description
initialSize	0	The initial number of connections that are created when the pool is started. Since: 1.2
maxTotal	8	The maximum of active connections that can be allocated from this pool at the same time, or negative for no limit.
maxIdle	8	The maximum of connections that can remain idle in the pool, without extra ones

		being released or negative ones for no limit.
validationQuery		The SQL query that will be used to validate connections from this pool before returning them to the caller. If specified, this query **MUST** be an SQL SELECT statement that returns at least one row. If not specified, connections will be validation by calling the isValid() method.
testOnCreate	false	The indication of whether objects will be validated after creation. If the object fails to validate, the borrow attempt that triggered the object creation will fail.
testOnBorrow	true	The indication of whether objects will be validated before being borrowed from the pool. If the object fails to validate, it will be dropped from the pool, and we will attempt to borrow another.
testOnReturn	false	The indication of whether objects will be validated before being returned to the pool.
testWhileIdle	false	The indication of whether objects will be validated by the idle object evictor (if any). If an object fails to validate, it will be dropped from the pool.
timeBetweenEvictionRunsMillis	-1	The number of milliseconds to sleep between runs of the idle object evictor thread. When non-positive, no idle object evictor thread will be run.
numTestsPerEvictionRun	3	The number of objects to examine during

		each run of the idle object evictor thread (if any).
minEvictableIdleTimeMillis	1000 * 60 * 30	The minimum amount of time an object may sit idle in the pool before it is eligible for eviction by the idle object evictor (if any).
softMiniEvictableIdleTimeMillis	-1	The minimum amount of time a connection may sit idle in the pool before it is eligible for eviction by the idle connection evictor, with the extra condition that at least "minIdle" connections remain in the pool. When miniEvictableIdleTimeMillis is set to a positive value, miniEvictableIdleTimeMillis is examined first by the idle connection evictor - i.e. when idle connections are visited by the evictor, idle time is first compared against miniEvictableIdleTimeMillis (without considering the number of idle connections in the pool) and then against softMinEvictableIdleTimeMillis, including the minIdle constraint.
maxConnLifetimeMillis	-1	The maximum lifetime in milliseconds of a connection. After this time is exceeded the connection will fail the next activation, passivation or validation test. A value of zero or less means the connection has an infinite lifetime.
connectionInitSqls	null	A Collection of SQL statements that will be used to initialize physical connections when

		they are first created. These statements are executed only once - when the configured connection factory creates the connection.
lifo	true	True means that borrowObject returns the most recently used ("last in") connection in the pool (if there are idle connections available). False means that the pool behaves as a FIFO queue - connections are taken from the idle instance pool in the order that they are returned to the pool.

More detail and explanation on Apache web site. Pool parameters can be setup dynamically by using JMX command (Ex: JConsole).

5 Create a JDBC-resource

5.1 JDBC resources element

JDBC Resources element creates a link between a JNDI name and Datasource-pool-properties. Elements in orange rows are mandatory.

WARNING: JNDI names with "/" are not supported. Don't use name such as jdbc/mydb but jdbc_mydb.

Element name	Description	Comment
dbConnector-name	Connection pool ID (Datasource-pool-properties ID) define in the previous step	Example : MYSQL Connector
JNDI Name	JNDI name that will be used by OpenESB components to access to a database. We don't use the notation proposed by JEE specification java/MyJNDIName. This can be modified in the next version.	Ex: MySQL01. If you want to migrate an application from OpenESB for Glassfish to OpenESB SE, JNDI name for databases must be reviewed
description	Description (Required for information only)	Example: Datasource connection to MySQ

Note: Many JNDI names can be linked to the same DBConnector.

6 Notice

OpenESB Naming features are set up when OpenESB starts. Context.XML is read during OE SE start-up. So OpenESB SE must be reset to take into account new datasources. However, pool configurations can be modified by JMX.

7 Help and support

7.1 From the community

You can find all our OpenESB documentations on the OpenESB official web site: www.open-esb.net.

If you have any questions or would like to share your feedback, use the OpenESB forum at: http://**openesb**-community-forum.794670.n2.**nabble**.com

Feel free to notify us with a bug or suggest how to improve our services on ： https://openesb.atlassian.net/secure/Dashboard.jspa

7.2 From Pymma

Pymma is deeply involved in the community and offers services and consulting on OpenESB. Pymma has professional services that can assist you from the development of your SOA design, implementation and ongoing management. All of our skills and background are based on our extensive first-hand experience and industry-leading methods.

Pymma releases an OpenESB Enterprise Edition with many additional enterprise features and a professional support.

In addition to OpenESB development, Pymma designed a new Service-Oriented development process named Rebecca to help business, architect and development team during the design and the implementation of their service oriented projects with OpenESB or any other service oriented development tool.

Feel free to contact us by email at contact@pymma.com for any further information on our OpenESB Services.

Appendix A. OpenESB context.xml example

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
   Document   : OEContext.xml
   Created on : 25 December 2014, 10:05
   Author     : polperez
   Description: OpenESB SE context.xml example
-->

<ns0:context  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
          xmlns:ns0='http://www.open-esb.net/standalone/jndi/'
          xsi:schemaLocation='http://www.open-esb.net/standalone/jndi/OEContext.xsd'>
   <ns0:dataSource-pool-properties>
      <ns0:dbConnector-name>MYSQL Connector</ns0:dbConnector-name>
      <ns0:datasource-classname>com.mysql.jdbc.jdbc2.optional.MysqlDataSource</ns0:datasource-classname>
      <ns0:resource-type>Datasource</ns0:resource-type>
      <ns0:database-name>MYSQL</ns0:database-name>
      <ns0:database-vendor>Oracle</ns0:database-vendor>
      <ns0:database-version>5.6</ns0:database-version>
      <ns0:dbconnector-description>DBConnector for MySQL</ns0:dbconnector-description>
      <ns0:dataSource-properties>
         <ns0:property>
            <ns0:name>user</ns0:name>
            <ns0:value>root</ns0:value>
            <ns0:description></ns0:description>
         </ns0:property>
         <ns0:property>
            <ns0:name>password</ns0:name>
            <ns0:value>password</ns0:value>
            <ns0:description></ns0:description>
         </ns0:property>
         <ns0:property>
            <ns0:name>hostName</ns0:name>
            <ns0:value>localhost</ns0:value>
            <ns0:description></ns0:description>
```

```
      </ns0:property>
      <ns0:property>
        <ns0:name>port</ns0:name>
        <ns0:value>3306</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>databaseName</ns0:name>
        <ns0:value>test</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
    </ns0:dataSource-properties>
    <ns0:pool-properties>
      <ns0:property>
        <ns0:name>initialSize</ns0:name>
        <ns0:value>11</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>maxActive</ns0:name>
        <ns0:value>20</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>maxIdle</ns0:name>
        <ns0:value>11</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>minIdle</ns0:name>
        <ns0:value>10</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
    </ns0:pool-properties>
  </ns0:dataSource-pool-properties>

  <ns0:dataSource-pool-properties>
    <ns0:dbConnector-name>Derby Connector</ns0:dbConnector-name>
    <ns0:datasource-classname>org.apache.derby.jdbc.ClientDataSource40</ns0:datasource-classname>
```

```
<ns0:resource-type>Datasource</ns0:resource-type>
<ns0:database-name>DERBY</ns0:database-name>
<ns0:database-vendor>OpenESB Community</ns0:database-vendor>
<ns0:database-version>10.7.1.1</ns0:database-version>
<ns0:dbconnector-description>DBConnector for Derby</ns0:dbconnector-description>
<ns0:dataSource-properties>
  <ns0:property>
    <ns0:name>user</ns0:name>
    <ns0:value>app</ns0:value>
    <ns0:description></ns0:description>
  </ns0:property>
  <ns0:property>
    <ns0:name>password</ns0:name>
    <ns0:value>app</ns0:value>
    <ns0:description></ns0:description>
  </ns0:property>
  <ns0:property>
    <ns0:name>serverName</ns0:name>
    <ns0:value>localhost</ns0:value>
    <ns0:description></ns0:description>
  </ns0:property>
  <ns0:property>
    <ns0:name>portNumber</ns0:name>
    <ns0:value>1527</ns0:value>
    <ns0:description></ns0:description>
  </ns0:property>
  <ns0:property>
    <ns0:name>databaseName</ns0:name>
    <ns0:value>sample</ns0:value>
    <ns0:description></ns0:description>
  </ns0:property>
</ns0:dataSource-properties>
<ns0:pool-properties>
  <ns0:property>
    <ns0:name>initialSize</ns0:name>
    <ns0:value>11</ns0:value>
    <ns0:description></ns0:description>
  </ns0:property>
  <ns0:property>
```

```
            <ns0:name>maxActive</ns0:name>
            <ns0:value>20</ns0:value>
            <ns0:description></ns0:description>
          </ns0:property>
          <ns0:property>
            <ns0:name>maxIdle</ns0:name>
            <ns0:value>10</ns0:value>
            <ns0:description></ns0:description>
          </ns0:property>
          <ns0:property>
            <ns0:name>minIdle</ns0:name>
            <ns0:value>10</ns0:value>
            <ns0:description></ns0:description>
          </ns0:property>
        </ns0:pool-properties>
      </ns0:dataSource-pool-properties>

      <ns0:dataSource-pool-properties>
        <ns0:dbConnector-name>Postgres Connector</ns0:dbConnector-name>
        <ns0:datasource-classname>org.postgresql.ds.PGSimpleDataSource</ns0:datasource-classname>
        <ns0:resource-type>Datasource</ns0:resource-type>
        <ns0:database-name>POSTGRESQL</ns0:database-name>
        <ns0:database-vendor>Postgres</ns0:database-vendor>
        <ns0:database-version>9.1-90</ns0:database-version>
        <ns0:dbconnector-description>DBConnector for postgres</ns0:dbconnector-description>
        <ns0:dataSource-properties>
          <ns0:property>
            <ns0:name>user</ns0:name>
            <ns0:value>postgres</ns0:value>
            <ns0:description></ns0:description>
          </ns0:property>
          <ns0:property>
            <ns0:name>password</ns0:name>
            <ns0:value>password</ns0:value>
            <ns0:description></ns0:description>
          </ns0:property>
          <ns0:property>
            <ns0:name>serverName</ns0:name>
            <ns0:value>localhost</ns0:value>
```

```
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>portNumber</ns0:name>
        <ns0:value>5432</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>databaseName</ns0:name>
        <ns0:value>postgres</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
    </ns0:dataSource-properties>
    <ns0:pool-properties>
      <ns0:property>
        <ns0:name>initialSize</ns0:name>
        <ns0:value>11</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>maxActive</ns0:name>
        <ns0:value>20</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>maxIdle</ns0:name>
        <ns0:value>10</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
      <ns0:property>
        <ns0:name>minIdle</ns0:name>
        <ns0:value>10</ns0:value>
        <ns0:description></ns0:description>
      </ns0:property>
    </ns0:pool-properties>
  </ns0:dataSource-pool-properties>

  <ns0:jdbc-resources>
    <ns0:dbConnector-name>MYSQL Connector</ns0:dbConnector-name>
```

```
      <ns0:jndi-name>MySQLServer01</ns0:jndi-name>
      <ns0:description>Datasource connection to MySQ</ns0:description>
   </ns0:jdbc-resources>

   <ns0:jdbc-resources>
      <ns0:dbConnector-name>Derby Connector</ns0:dbConnector-name>
      <ns0:jndi-name>Derby01</ns0:jndi-name>
      <ns0:description>Datasource connection to Derby</ns0:description>
   </ns0:jdbc-resources>

   <ns0:jdbc-resources>
      <ns0:dbConnector-name>Postgres Connector</ns0:dbConnector-name>
      <ns0:jndi-name>Postgres01</ns0:jndi-name>
      <ns0:description>Datasource connection to Postgres</ns0:description>
   </ns0:jdbc-resources>
</ns0:context>
```

Appendix B. Microsoft SQL Server Configuration

MS SQL setting does not follow the regular configuration described in the previous appendix but requires a special configuration. Connection parameters such as server, database, password must be embedded in a property named **dataSourceURL**.

Follow the example below to connect OpenESB to Microsoft SQL Server or Sybase SQL Server

```
<?xml version="1.0" encoding="UTF-8"?>
<context xmlns="http://www.open-esb.net/standalone/jndi/">
    <dataSource-pool-properties>
        <dbConnector-name>MSSQLPoll</dbConnector-name>
        <datasource-classname>com.microsoft.sqlserver.jdbc.SQLServerDataSource</datasource-classname>
        <resource-type>Datasource</resource-type>
        <database-vendor>Microsoft</database-vendor>
        <database-version>12.0.2</database-version>
        <dbconnector-description>DBConnector for mssql</dbconnector-description>
        <dataSource-properties>
            <property>
                <name>dataSourceURL</name>
                <value>jdbc:sqlserver://myserver:myPort;databaseName=myDatabase;user=User;
```

```
        password=myPassword </value>
        <description>URL value  for MSSLQ Connection </description>
    </property>
</dataSource-properties>
<pool-properties>
    <property>
        <name>initialSize</name>
        <value>11</value>
        <description></description>
    </property>
    <property>
        <name>maxActive</name>
        <value>20</value>
        <description></description>
    </property>
    <property>
        <name>maxIdle</name>
        <value>10</value>
        <description></description>
    </property>
    <property>
```

```
            <name>minIdle</name>
            <value>10</value>
            <description></description>
        </property>
    </pool-properties>
</dataSource-pool-properties>

<jdbc-resources>
    <dbConnector-name> MSSQLPoll </dbConnector-name>
    <jndi-name>jndiMSSQL</jndi-name>
    <description>Datasource connection to MSSQL</description>
</jdbc-resources>
```