

# **JasperServer Open Source Source Build Guide Version 2.1**

**<http://www.jaspersoft.com/>**

© 2007 JasperSoft Corporation. All rights reserved. Printed in the U.S.A. JasperSoft, the JasperSoft logo, JasperAnalysis, JasperServer, JasperETL, JasperReports, JasperStudio, iReport, and Jasper4 products are trademarks and/or registered trademarks of JasperSoft Corporation in the United States and in jurisdictions throughout the world. All other company and product names are or may be trade names or trademarks of their respective owners.

# Table of Contents

<b>1</b>	<b>Introduction.....</b>	<b>4</b>
1.1	Get the JasperServer Source Code.....	4
1.2	Install Java JDK.....	4
1.3	Install Maven .....	5
<b>2</b>	<b>Begin Setup of Build Environment.....</b>	<b>5</b>
2.1	Download and Unpack the Source Code.....	5
2.2	Configure Maven and Other Build Files .....	6
2.2.1	Create the settings.xml .....	6
2.2.2	Create hibernate.cfg.xml File .....	8
2.2.3	Create the js.jdbc.properties File .....	9
2.2.4	Create the js.mail.properties File .....	9
<b>3</b>	<b>Install MySQL and Create JasperServer Databases.....</b>	<b>9</b>
3.1.1	Create MySQL Databases .....	10
<b>4</b>	<b>Build JasperServer Source Code .....</b>	<b>10</b>
4.1	Build JasperServer Source Code.....	10
<b>5</b>	<b>Prepare to Run JasperServer.....</b>	<b>11</b>
5.1	Install Apache Tomcat 5.5.....	11
5.2	Copy MySQL Driver Jar .....	11
5.3	Validate Tomcat Related Config Files.....	11
<b>6</b>	<b>Run JasperServer.....</b>	<b>12</b>
6.1	Copy the jasperserver War directory .....	12
6.2	Java Options for JasperServer.....	12
6.3	Start Tomcat.....	13
6.4	Login to JasperServer .....	13
6.5	JasperServer Logging Location .....	13
<b>7</b>	<b>Troubleshooting.....</b>	<b>13</b>
7.1	Database Troubleshooting .....	13
7.2	Maven Troubleshooting.....	13
7.2.1	Maven Error Due to Maven Version .....	13
7.2.2	Maven Warnings .....	13
7.2.3	Maven Error: Transferring Jar File.....	14
7.2.4	Maven Build Error Due to Bad Path to jasperserver-repo in Settings.xml.....	14
7.2.5	Maven Binary Versions .....	15
7.3	Ant Binary Not Found.....	15
<b>Appendix A</b>	<b>Tomcat 5.0 Configuration.....</b>	<b>15</b>

**Appendix B    Building JPivot Source Code.....17**

    B.1    JasperJPivot source code build and deployment ..... 17

    B.2    Prepare build environment ..... 17

    B.3    Unpack JasperJPivot source ..... 17

    B.4    Build JasperJPivot component..... 18

    B.5    Build outputs..... 18

    B.6    Sample build in Windows ..... 19

    B.7    Troubleshooting.....20

# 1 Introduction

This guide is written for developers to assist in obtaining, setting up, building and running JasperServer from the source files.

The JasperServer development environment has the concept of a "default" or "standard" build that relies on a number of tool components (3<sup>rd</sup> party jars) and assumes the existence of the Apache Tomcat web container and the MySQL database.

For a complete build and start up of JasperServer there are the following main steps:

- Get the JasperServer Source Code
- Install Java
- Install Maven
- Unpack Source Packages
- Configure the Development Environment
- Install MySQL
- Build the JasperServer Source Code
- Install Apache Tomcat
- Run JasperServer

**Note:** In regards to the build environment, this document describes how to build from a command line shell under Unix/Linux or Windows XP. Setting up for a build within an IDE (such as Eclipse) is outside the scope of this document.

## 1.1 Get the JasperServer Source Code

### Download ZIP File

The JasperServer source code is available as a ZIP file download from JasperForge.org. You will need to register on the JasperForge.org site in order to gain access to the downloads area.

<http://www.jasperforge.org>

Look for the file:

`jasperserver-<ver>-src.zip`

### Subversion Checkout

The JasperServer source code will soon be made available for checkout from a Subversion repository on JasperForge.org. This repository will hold the most current trunk source code. Look on the JasperForge.org forums page for updated information on this.

### JPivot Source Code

JasperSoft has made changes to the JPivot code that is used in JasperServer. You do not need to compile this source because the classes are included in the `jpivot-<ver>.jar` found in the `jasperserver-repo`. However, if you would like to build this code you can download the source as a separate ZIP on JasperForge.org. This document has an Appendix B which covers the building of this source.

## 1.2 Install Java JDK

The JasperServer source code should be compiled under Java 1.5 or later. While JasperServer does not use any of the Java 1.5 language features, there are Java 1.5 dependencies with some of the 3rd party jars.

To install Java, follow the download and installation instructions found at Sun's Java web site:

<http://java.sun.com>

Java 1.5 is also known as the Java SE Development Kit (JDK) and it includes:

- The Java Runtime Environment (JRE)
- Command-line development tools, such as compilers and debuggers, that are necessary for developing Java applications.

## 1.3 Install Maven

The JasperServer development team has found Maven2 to be an effective tool for development, particularly with its ability to manage 3rd party tool (jar) dependencies via remote, online repositories.

Maven is used to compile, build, and package the JasperServer source code.

The driver of the Maven build system is the XML based pom.xml file. Each directory with a pom.xml creates a single "artifact", which can be a jar, war, a project web site, etc. Pom.xml files refer to dependencies on other artifacts and these dependencies allow for class and other resource resolution in the current directory and are used to determine what resources need to be included in final artifacts.

Information on Maven2 can be found on the Maven site.

<http://maven.apache.org/>

The Maven2 download and install can be initiated from the Maven site:

<http://maven.apache.org/download.html#installation>

The mvn binary should be put on your machine's PATH so that you can execute mvn from the command line. To see that this is working try the following::

```
mvn -version
```

**Note:** There have been problems with newer versions of maven. This build process has been tested with maven 2.0.4.

## 2 Begin Setup of Build Environment

We will begin the setup of the build environment by unpacking the source package. Then we will start the configuration file setup.

### 2.1 Download and Unpack the Source Code

If you have obtained the source code from a download ZIP, you should now unpack the ZIP file.

```
jasperserver-<ver>-src.zip
```

**Create the development directory:**

Create a directory that you will use for building the source code. Examples would be:

```
mkdir C:\js-builds
```

```
mkdir /home/<username>/js-builds
```

We will call this src directory <js-builds>.

Unzip the jasperserver-<ver>-src.zip to the development location:

```
Unzip jasperserver-<ver>-src.zip to <js-builds>
```

After the "unzip" operation, you should have the following structure in your development directory:

```
<js-builds>/
```

```
    /jasperserver
```

```
    /jasperserver-repo
```

jasperserver – JasperServer open source code

jasperserver-repo – jar files used in the build that are not (or were not) easily available publicly

## 2.2 Configure Maven and Other Build Files

We will setup the following build configuration files:

<b>settings.xml</b>	- maven settings file
<b>hibernate.cfg.xml</b>	- specifies hibernate settings
<b>js.jdbc.properties</b>	- specifies database settings
<b>js.mail.properties</b>	- specifies mail gateway settings

Samples of these files are included in the source code package under the following directory:

jasperserver/scripts/dev-setup

You can take the example files from the **dev-setup** dir, or you can cut and paste from the examples below.

**Note:** If you cut and paste directly from this document, please watch out for potentially bad characters in the target pasted files.

### 2.2.1 Create the settings.xml

Settings.xml is the main configuration and properties setting file that is used by the Maven build tool.

In order to configure Maven, you will need to create your own version of the settings.xml file. Maven expects this configuration file to be in a directory named ".m2" within your "home" directory. To create this directory, do the following:

Under Windows:

```
cd \Document and Settings\<username>
mkdir .m2                (please note the "dot" in ".m2" -i.e. a "hidden" directory)
```

Under Linux:

```
cd $HOME
mkdir .m2                (please note the "dot" in ".m2" - i.e. a "hidden" directory)
```

The settings.xml file will go into the root of your .m2 directory:

```
C:\Documents and Settings\<username>\.m2\settings.xml
/home/<username>/.m2/settings.xml
```

Maven uses this settings.xml file for all of the configuration options which affect the build.

#### Sample Settings.xml File

**Note:** Modify items in **bold** to match your own environment.

**Note:** It is better to take the settings.xml file from the scripts/dev-setup directory so that you do not get bad character values from a cut and paste operation of this document.

```
<settings>
  <!-- <offline>true</offline> -->
  <profiles>
    <profile>
      <id>JasperServerProfile</id>
      <properties>
        <test.hibernate.cfg>/home/devuser/.m2/hibernate.cfg.xml</test.hibernate.cfg>
        <test.hibernate.jdbc.properties>/home/devuser/.m2/js.jdbc.properties</test.hibernate.jdbc.properties>
        <repository.database.driver.groupId>mysql</repository.database.driver.groupId>
        <repository.database.driver.artifactId>mysql-connector-java</repository.database.driver.artifactId>
        <repository.database.driver.version>3.1.11</repository.database.driver.version>
        <js.mail.properties>/home/devuser/.m2/js.mail.properties</js.mail.properties>
        <js.quartz.script>/home/devuser/js-builds/jasperserver/scripts/quartz/tables_mysql_innodb.sql</js.quartz.script>
        <metadata.database.generate>true</metadata.database.generate>
      </properties>
    </profile>
  </profiles>
</settings>
```

```
<repositories>
  <repository>
    <id>ji-repo-svn</id>
    <name>Internal dependencies from SVN</name>
    <url>file:/home/devuser/js-builds/jasperserver-repo</url>
    <snapshots>
      <enabled>true</enabled>
      <updatePolicy>always</updatePolicy>
    </snapshots>
  </repository>
  <repository>
    <id>JasperForge Maven Repository</id>
    <url>http://www.jasperforge.org/maven2</url>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
    <releases>
      <enabled>false</enabled>
    </releases>
  </repository>
  <repository>
    <id>Repo1 Maven</id>
    <url>http://repo1.maven.org/maven2</url>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
    <releases>
      <enabled>false</enabled>
    </releases>
  </repository>
  <repository>
    <id>Maven Snapshots</id>
    <url>http://snapshots.maven.codehaus.org/maven2</url>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
    <releases>
      <enabled>false</enabled>
    </releases>
  </repository>
</repositories>
<pluginRepositories>
  <pluginRepository>
    <id>Maven Snapshots</id>
    <url>http://snapshots.maven.codehaus.org</url>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
    <releases>
      <enabled>false</enabled>
    </releases>
  </pluginRepository>
  <pluginRepository>
    <id>codehaus-plugins</id>
    <url>http://dist.codehaus.org</url>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
    <releases>
      <enabled>true</enabled>
    </releases>
  </pluginRepository>
</pluginRepositories>
</profile>
```

```

</profiles>
<activeProfiles>
  <activeProfile>JasperServerProfile</activeProfile>
</activeProfiles>
<servers>
  <server>
    <id>myserver</id>
    <username>tomcat</username>
    <password>tomcat</password>
  </server>
</servers>
</settings>

```

### Double-check values in settings.xml File:

**Note:** Modify items in **bold** to match your own environment.

```

<test.hibernate.cfg>/home/devuser/.m2/hibernate.cfg.xml</test.hibernate.cfg>           (Linux style)
<test.hibernate.jdbc.properties>/home/devuser/.m2/js.jdbc.properties</test.hibernate.jdbc.properties>
<js.mail.properties>/home/devuser/.m2/js.mail.properties</js.mail.properties>
<js.quartz.script>/home/devuser/js-builds/jasperserver/scripts/quartz/tables_mysql_innodb.sql</js.quartz.script>

<test.hibernate.cfg>C:/Documents and Settings/devuser/.m2/hibernate.cfg.xml</test.hibernate.cfg>   (Windows style)
<test.hibernate.jdbc.properties>C:/Documents and Settings/devuser/.m2/js.jdbc.properties</test.hibernate.jdbc.properties>
<js.mail.properties>C:/Documents and Settings/devuser/.m2/js.mail.properties</js.mail.properties>
<js.quartz.script>C:/Documents and Settings/devuser/js-builds/jasperserver/scripts/quartz/tables_mysql_innodb.sql</js.quartz.script>

<repository>
  <id>jasperServer</id>
  <name>Base repository for Jasper Server</name>
  <url>file:/home/devuser/js-builds/jasperserver-repo</url>           (Linux style)
</repository>

<repository>
  <id>jasperServer</id>
  <name>Base repository for Jasper Server</name>
  <url>file://C:/js-builds/jasperserver-repo</url>           (Windows style)
</repository>

```

### 2.2.2 Create hibernate.cfg.xml File

The hibernate.cfg.xml file would typically go into your maven <home-dir>/.m2 directory.

#### Sample hibernate.cfg.xml File

**Note:** Modify items in **bold** to match your own environment.

```

<!DOCTYPE hibernate-configuration PUBLIC
  "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
  "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
  <session-factory name="metadata">

    <!-- Database connection settings -->
    <property name="connection.driver_class">com.mysql.jdbc.Driver</property>
    <property name="connection.url">jdbc:mysql://localhost:3306/jasperserver?useUnicode=true&characterEncoding=UTF-8</property>
    <property name="connection.username">root</property>
    <property name="connection.password">password</property>

    <property name="dialect">org.hibernate.dialect.MySQLInnoDBDialect</property>

    <property name="show_sql">>false</property>
    <property name="format_sql">>false</property>

  </session-factory>
</hibernate-configuration>

```



### 2.2.3 Create the js.jdbc.properties File

The js.jdbc.properties file would typically go into your maven <home-dir>/m2 directory.

#### Sample js.jdbc.properties File

**Note:** Modify items in **bold** to match your own environment.

```
# Property that determines the Hibernate dialect
# (only applied with "applicationContext-hibernate.xml")
metadata.hibernate.dialect=org.hibernate.dialect.MySQLDialect

metadata.jdbc.driverClassName=com.mysql.jdbc.Driver
metadata.jdbc.url=jdbc:mysql://localhost:3306/jasperserver?useUnicode=true&characterEncoding=UTF-8
metadata.jdbc.username=root
metadata.jdbc.password=password

metadata.jndi=jdbc/jasperserver

test.jdbc.driverClassName=com.mysql.jdbc.Driver
test.jdbc.url=jdbc:mysql://localhost:3306/sugarcrm
test.jdbc.username=root
test.jdbc.password=password

test.jndi=jdbc/sugarcrm

foodmart.jdbc.driverClassName=com.mysql.jdbc.Driver
foodmart.jdbc.url=jdbc:mysql://localhost:3306/foodmart
foodmart.jdbc.username=root
foodmart.jdbc.password=password

foodmart.jndi=jdbc/foodmart
```

### 2.2.4 Create the js.mail.properties File

The js.mail.properties file would typically go into your maven <home-dir>/m2 directory. If you do not plan to connect to a mail gateway, you can put dummy values into the js.mail.properties file.

#### Sample js.mail.properties File

**Note:** Modify items in **bold** to match your own environment.

```
js.report.scheduler.mail.sender.host=mail.my-company.com
js.report.scheduler.mail.sender.port=25
js.report.scheduler.mail.sender.protocol=smtp
js.report.scheduler.mail.sender.username=admin
js.report.scheduler.mail.sender.password=password
js.report.scheduler.mail.sender.from=admin@my-company.com
```

## 3 Install MySQL and Create JasperServer Databases

In order to run the unit tests and the database creation and load scripts MySQL needs to be available so we will install and configure MySQL in this section.

The JasperServer development team has used MySQL as the default database for development and testing. This means that the default build configurations are set to use MySQL.

To install MySQL, follow the links for downloading and installing it at the MySQL web site:

<http://www.mysql.com>

### 3.1.1 Create MySQL Databases

The default database configuration information used by the JasperServer build is the following:

Parameter	Default Value
Database Hostname	localhost
Database Port	3306
Database Username	root
Database Password	password

If your database is installed and configured using the values above then changes to the JasperServer build configuration files will be minimal.

If you prefer to not use the root MySQL account for development then it is typical for JasperServer to use a database user with the name of **jasperadmin**.

The following example command can create a new DB user:

```
mysql -u root -p                                (log into MySQL)
mysql> grant all on *.* to jasperadmin@localhost identified by 'password';
```

Whatever your database settings are, they should be used in the build configuration files mentioned in the previous section (2.2 “Configure Maven and Other Build Files”)

#### Create the Databases

The JasperServer unit tests assume the existence of the JasperServer sample data so we will create and populate these databases as well.

Note: Unzip the database files in the jasperserver/scripts/mysql directory.

Log into MySQL and create databases (including sample databases):

```
cd <js-builds>/jasperserver/scripts/mysql
mysql -u root -p -h localhost                    (log into MySQL)
mysql> create database jasperserver character set utf8;
mysql> create database sugarcrm;
mysql> create database foodmart;
mysql> use sugarcrm;
mysql> source sugarcrm-mysql.sql                  (sugarcrm sample database)
mysql> use foodmart;
mysql> source foodmart-mysql.sql                  (foodmart sample database)
```

## 4 Build JasperServer Source Code

We are now ready to build the JasperServer source code.

### 4.1 Build JasperServer Source Code

To build JasperServer do the following:

```
cd <js-builds>/jasperserver
```

```
mvn clean install
cd jasperserver-repository-hibernate/build_db           (create and populate db schema)
mvn clean install
cd ../jasperserver-unit-tests                          (run the unit tests)
mvn clean install
```

If you encounter errors, please refer to the troubleshooting section to help debug any errors.

The **mvn clean install** that was executed in the jasperserver directory is the one that builds the jasperserver.war file. If everything compiled cleanly you will find this war file in the jasperserver/jasperserver-war/target directory.

## 5 Prepare to Run JasperServer

Before running JasperServer, we will have to have the Tomcat application server available and configured.

### 5.1 Install Apache Tomcat 5.5

The JasperServer development team has used Tomcat 5.5 as the default application server/servlet container. JasperServer will run under Tomcat 5.5 as well as 5.0. These steps are assuming Tomcat 5.5.

To install Tomcat, follow the download and install instructions from the Apache Tomcat web site:

<http://apache.tomcat.org>

(Note: Apache Tomcat can be installed as a service or into a developer specified directory.)

### 5.2 Copy MySQL Driver Jar

You will need to get a MySQL driver in order to allow JasperServer (within Tomcat) to connect to the MySQL database.

You can find a MySQL driver in the jasperserver/scripts/mysql directory.

Copy the following driver jar:

```
<js-builds>/jasperserver/scripts/mysql/mysql-connector-java-3.1.11-bin.jar
```

to:

```
<tomcat-install-dir>/common/lib           (Tomcat 5.5)
```

```
<tomcat-install-dir>/lib                  (Tomcat 6.x)
```

### 5.3 Validate Tomcat Related Config Files

#### Validate Context.xml

The JasperServer build process creates a “context.xml” file that is used by Tomcat to connect to the MySQL database.

It is a good idea to verify that the context.xml was created with the settings that you expect (that match the database settings that you put into your \$HOME/.m2/js.jdbc.properties configuration file).

To verify that your context.xml was setup properly, check the following file:

```
<js-builds>/jasperserver/jasperserver-war/target/jasperserver/META-INF/context.xml
```

The settings should look something like the following, but have the correct information for your database setup:

```
<Resource name="jdbc/jasperserver" auth="Container" type="javax.sql.DataSource"
    maxActive="100" maxIdle="30" maxWait="10000"
    username="root" password="password" driverClassName="com.mysql.jdbc.Driver">
```

```
url="jdbc:mysql://localhost:3306/jasperserver?useUnicode=true&characterEncoding=UTF-8"/>
```

```
<Resource name="jdbc/sugarcrm" auth="Container" type="javax.sql.DataSource"
    maxActive="100" maxIdle="30" maxWait="10000"
    username="root" password="password" driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost:3306/sugarcrm?useUnicode=true&characterEncoding=UTF-8"/>
```

```
<Resource name="${foodmart.jndi}" auth="Container" type="javax.sql.DataSource"
    maxActive="100" maxIdle="30" maxWait="10000"
    username="root" password="password" driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost:3306/foodmart"/>
```

### Other Database Related Files

If you have trouble running JasperServer (database failures) you can also check the following files:

```
<js-builds>/jasperserver/jasperserver-war/target/jasperserver /WEB-INF/web.xml
```

```
<js-builds>/jasperserver/jasperserver-war/target/jasperserver /WEB-INF/hibernate.properties
```

## 6 Run JasperServer

### 6.1 Copy the jasperserver War directory

A war file is a Web Archive file. It is meant to contain all components to execute within a web application container such as Apache Tomcat.

Now that Tomcat is configured, you should be able to move the jasperserver war directory over to Tomcat and then startup.

Copy the entire directory:

```
<js-builds>/jasperserver/jasperserver-war/target/jasperserver
```

To:

```
<apache-tomcat>/webapps
```

### 6.2 Java Options for JasperServer

JasperServer has analysis capability using the JPivot and Mondrian project jars. This OLAP analysis can handle very large datasets. The JasperServer sample data includes OLAP cubes for the standard Mondrian FoodMart as well as a special sample cube for SugarCRM. When running large datasets, you could encounter out of memory errors.

It is recommended that you set the following java options when running JasperServer.

#### Java Options

Add to your Tomcat JAVA\_OPTS settings:

```
-Xms512m -Xmx1024m
```

In addition, if you get problems with PermGen space, you can the following options:

```
-XX:PermSize=128m -XX:MaxPermSize=512m
```

These settings would be added to your tomcat setclasspath script:

```
<apache-tomcat>/bin/setclasspath.bat
```

```
<apache-tomcat>/bin/setclasspath.sh
```

## 6.3 Start Tomcat

<apache-tomcat>/bin/startup.bat - Windows

<apache-tomcat>/bin/startup.sh - Linux

Tomcat normally starts a console window where you can look for any startup errors. The most common errors are database related.

## 6.4 Login to JasperServer

You can now login to JasperServer via a web browser.

Use the following URL:

`http://localhost:8080/jasperserver`

You should be able to use the following login:

Username: `jasperadmin` (admin user)

Password: `<password>`

You should now be logged into the JasperServer application. You may now view reports, view OLAP analysis views, create Ad-Hoc report, edit resources, etc. You may refer to the JasperServer User Guide found in the docs directory for more information on using the application.

If you are unable to login or have other troubles, please refer to the troubleshooting section of this guide or to the JasperServer Install Guide which has additional troubleshooting.

## 6.5 JasperServer Logging Location

The JasperServer log gets written to the following location:

<apache-tomcat>/webapps/jasperserver/WEB-INF/logs/jasperserver.log

The log4j logging level can be controlled via the log4j.properties file in the following location:

<apache-tomcat>/webapps/jasperserver/WEB-INF/log4j.properties

# 7 Troubleshooting

## 7.1 Database Troubleshooting

The most common error encountered when building JasperServer involves the database. Common errors are not being able to connect to the database. The JasperServer Install Guide has the most detail on troubleshooting database problems.

The JasperServer Install Guide also contains directions for configuring JasperServer from the War File Binary Distribution ZIP file as well as troubleshooting the War File Distribution.

## 7.2 Maven Troubleshooting

### 7.2.1 Maven Error Due to Maven Version

There have been problems with newer versions of maven. The build process in this document has been tested with maven 2.0.4. Maven 2.0.7 has been tested and works for our developers. However, **maven 2.0.6 does not** work correctly with the JasperServer build.

### 7.2.2 Maven Warnings

Maven2 generates warnings during the artifact validation processing. It is typical to get warnings regarding non-standard layouts of artifacts (such as having a jar file without a corresponding pom file, or if a checksum file is not available). These warnings can be ignored

The example below has a warning but the jar file needed was downloaded successfully:

```
[WARNING] Unable to get resource from repository jasperServer (file://C:/svn/jasperintel/jasperserver
Downloading: http://repo1.maven.org/maven2/commons-logging/commons-logging/1.0/commons-logging-1.0.pom
163b downloaded
```

### 7.2.3 Maven Error: Transferring Jar File

It is not uncommon for one of the jars to have a transfer error when downloading, given that there are many downloads on the first build. In the example below, there was a transfer error on the castor.jar file.

This problem was fixed by re-running the "mvn install" command (ie. Restart the build).

```
[ERROR] BUILD ERROR
[INFO] -----
[INFO] Error building POM (may not be this project's POM).
Project ID: castor:castor
Reason: Error getting POM for 'castor:castor' from the repository: Error transferring file
castor:castor:pom:1.0
```

from the specified remote repositories:

```
Maven Snapshots (http://snapshots.maven.codehaus.org/maven2/),
central (http://repo1.maven.org/maven2/),
ApacheSVN-central (http://svn.apache.org/maven-snapshot-repository),
jasperServer (file://C:\workspaces\src\jasperserver-repo\jasperserver-maven)
```

### 7.2.4 Maven Build Error Due to Bad Path to jasperserver-repo in Settings.xml

If the jasperserver-repo directory is not correctly pointed to in the settings.xml file, many of the dependent jars will not be able to be found. The error would look similar to the error listing below. In this case, you will need to double-check the \$HOME/.m2/settings.xml file and validate that it is able to find the <js-builds>/jasperserver-repo directory.

```
[ERROR] BUILD ERROR
[INFO] -----
[INFO] Failed to resolve artifact.
Missing:
-----
1) javax.transaction:jta:jar:1.0.1B
```

Try downloading the file manually from:  
<http://java.sun.com/products/jta>

Then, install it using the command:  
mvn install:install-file -DgroupId=javax.transaction -DartifactId=jta \
-Dversion=1.0.1B -Dpackaging=jar -Dfile=/path/to/file

Path to dependency:

- 1) com.jaspersoft.jasperserver.api.metadata:jasperserver-api-metadata:jar:1.0.0
- 2) org.acegisecurity:acegi-security:jar:1.0.0
- 3) org.springframework:spring-jdbc:jar:2.0-m2
- 4) org.springframework:spring-dao:jar:2.0-m2
- 5) javax.transaction:jta:jar:1.0.1B

2) jasperreports:jasperreports:jar:1.2.3

Try downloading the file manually from the project website.  
mvn install:install-file -DgroupId=jasperreports -DartifactId=jasperreports \
-Dversion=1.2.3 -Dpackaging=jar -Dfile=/path/to/file

Path to dependency:

- 1) com.jaspersoft.jasperserver.api.metadata:jasperserver-api-metadata:jar:1.0.0

2) jasperreports:jasperreports:jar:1.2.3

-----  
2 required artifacts are missing.

for artifact:

com.jaspersoft.jasperserver.api.metadata:jasperserver-api-metadata:jar:1.0.0  
from the specified remote repositories:  
Maven Snapshots (<http://snapshots.maven.codehaus.org/maven2/>),  
central (<http://repo1.maven.org/maven2/>),  
ApacheSVN-central (<http://svn.apache.org/maven-snapshot-repository>),  
jasperServer (<file://C:\workspace\src\jasperserver-repo>)

### 7.2.5 Maven Binary Versions

The current recommended version is maven 2.0.7. In addition, maven 2.0.4 has worked fine.

There have been problems with some particular versions of the maven binary (mvn or mvn.exe). We have found that maven 2.0.6 **does not** properly work to build our source code.

## 7.3 Ant Binary Not Found

The apache ant build tool is also needed in addition to maven. A number of the maven pom.xml files make calls to ant functionality. Ant should be on the build path. To download ant go to the following location:

<http://ant.apache.org/>

## Appendix A Tomcat 5.0 Configuration

If you are running Tomcat 5.0, the context.xml has a different format.

Here is a sample that can be used:

```
<Context docBase="jasperserver" path="/jasperserver">
  <Resource name="jdbc/jasperserver" type="javax.sql.DataSource" auth="Container"
    debug="5" reloadable="true" crossContext="true"/>
  <ResourceParams name="jdbc/jasperserver">
    <parameter>
      <name>factory</name>
      <value>org.apache.commons.dbcp.BasicDataSourceFactory</value>
    </parameter>
    <parameter>
      <name>url</name>
      <value>jdbc:mysql://localhost:3306/jasperserver?autoReconnect=true</value>
    </parameter>
    <parameter>
      <name>driverClassName</name><value>com.mysql.jdbc.Driver</value>
    </parameter>
    <parameter>
      <name>username</name>
      <value>root</value>
    </parameter>
    <parameter>
      <name>password</name>
      <value>password</value>
    </parameter>
    <parameter>
      <name>maxIdle</name>
      <value>30</value>
    </parameter>
  </ResourceParams>
</Context>
```

```
<parameter>
  <name>maxActive</name>
  <value>100</value>
</parameter>
<parameter>
  <name>maxWait</name>
  <value>10000</value>
</parameter>
</ResourceParams>
</Context>
```



## Appendix B Building JPivot Source Code

### B.1 JasperJPivot source code build and deployment

This section describes the source code build and deployment steps of the JasperJPivot component for developers who want to make changes and extensions.

JasperJPivot is adapted from the JPivot open source project. It provides the Web GUI for the JasperAnalysis of the JasperServer.

The JasperJPivot component includes the following jars:

- jpivot-2.1.0.jar
- wcf-2.1.0.jar
- tbutils-wcf-2.1.0.jar

where 2.1.0 is the JasperJPivot version.

The JasperJPivot component is built using Ant with a number of 3<sup>rd</sup> party jars. The main build and deployment steps are:

- Prepare build environment
- Unpack JasperJPivot source
- Build JasperJPivot component
- Deploy JasperJPivot component

### B.2 Prepare build environment

The JasperJPivot source code can be built under either Java 1.4 or Java 1.5. Like JasperServer, the JasperJPivot source code standard is to not use any of the new language features found in Java 1.5 in order to support customers who run within a Java 1.4 environment.

If you have not installed java, follow the download and installation instructions found at Sun's Java web site:

<http://java.sun.com/>

While the JasperServer is build with Maven, the JasperJPivot is build with Ant, which is derived from the original JPivot open source project. Information on Ant can be found on the Ant site.

<http://ant.apache.org/>

Finally, make sure JasperServer source is downloaded and successfully built (See section 4 for details).

### B.3 Unpack JasperJPivot source

The JasperJPivot source is distributed in a zip file. For version 2.1.0, as an example, the zip file is called `jasperjpivot-2.1.0-src.zip`. As in the original JPivot project, It contains the following sub-directories:

- jpivot
- jpivot\_repository

- mondrian-schema
- wcf

Simply unpack the directory in a JasperServer project directory, such as c:\jaspersoft\jasperserver\jasperserver-jpivot.

## B.4 Build JasperJPivot component

The build process consists of building two sub-components: wcf and jpivot. First, configure the build environment by changing the following properties in build.properties:

- js.repo.home, e.g., c:\jaspersoft\jasperserver-repo
- js.repository.home, e.g., c:/Docume~1/jshih/.m2/repository

where js.repo.home defines the location of JasperServer local repository and js.repository.home defines the location of Maven repository.

Then, invoke the build with Ant using the build.xml from the command line with the wcf build first, for example:

```
cd c:\jaspersoft\jasperserver\jasperserver-jpivot\wcf
ant clean build
cd c:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot
ant clean build
```

where c:\jaspersoft\jasperserver\jasperserver-jpivot is the project directory of the JasperJPivot source code.

## B.5 Build outputs

Once the build is completed successfully, the output can be found in the c:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot\dist directory. The outputs consists of jar files and application server specific files.

### B.5.1 Jar files

The following jar files will be placed in the JasperServer local repository directory. For example, if the JasperServer repository is located at c:\jaspersoft\jasperserver-repo, the target directories are:

- c:\jaspersoft\jasperserver-repo\com\jaspersoft\jasperserver\jpivot-ui\jpivot\2.1.0\jpivot-2.1.0.jar
- c:\jaspersoft\jasperserver-repo\com\jaspersoft\jasperserver\jpivot-ui\wcf\2.1.0\wcf-2.1.0.jar
- c:\jaspersoft\jasperserver-repo\com\jaspersoft\jasperserver\jpivot-ui\tbutils-wcf\2.1.0\tbutils-wcf-2.1.0.jar

where 2.1.0 is the JasperJPivot version.

### B.5.2 Application Server files (e.g., Tomcat)

The following application server files are located in the JasperServer deployment directories, i.e., under jasperserver-war directory. For example, if the JasperServer source is located at c:\jaspersoft\jasperserver, the target directories are:

JasperJPivot files	JasperServer deployment directories
C:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot\dist\web\jpivot\*.*	c:\jaspersoft\jasperserver\jasperserver-war\src\main\webapp\jpivot
C:\jaspersoft\jasperserver\jasperserver-	c:\jaspersoft\jasperserver\jasperserver-

jpivot\jpivot\dist\web\wcf\*. *	war\src\main\webapp\wcf
C:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot\dist\web\WEB-INF\jpivot\*. *	c:\jaspersoft\jasperserver\jasperserver-war\src\main\webapp\WEB-INF\jpivot
C:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot\dist\web\WEB-INF\wcf\*. *	c:\jaspersoft\jasperserver\jasperserver-war\src\main\webapp\WEB-INF\wcf

## B.6 Sample build in Windows

- Download and install java to c:\Java\jdk1.5.0\_06
- 'set JAVA\_HOME=c:\Java\jdk1.5.0\_06'
- Download and install ant to c:\ant\apache-ant-1.6.5
- 'set ANT\_HOME=c:\ant\apache-ant-1.6.5'
- Add C:\ant\apache-ant-1.6.5\bin to PATH
- Download and unzip JasperServer source (e.g., jasperserver-2.1.0-src.zip) to c:\jaspersoft\jasperserver. The target directory should contain c:\jaspersoft\jasperserver and c:\jaspersoft\jasperserver-repo directories (See section 4 for details)
- 'md c:\jaspersoft\jasperserver\jasperserver-jpivot'
- Download and unzip JasperServer JPivot source (e.g., jasperjpivot-2.1.0-src.zip) to c:\jaspersoft\jasperserver\jasperserver-jpivot
- 'cd c:\jaspersoft\jasperserver\jasperserver-jpivot\wcf'
- Configure build.properties by setting the following properties, e.g.,:
  - o js.repo.home=c:/jaspersoft/jasperserver-repo
  - o js.repository.home=c:/Docume~1/jshih/.m2/repository (This is the Maven repository used by JasperServer)
- 'ant clean build'
- 'cd c:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot'
- Configure build.properties by setting the following properties:
  - o js.repo.home=c:/jaspersoft/jasperserver-repo
  - o js.repository.home=c:/Docume~1/jshih/.m2/repository
- 'ant clean build'
- Check and see if the jar files are created in c:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot\dist\web\WEB-INF\lib
- Copy the jar files to corresponding directories in c:\jaspersoft\jasperserver-repo:

Source	Target
c:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot\dist\web\WEB-INF\lib\jpivot-2.1.0.jar	C:\jaspersoft\jasperserver-repo\com\jaspersoft\jasperserver\jpivot-ui\jpivot\2.1.0
c:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot\dist\web\WEB-INF\lib\tbutils-wcf-	C:\jaspersoft\jasperserver-repo\com\jaspersoft\jasperserver\jpivot-

2.1.0.jar	ui\tbutils-wcf\2.1.0
c:\jaspersoft\jasperserver\jasperserver-jpivot\jpivot\dist\web\WEB-INF\lib\wcf-2.1.0.jar	C:\jaspersoft\jasperserver-repo\com\jaspersoft\jasperserver\jpivot-ui\wcf\2.1.0

## B.7 Troubleshooting

### B.7.1 Directory not found

Symptom:

compile:

```
[javac] Compiling 322 source files to c:\jaspersoft\jasperserver\jasperserver-jpivot\wcf\build\web\WEB-INF\classes
```

BUILD FAILED

```
c:\jaspersoft\jasperserver\jasperserver-jpivot\wcf\build.xml:180: c:\jaspersoft\jasperserver\jasperserver-jpivot\wcf\..\jpivot_repository\jakarta-tomcat-4.1.34\common\endorsed not found.
```

Cause:

The c:\jaspersoft\jasperserver\jasperserver-jpivot\wcf\jpivot\_repository\jakarta-tomcat-4.1.34\common\endorsed directory does not exist or has the wrong name

Resolution:

Make sure the directory is unpacked to the specific location.