JasperReports Server Community Project Source Build Guide

Release 5.5.0

http://community.jaspersoft.com

This is version 1013-JSO55-25 of the *JasperReports Server Source Build Guide*.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

JasperReports Server builds on JasperReports as a comprehensive family of Business Intelligence (BI) products, providing robust static and interactive reporting, report server, and data analysis capabilities.

This guide assists developers in obtaining, setting up, building, and running JasperReports Server from its source files.

Jaspersoft provides several other source of information to help extend your knowledge of JasperReports Server:

- Our Ultimate Guides document advanced features and configuration. The guides are available as downloadable PDFs.
- Our Business Intelligence Tutorials let you learn at your own pace, and cover topics for developers, system administrators, business users, and data integration users.
- Samples, which are installed with JasperReports, iReport, and JasperReports Server, are documented online. The samples documentation can be found on our community website.

This document describes how to build from a command line shell under Linux or Windows. It does not address the process of building within an IDE (Integrated Development Environment) such as Eclipse or IntelliJ.

1.1 Supported Build Configurations

The following table lists the target configurations that can be built from the source:

Application Server	Database
Tomcat, JBoss or GlassFish	PostgreSQL
	MySQL

1.2 JasperReports Server Source Code Archives

The following table lists the source code archive files for JasperReports Server:

File	Description	Documented In
jasperreports-server-cp-5.5.0-src.zip	JasperReports Server source code	**Chapter 2** **Chapter 3**
jasperjpivot-5.1.0-src.zip	JasperJPivot source code	Appendix **A.1**
jasperserver-portlet-5.1.0-src.zip	JasperReports Server Portlet source code	Appendix **A.2**

CHAPTER 2 COMPONENTS REQUIRED FOR SOURCE BUILD

The components and versions listed in this section are required in order to build and run JasperReports Server:

- **Check Your Java JDK**
- **Check Your Maven Version**
- **Check Your Application Server**
- **Check Your Database Instance**

2.1 Check Your Java JDK

The JasperReports Server source code can be compiled under Java 1.6 or 1.7. JasperReports Server does not run with versions of Java earlier than 1.6.

To check the version of your JDK (Java Development Kit), run the following command:

```
javac -version
```

To download the Java JDK, follow the instructions found at the Java web site: http://www.oracle.com/technetwork/java/javase/downloads/index.html.

The Oracle/Sun JDK is the certified Java platform for JasperReports Server. OpenJDK 1.6 is also supported.

2.2 Check Your Maven Version

Apache Maven is used to compile, build, and package the JasperReports Server source code. The JasperReports Server development team uses Maven because of its ability to manage third party dependencies via online repositories.

To download and install Maven you can use this URL: http://maven.apache.org/download.html#installation

To execute `mvn` from the command line, put the maven binary (`mvn` or `mvn.exe`) in your environment `PATH`. To check your Maven version, run this command:

```
mvn -version
```

For information about Maven, see section **A.4, "Maven Troubleshooting," on page 27**.

2.3 Check Your Application Server

To run JasperReports Server, you need an application server on the same computer as your source code.

Supported application servers are the following:

Application Server	Comments
Apache Tomcat	Source build can automatically deploy to this application server.
Glassfish	Source build can automatically deploy to this application server.
JBoss	Source build can automatically deploy to this application server.

2.4 Check Your Database Instance

To run JasperReports Server, you need a database instance.

Supported databases are the following:

Database	Comments
PostgreSQL	Source build automatically creates the `jasperserver` database.
MySQL	Source build automatically creates the `jasperserver` database.

CHAPTER 3 BUILDING JASPERREPORTS SERVER SOURCE CODE

This document describes how to build from a command line shell under Linux or Windows. It does not address the process of building within an IDE (Integrated Development Environment) such as Eclipse or IntelliJ.

3.1 Introduction to Buildomatic Source Build Scripts

The JasperReports Server source code comes with a set of configuration and build scripts based on Apache Ant known as the buildomatic scripts. These scripts are found in the following directory:

`<js-src>/jasperserver/buildomatic`

The buildomatic scripts automate most aspects of configuring, building, and deploying the source code. Apache Ant is bundled into the source code distribution to simplify the setup.

3.2 Downloading and Unpacking JasperReports Server Source Code

3.2.1 Downloading the Source Archive

Download the source code package zip for JasperReports Server from the Jaspersoft Community site:

http://community.jaspersoft.com

The download package is named:

`jasperreports-server-cp-5.5.0-src.zip`

3.2.2 Unpacking the Source Archive

Unpack the `jasperreports-server-cp-5.5.0-src.zip` file to a directory location, such as `C:\` or `/home/<user>`. The resulting location is referred to as `<js-src>` in this document.

Windows: <js-src> example is C:\jasperreports-server-cp-5.5.0-src

Linux: <js-src> example is /home/<user>/jasperreports-server-cp-5.5.0-src

3.2.3 Source Code Package Structure

After unpacking the zip file, the folder directory has the following structure:

Directory or file	Description
<js-src>/apache-ant	Bundled version of Apache Ant build tool
<js-src>/jasperserver	JasperReports Server open source code for core functionality
<js-src>/jasperserver-repo	Dependent jar files (not readily available publicly)

3.3 Check Apache Ant

The Apache Ant tool is bundled (pre-integrated) into the source code distribution package so you do not need to download or install Ant in order to run the buildomatic scripts. For example:

```
cd <js-src>/jasperserver/buildomatic
js-ant help   or
./js-ant help (Linux)
```

3.4 Configuring the Buildomatic Properties

The buildomatic scripts are found at the following location:

```
<js-src>/jasperserver/buildomatic
```

The buildomatic scripts are used to build the source code and to configure settings for all supported application servers and databases. The file for configuring these settings is `default_master.properties`. The source distribution includes a properties file that is specific to each type of database. You will add your specific settings to this file and rename it to:

```
default_master.properties:
```

When specifying paths with Apache Ant and Java under Windows, a single forward slash (/) normally works the same as “escaped” double backlashes (\\).

3.4.1 PostgreSQL

1. Go to the buildomatic directory in the source distribution:

   ```
   cd <js-src>/jasperserver/buildomatic
   ```

2. Copy the PostgreSQL specific file to the current directory and change its name to `default_master.properties`:

   Windows: `copy sample_conf\postgresql_master.properties default_master.properties`

   Linux: `cp sample_conf/postgresql_master.properties default_master.properties`

3. Edit the new `default_master.properties` file and set the following properties for your local environment:

Property	Examples
`appServerType`	`appServerType=tomcat7 (tomcat5/6, jboss/-as-7, glassfish2/3, skipAppServerCheck)`
`appServerDir`	`appServerDir = C:\\Program Files\\Apache Software Foundation\\Tomcat 7.0` `appServerDir = /home/user/apache-tomcat-7.0.26`
`dbHost`	`dbHost=localhost`
`dbUsername`	`dbUsername=postgres`

Property	Examples
dbPassword	dbPassword=postgres
js-path	js-path = C:\\jasperreports-server-cp-5.5.0-src\\jasperserver js-path = /home/<user>/jasperreports-server-cp-5.5.0-src/jasperserver
repo-path	repo-path = C:\\jasperreports-server-cp-5.5.0-src\\jasperserver-repo repo-path = /home/<user>/jasperreports-server-cp-5.5.0-src/jasperserver-repo

3.4.2 MySQL

1. Go to the buildomatic directory in the source distribution:
   `cd <js-src>/jasperserver/buildomatic`
2. Copy the MySQL specific file to the current directory and change its name to `default_master.properties`:

   Windows: `copy sample_conf\mysql_master.properties default_master.properties`
   Linux: `cp sample_conf/mysql_master.properties default_master.properties`
3. Edit the new `default_master.properties` file and set the following properties to your local environment:

Property	Examples
appServerType	appServerType = tomcat7 (or tomcat5/6, jboss, or glassfish2/3)
appServerDir	appServerDir = C:\\Program Files\\Apache Software Foundation\\Tomcat 7.0 appServerDir = /home/user/apache-tomcat-7.0.26
dbHost	dbHost = localhost
dbUsername	dbUsername = root
dbPassword	dbPassword = password
js-path	js-path = C:\\jasperreports-server-cp-5.5.0-src\\jasperserver js-path = /home/<user>/jasperreports-server-cp-5.5.0-src/jasperserver
repo-path	repo-path = C:\\jasperreports-server-cp-5.5.0-src\\jasperserver-repo repo-path = /home/<user>/jasperreports-server-cp-5.5.0-src/jasperserver-repo

3.5 Build Source Code

Now that your `default_master.properties` file has been setup, you can build the source code.

To build JasperReports Server:

1. Setup the `default_master.properties` file for your environment (as described above).
2. Start the database server.
3. Stop the application server unless it's GlassFish, which should be running.
4. Run the commands shown below:
   After executing each Ant target in **Table 3-1**, look for the message BUILD SUCCESSFUL.

Table 3-1 Commands for Building JasperReports Server

Commands	Description
cd <js-src>/jasperserver/buildomatic	
js-ant add-jdbc-driver	

Table 3-1 Commands for Building JasperReports Server, continued

Commands, continued	Description
`js-ant build-ce`	Builds the Community Project source code.
`js-ant create-load-js-db-ce`	Creates and loads the `jasperserver` database, imports core bootstrap data
`js-ant deploy-webapp-ce`	Deploys the jasperserver war file to the application server.

3.6 Set Java Options

JasperReports Server needs Java memory options that are larger than the standard defaults.

3.6.1 Set Increased JAVA_OPTS Settings

JasperReports Server needs greater heap and permgen memory settings in order for all functionality to operate. For testing your deployed JasperServer you should set your JAVA_OPTS to that same default values described in the Installation Guide. For a 64 bit system the settings would be similar to the following:

export JAVA_OPTS="$JAVA_OPTS -Xms1024m -Xmx2048m -XX:PermSize=32m -XX:MaxPermSize=512m" (Linux)

set JAVA_OPTS=%JAVA_OPTS% -Xms1024m -Xmx2048m -XX:PermSize=32m -XX:MaxPermSize=512m (Windows)

These settings should be added to your application server startup script:

Apache Tomcat: `<tomcat>/bin/setclasspath.sh` (.bat for Windows)

JBoss: `<jboss>/bin/run.sh` (.bat for Windows)

For details on setting Java memory options, please see section **A.1, "Setting Java JVM Options," on page 25**.

3.7 Starting JasperReports Server

You can now start your application server or restart GlassFish. Your database should already be running.

3.8 Logging into JasperReports Server

You can now login to JasperReports Server through a web browser:

Enter the login URL with the default port number:

`http://localhost:8080/jasperserver`

Log into JasperReports Server as jasperadmin:

User ID: `jasperadmin` Password: `jasperadmin`

If you are unable to login or have other problems, refer to **Appendix B, "Troubleshooting," on page 31**, or refer to the *JasperReports Server Installation Guide*, which provides additional troubleshooting information.

3.9 JasperReports Server Log Files

If you encounter any startup or runtime errors you can check the application server log files. For Apache Tomcat the log file will be found here:

`<tomcat>/logs/catalina.out`

There is also a `jasperserver.log` file. The debug output level can be increased by editing the `log4j.properties` file.

The JasperReports Server runtime log is here:

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

The `log4j.properties` file is here:

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

foob

CHAPTER 4 CREATE AND LOAD SAMPLE DATA

The steps in Chapter 3 build the JasperReports Server source code and load core data required to start the application. However, the steps do not create any sample data such. Sample data would include such things as sample reports to run and sample databases from which to pull data.

The steps below show how to create and load sample data.

4.1 Load Sample Data

The buildomatic scripts can load sample resources and sample databases. Note: In the procedure below, your `jasperserver` database will be deleted and re-created unless you choose 'n' for No when prompted.

1. Your `default_master.properties` should have already been created.
2. Start your database server.
3. Stop your application server.
4. Run the commands shown below:

Commands	Description
`cd <js-src>/jasperserver/buildomatic`	
`js-ant create-load-all-dbs-ce`	Creates and loads the `jasperserver` database Imports core bootstrap resources Creates and loads sample databases Imports sample resources (Choose 'n' when prompted if you do not want to recreate your `jasperserver` database.)

4.2 Generate Your Own Sample Resources

The build procedure can generate sample resources "from scratch". To do this, you can execute the sample creation code found in the following folder:

```
<js-src>/jasperserver/production-tests
```

The resources generated with this procedure are the same ones imported and used by the released version of JasperReports Server. Your jasperserver database will be deleted and re-created.

1. Your `default_master.properties` should have already been created.
2. Start your database server.
3. Stop your application server.
4. Run the commands shown below:

Commands	Description
`cd <js-src>/jasperserver/buildomatic`	
`js-ant re-init-js-db-ce`	Drop, create, and initialize the `jasperserver` database
`js-ant run-production-data-ce`	Generate sample resources using the processing logic from the production-test source code.

CHAPTER 5 ADDITIONAL BUILDOMATIC INFORMATION

The Ant-based buildomatic scripts contain support files for the setup and configuration of a number of databases and application servers. This chapter gives the locations of many of these files.

5.1 Detailed Description of the deploy-webapp-ce Target

The `deploy-webapp-ce` target carries out the following actions in your application server environment:

- Deletes any existing `jasperserver` WAR file.
- Copies the JDBC driver to the appropriate application server directory.
- Copies additional JDBC drivers to the application server to support DataSource creation in the UI
- Adds a data source definition to the appropriate application server directory.
- Deploys the newly built `jasperserver` WAR file.
- Deletes files within the application server work directory (to clear out compiled JSP files and other cached files).
- Under Tomcat, delete the old version of `<tomcat>/conf/Catalina/Localhost/jasperserver.xml` if present.

5.2 Running Ant in Debug Mode

Ant can be run with a -v (verbose) or a -d (debug) option to help with troubleshooting, for example:

```
js-ant -v build-ce
```

5.2.1 Regenerate Your Buildomatic Property Settings

If you change your `default_master.properties` file, buildomatic will automatically clean and regenerate all configuration settings. If you want to explicity clean and regenerate your settings manually you can run the following commands:

Commands	Description
`js-ant clean-config` `js-ant gen-config`	Clears the `buildomatic/build_conf/default` directory. Rebuilds the `buildomatic/build_conf/default` directory.

Anytime you modify the `default_master.properties` file, configuration settings get automatically re-generated into the `buildomatic/build_conf/default` folder.

5.3 Using Your Own Apache Ant: Get ant-contrib.jar

If you prefer to use your own version of Apache Ant, get the file `ant-contrib-1.0b3.jar`. This JAR enables conditional logic in Ant scripts.

1. Make sure you are using Apache Ant 1.8.1 or higher.
2. Copy the file `ant-contrib-1.0b3.jar` from the `<js-src>/apache-ant/lib` folder to your `<ant-home>/lib` folder:

   From:

   `<js-src>/apache-ant/lib/ant-contrib.jar` or

   `<js-src>/jasperserver/buildomatic/install_resources/extra-jars/ant-contrib.jar`

   To:

`<ant-home>/lib`	(General example)
`C:\apache-ant-1.8.1\lib`	(Windows example)
`/usr/share/java/apache-ant/lib`	(Linux example)
`/usr/share/ant/lib`	(Mac example)

5.4 Generated Property Files

After you set your database and application server property values, you run buildomatic scripts to generate the database and application server configuration files to run JasperReports Server. Generated property files are in the following directory:

`<js-src>/jasperserver/buildomatic/build_conf/default`

Some of the key configuration files are:

js.jdbc.properties

maven_settings.xml - (This is the maven settings file used by the source build)

More generated property files are in the following directory:

`<js-src>/jasperserver/buildomatic/build_conf/default/webapp`

Some of the configuration files in this directory are:

META-INF/context.xml

WEB-INF/hibernate.properties

WEB-INF/js.quartz.properties

Running `clean-config` removes these generated files. Running `gen-config` or any other target, regenerates these files.

5.5 Existing and Generated Database SQL Files

Buildomatic files that support various databases are located in:

`<js-src>/jasperserver/buildomatic/install_resources/sql/<db-type>`

The source code build procedure creates the `jasperserver` repository database schema using these files:

`js-create.ddl`

`js-drop.ddl`

When the buildomatic target `create-js-ddl-ce` is run, these database files are freshly generated for your specified database platform. The files are generated to the following location:

```
<js-src>/jasperserver/jasperserver-repository-hibernate/build-db/target/sql
```

Then, the files are automatically copied into their buildomatic directory location:

```
<js-src>/jasperserver/buildomatic/install_resources/sql/<db-type>
```

These generated files also overwrite the ones already existing in the buildomatic directory location.

5.6 Generated WAR File Location and deploy-webapp-ce Target

The JasperReports Server source code build creates a `jasperserver` WAR file. The build assembles the WAR file into the following location:

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

When the `build-ce` target is run, buildomatic assembles the `jasperserver` WAR file, and copies the file to this location for use by subsequent buildomatic targets:

```
<js-src>/jasperserver/buildomatic/install_resources/war/jasperserver
```

Later, when you run the buildomatic target `deploy-webapp-ce`, the following actions take place under Tomcat, for example:

Files: <js-src>/jasperserver/buildomatic/install_resources/war/jasperserver/*
Copied to: <tomcat>/webapps

File: <js-src>/jasperserver/buildomatic/build_conf/default/webapp/META-INF/context.xml
Copied to: <tomcat>/webapps/jasperserver/META-INF

Files: <js-src>/jasperserver/buildomatic/build_conf/default/webapp/WEB-INF/hibernate.properties
<js-src>/jasperserver/buildomatic/build_conf/default/webapp/WEB-INF/js.quartz.properties
Copied to: <tomcat>/webapps/jasperserver/WEB-INF

File: <js-src>/jasperserver/buildomatic/build_conf/db/postgresql/jdbc/postgresql-9.2-1002.jdbc4.jar
Copied to: <tomcat>/lib

Files: <js-src>/jasperserver/buildomatic/conf_source/db/app-srv-jdbc-drivers/*/jar
Copied to: <tomcat>/lib

5.7 Details on Database Load Build Targets

The buildomatic targets shown below are used in **Chapter 3, "Building JasperReports Server Source Code," on page 5** to create and populate the databases used with JasperReports Server. These targets consolidate and simplify the handling of the `jasperserver` database and the optional sample databases:

- `create-load-js-db-ce`
- `create-load-all-dbs-ce`

5.7.1 create-load-js-db-ce

This buildomatic target is a consolidation of the following targets:

- (`drop-js-db`, if necessary)
- `create-js-db`

- `init-js-db-ce`
- `import-minimal-ce`

Additionally, functionality has been added to check whether or not the `jasperserver` database already exists. If the database already exists, then a command line prompt asks the user whether or not to delete and re-create the database.

5.7.2 create-load-all-dbs-ce

This buildomatic target is a consolidation of the following targets:

- (`drop-js-db`, if necessary)
- `create-js-db`
- `init-js-db-ce`
- `import-minimal-ce`
- `import-sample-data-ce`
- (`drop-foodmart-db`, if necessary)
- `create-foodmart-db`
- `load-foodmart-db`
- (`drop-sugarcrm-db`, if necessary)
- `create-sugarcrm-db`
- `load-sugarcrm-db`

Additionally, functionality has been added to check whether or not the `jasperserver` database already exists. If the database already exists, then a command line prompt asks the user whether or not to delete and re-create the database. The same logic is used for the sample databases: foodmart and sugarcrm.

5.8 General Fresh Database Schema File

The consolidated database scripts do not regenerate the database schema file. Instead, the existing, default database schema files are used. To regenerate the database schema files, run the following target:

```
js-ant build-js-ddl-ce
```

The files are generated to the following location:

```
<js-src>/jasperserver/jasperserver-repository-hibernate/build-db/target/sql
```

Then, the files are automatically copied into their buildomatic directory location:

```
<js-src>/jasperserver/buildomatic/install_resources/sql/<db-type>
```

5.9 Older Buildomatic Commands

This section describes the original steps that were in place before the 4.7 release to build JasperReports Server source code. These steps can still be executed as shown below to build the source code.

The recommended way to build the JasperReports Server source code is to use the buildomatic scripts as described in **Chapter 3, "Building JasperReports Server Source Code," on page 5**. There are less command that need to be typed.

To build JasperReports Server using older Buildomatic commands:

1. Edit the `default_master.properties` file for your particular environment.
2. Start the database server.
3. Stop the application server (unless it's GlassFish which should be running).

   After you execute the first build target, the buildomatic scripts automatically configure the necessary properties and store these settings in the following directory:

   ```
   <js-src>/jasperserver/buildomatic/build_conf/default
   ```

After executing each Ant target below, look for the message `BUILD SUCCESSFUL`.

4. Execute the following steps at the command line:

Commands	Description
`cd <js-src>/jasperserver/buildomatic`	
`js-ant add-jdbc-driver`	Installs the JDBC driver to mvn local repository
`js-ant build-ce`	Builds the Community Project source code
`js-ant create-js-db`	If the jasperserver database already exists, first run `js-ant drop-js-db`
`js-ant create-sugarcrm-db`	Creates sample data for integration-tests
`js-ant load-sugarcrm-db`	
`js-ant create-foodmart-db`	Creates sample data for integration-tests
`js-ant load-foodmart-db`	Can run for 10 minutes or more
`js-ant build-js-ddl-ce`	Creates the database schema files for your database type
`js-ant init-js-db-ce`	Loads the schema into database
`js-ant run-production-data-ce`	Put core bootstrap and Sample data into the jasperserver db
`js-ant deploy-webapp-ce`	Deploys JasperReports Server to the application server

5.10 Manual Creation of Databases

JasperReports Server runs with a repository database that is typically named `jasperserver`. The creation of the jasperserver database and, additionally, the sample databases is automatically handled by the automated buildomatic steps. If you would like to manually create your databases, here is an example with the PostgreSQL database.

5.10.1 Manually Creating Databases: PostgreSQL

The scripts used by buildomatic to create and populate databases can also be executed manually.

Here is an example for PostgreSQL:

1. To create the `jasperserver` database, use a client tool to log into PostgreSQL:

```
cd <js-install>/buildomatic/install_resources/sql/postgresql

psql -U postgres -W
postgres=#create database jasperserver encoding='utf8';
postgres=#\c jasperserver;
postgres=#\i js-create.ddl
postgres=#\i quartz.ddl
postgres=#\q
```

2. Run the following commands if you want to create the sample databases:

```
cd <js-install>/buildomatic/install_resources/sql/postgresql

psql -U postgres -W
postgres=#create database sugarcrm encoding='utf8';
postgres=#create database foodmart encoding='utf8';
postgres=#\c sugarcrm;
postgres=#\i sugarcrm-postgresql.sql; (first make sure the file is unzipped)
postgres=#\c foodmart;
postgres=#\i foodmart-postqresql.sql; (first make sure the file is unzipped)
postgres=#\i supermart-update.sql;
postgres=#\q
```

5.10.2 Additional Databases

For information on manual setup of databases other than PostgreSQL, refer to the *JasperReports Server Installation Guide*.

CHAPTER 6 JASPERSOFT INTERNAL DEVELOPERS AND ADVANCED DEVELOPERS

This chapter is for Jaspersoft Internal Developers and for Advanced Developers who are customers who want to use some of the additional options that can be set via the buildomatic property settings.

6.1 Internal Developers and Advanced Developers

As of Release 4.7, Jaspersoft has setup an internal Maven repository using the Artifactory server software. This repository holds all third party component dependencies required to build the source code. It holds components directly or acts as a proxy for the standard public Maven repositories such as repo1.maven.org

This internal repository is convenient for internal Jaspersoft developers because the developer can point to one location to get all dependencies resolved.

In `default_master.properties`, internal developers should comment out `maven.build.type=repo` and `repo-path=<path>`:

```
# maven.build.type=repo
# repo-path=<path>
```

External developers (ie customers) who download the `jasperreports-sever-<ver>-src.zip` package from jaspersoft.com should set all the properties described in section **3.4, "Configuring the Buildomatic Properties," on page 6** before building JasperReports Server.

There are additional buildomatic property settings for advanced external developers. If an external developer is working within an enterprise or on a project that has an internal Maven repository server then the `mirror` value can be utilized. The following property settings and values will enable a local Maven repository:

```
maven.build.type=mirror
mvn-mirror=<repo-url>
```

Additionally, if an external developer has other build configurations to add, this can be accommodated with the `maven.build.type=custom` property setting. If this value is set, then the following file will be used as the template to set up the JasperReports Server build configuration:

```
<js-path>/buildomatic/conf_source/templates/maven_settings_custom.xml
```

The file above can be edited to add whatever configurations are desired.

After being processed by the buildomatic auto-setup, you can see the final maven settings file that is used for the JasperReports Server build at this location:

```
<js-path>/buildomatic/build_conf/default/maven_settings_custom.xml
```

6.2 Additional Properties in default_master.properties

The properties shown in the table below can be utilized for various customizations of the JasperReports Server build:

Property Setting	Purpose
SKIP_TEST_ARG=skipTests	Enable this property to skip unit test execution. This will speed the jasperserver-ce and jasperserver-pro source builds.
VERBOSE_LOGGING=true	Enable this property to increase the INFO logging from the Maven package. Maven is a verbose build tool, and as of Release 5.1 the logging level for JasperServer builds has been decreased.
OFFLINE_ARG=-o	Enable this property if you would like to build in “offline” mode. In order to run in offline mode you need to have successfully build JasperServer at least once.
SKIP_EXPORT_FILES=true	Enable this property to skip the copying of files which setup the command line import-export configuration. This will save time on file copying.
maven.build.type=repo	Use this setting for the build type if you have downloaded the source code zip package from the jaspersoft.com site and you are building the source code as a customer (external developer) would build it. You will also need to set the repo-path property. maven.build.type=repo is the default value used in the sample <dbType>_master.properties files.
maven.build.type=community	Use this setting for the build type if you are building only the Community source code. This setting supports Community members who have checked out JasperReports Server source code from the Community site: code.jaspersoft.com/svn/repos/jasperserver
maven.build.type=mirror	If an external developer has a central Maven style repository for their enterprise or project, this setting can be used to specify the local central repository. If this property value is set then the mvn-mirror property should also be set.
maven.build.type=custom	If an external developer’s build requires additional configurations, these can be supported using this property value. In this case, the following template file will be used: <js-path>/buildomatic/conf_source/templates/ maven_settings_custom.xml. This file can be manually edited to add additional configurations. The file will be processed by buildomatic and copied to its final location after executing a buildomatic command: buildomatic/build_conf/default/ maven_settings_custom.xml
mvn-mirror=<repo-url>	mvn-mirror=http://mvnrepo.jaspersoft.com:8081/artifactory/repo The value shown is the default repo-url used by Jaspersoft internal development.
repo-path=<path>	Set a local path value for this property if you are using maven.build.type=repo (this is the default configuration from the source code zip download from jaspersoft.com).

CHAPTER 7 BUILDING FROM PUBLIC SOURCES

The JasperReports Server Community Project public source code can be checked out from this location:

http://code.jaspersoft.com/svn/repos/jasperserver/trunk

Here is an example checkout command:

svn checkout --username anonsvn --password anonsvn

http://code.jaspersoft.com/svn/repos/jasperserver/tags/lastReviewed-trunk jasperserver

In this example, `tags/lastReviewed-trunk` represents the most current source code version that has been successfully built and reviewed by the Jaspersoft QA team.

On the Jaspersoft Community Site there is a wiki article covering the steps required for building the public source code.

This wiki page is located here:

http://community.jaspersoft.com/wiki/building-jasperreports-server-source-code

CHAPTER 8 BUILDING OTHER SOURCE CODE PACKAGES

This appendix describes how to build other JasperReports Server components:

- **Building JasperJPivot Source Code**
- **Building and Running Jasper-Portlet**

8.1 Building JasperJPivot Source Code

JasperJPivot is adapted from the JPivot open source project. It provides the web interface for Jaspersoft OLAP. In addition, JasperJPivot includes usability enhancements in the areas of navigation, configuration, and scalability.

You can get the source code package from the Jaspersoft technical support website (login required).

Download the source code package:

On the the Jaspersoft technical support website (login required).

Look for a file with the following name:

jasperjpivot-5.1.0-src.zip

Build the source code package:

Unpack the downloaded source code package zip file.

Next, follow the instructions found in the <unpacked-src>/Building-JasperJPivot-Source.pdf.

The process of building the JasperJPivot requires Apache Maven. For more information, see section **2.2, "Check Your Maven Version," on page 3**.

8.2 Building and Running Jasper-Portlet

The JasperReports Server portlet can be deployed to the Liferay Portal or to the JBoss Portal so that reports in the JasperReports Server repository can be displayed in your Portal environment.

Jaspersoft provides the source code for the JasperReports Server portlet so that developers can customize and extend the application for their specific needs. You can get the source code package from the Jaspersoft technical support website (login required).

The process of building the JasperReports Server Portlet WAR file requires Apache Maven. For more information, see section **2.2, "Check Your Maven Version," on page 3**.

Download the source code package:

On the the Jaspersoft technical support website (login required).

Look for a file with the following name:

JasperReportsServer-portlet-<ver>-src.zip

Build the source code package:

First, unpack the downloaded source code package zip file.

Next, follow the instructions in the Build Readme.txt file (found in the root unpacked folder).

Also, look for additional Readme.txt information in the <unpacked-folder>/docs directory.

For instructions on deploying and running the JasperReports Server Portlet, refer to the *JasperReports Server Administrator Guide* and the readme files at the root of the unpacked zip file.

APPENDIX A JAVA OPTIONS DETAILS

A.1 Setting Java JVM Options

JasperReports Server needs larger Java memory settings than default values in order to run properly. However, for development work, the settings used can be simpler than settings that are recommended for a production deployment. For full information on recommended JAVA_OPTS settings, please refer to the *JasperReports Server Installation Guide*.

A.1.1 Tomcat and JBoss JVM Options

The following tables present some typical settings of JVM options that affect JasperReports Server.

JVM Options on Windows	
Options for Java 1.6 and 1.7	set JAVA_OPTS=%JAVA_OPTS% -Xms1024m -Xmx2048m -XX:PermSize=32m -XX:MaxPermSize=512m set JAVA_OPTS=%JAVA_OPTS% -Xss2m -XX:+UseConcMarkSweepGC -XX:+CMSClassUnloadingEnabled
For Oracle	set JAVA_OPTS=%JAVA_OPTS% -Doracle.jdbc.defaultNChar=true

JasperReports Server doesn't provide a virtual X frame buffer on Linux. If your Linux applications are graphical, set the -Djava.awt.headless=true to prevent Java from trying to connect to an X-Server for image processing.

JVM Options on Linux and Mac OSX	
Options for Java 1.6 and 1.7	export JAVA_OPTS="$JAVA_OPTS -Xms1024m -Xmx2048m -XX:PermSize=32m -XX:MaxPermSize=512m" export JAVA_OPTS="$JAVA_OPTS -Xss2m -XX:+UseConcMarkSweepGC -XX:+CMSClassUnloadingEnabled"
For Oracle	export JAVA_OPTS="$JAVA_OPTS -Doracle.jdbc.defaultNChar=true"

There are a number of ways to set JVM options. For example, you can add your JAVA_OPTS settings to these files:

File	Add JVM Options Below the Lines Shown Here:
<tomcat>/bin/setclasspath.bat	`set JAVA_ENDORSED_DIRS=%BASEDIR%\common\endorsed`
<tomcat>/bin/setclasspath.sh	`JAVA_ENDORSED_DIRS="$BASEDIR"/common/endorsed`

File	Add JVM Options Below the Lines Shown Here:
<tomcat>/bin/setenv.bat or <tomcat>/bin/setenv.sh	JAVA_OPTS setting can go anywhere in this file.
<jboss>/bin/run.bat <jboss>/bin/run.sh	`set JAVA_OPTS=%JAVA_OPTS% -Dprogram.name=%PROGNAME%` or `export JAVA_OPTS="$JAVA_OPTS -Dprogram.name=$PROGNAME"`

For information on recommended JAVA_OPTS settings for all certified application servers, please refer to the *JasperReports Server Installation Guide*.

A.2 Build Troubleshooting

A.2.1 Name Undefined Error (Old Ant Version)

If you are not using the bundled version of Apache Ant included with the JasperReports Server source code package, you could get the following error when running the buildomatic scripts:

```
BUILD FAILED
c:\js-builds\jasperserver\buildomatic\install.xml:6: Problem: failed to create task or
type if
Cause: The name is undefined.
 Action: Check the spelling.
 Action: Check that any custom tasks/types have been declared.
 Action: Check that any <presetdef>/<macrodef> declarations have taken place.
```

Solution:

The buildomatic scripts require ant version 1.8.1 or higher. Additionally, the ant-contrib.jar file needs to be included in your ant/lib directory. If you are running with your own ant version, you can copy this jar to your <ant-home>/lib directory:

From:

<js-src>/apache-ant/lib/ant-contrib.jar or
<js-src>/jasperserver/buildomatic/extra-jars/ant-contrib.jar

To:

<ant-home>/lib	(General example)
C:\apache-ant-1.8.1\lib	(Windows example)
/usr/share/java/apache-ant/lib	(Linux example)
/usr/share/ant/lib	(Mac example)

A.3 Database Troubleshooting

The most common error encountered when building JasperReports Server involves the database. These errors often result from not being able to connect to the database. For information about database connection problems, see the Troubleshooting Appendix of the *JasperReports Server Installation Guide*.

A.4 Maven Troubleshooting

A.4.1 Maven Error on Linux or Mac

If Maven is installed on Linux via rpm, apt-get, or yum (or on Mac) then it will be typical that the Maven binary and the Maven libraries are in separate locations. This can potentially cause a problem with the source build.

A.4.1.1 /usr/boot Does Not Exist Error

When building under Linux or Mac, it is possible to get an error similar to the following:

```
BUILD FAILED
/home/devuser/js-builds/jasperserver/buildomatic/bin/dev.xml:91:
/usr/boot does not exist
```

The Buildomatic scripts attempt to find the MAVEN_HOME setting and can be unsuccessful when the maven binary is installed in the `/usr/bin/mvn` location. The workaround is to update you `default_master.properties` file:

`cd <js-src>/jasperserver/buildomatic`

edit `default_master.properties`

Un-comment the maven.home line so that it looks like this:

`maven.home = /usr/share/maven2` (Linux

For Mac, the location of the Maven library files typically slighly different:

`maven.home = /usr/share/maven` (Mac)

A.4.2 Clear JasperReports Server Artifacts in Maven Local Repository

If you have an existing source build environment and you add new code, such as a bug fix source patch update, you can clear the JasperReports Server artifacts in your Maven local repository to ensure that the newly built artifacts contain the necessary new content. Maven updates the artifacts automatically, but if you havetrouble building or pulling in the modified code, you can try deleting these artifact trees.

To clear existing JasperReports Server artifacts:

1. Go to the repository directory:

   ```
   cd <home-dir-path>/.m2/repository
   ```

2. Remove the old versions by deleting the following directories and their contents:

   `com/jaspersoft`: Community Project artifact tree

   `jaspersoft`: Commercial version artifact tree

A.4.3 Clear Entire Local Repository

If you want to completely rebuild everything, remove all of the cached jars in your Maven local repository. To do this you can delete (or rename) the entire local repository.

Then when you build JasperServer, all dependencies are re-downloaded.

```
cd <home-dir-path>/.m2
```

rm -rf `repository`

A.4.4 Maven Warnings

Maven generates verbose warnings during the artifact validation process.

The following example shows a warning, even though the required JAR file was downloaded successfully:

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

A.4.5 Maven Error: Transferring Files

With the Maven build, there are many files that are downloaded on the very first build. It is not unusual to get an error downloading a file. You can usually get around a file transfer error by kicking off the build again.

In the following example, there was a transfer error on the castor.jar file:

```
[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:\jasperserver-3.7.0-src\jasperserver-repo\jasperserver-maven)
```

Such problems can be fixed by re-running the `mvn install` command, which effectively restarts the build.

A.4.6 Maven Build Error: Failed to Resolve Artifact

In some cases, Maven may return the following error:

```
[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:3.0.0
  2) org.springframework.security:spring-security:jar:2.0-m2
  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:3.0.0

  Try downloading the file manually from the project website.
    mvn install:install-file -DgroupId=jasperreports -DartifactId=jasperreports \
    -Dversion=3.0.0 -Dpackaging=jar -Dfile=/path/to/file
Path to dependency:
  1) com.jaspersoft.jasperserver.api.metadata:jasperserver-api-metadata:jar:3.1.0
  2) jasperreports:jasperreports:jar:3.1.0
----------
2 required artifacts are missing.


for artifact:
  com.jaspersoft.jasperserver.api.metadata:jasperserver-api-metadata:jar:3.1.0
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:\jasperserver-3.7.0-src\jasperserver-repo)
```

This error may indicate that the setting.xml file doesn't point correctly to the jasperserver-repo directory.

In this case, many of the dependent JARs cannot be found. To resolve the problem, double-check the $HOME/.m2/settings.xml file and ensure that it properly specifies the <js-src>/jasperserver-repo directory. See section **5.3, "Creating the settings.xml File," on page 20**.

A.4.7 Old Maven Binary

In general, it is best to use the most current stable version of the Maven tool.

Jaspersoft has found that Maven version 3.0.3 will get errors resolving dependencies; therefore this version should not be used.

A.5 Other Build Troubleshooting

A.5.1 Error When Building Database Scripts

when compiling in the jasperserver-repository-hibernate/build-db directory, you might see an error containing the following message:

```
[ERROR] BUILD ERROR
[INFO] ------------------------------------------------------------------------
[INFO] Error executing ant tasks
Embedded error: Source file does not exist!
```

The most likely problem is that your .m2/settings.xml file doesn't point to the correct source location, and the build step did not find the Quartz scripts. The settings.xml file should contain the path to the quartz script corresponding to your database, for example:

<js.quartz.script>/home/<user>/<js-src>/jasperserver/scripts/quartz/tables_<database>.sql</js.quartz.script>

If you use the buildomatic scripts you should not get this kind of error. For more information, see section **5.3, "Creating the settings.xml File," on page 20**.