



JASPERSERVER COMMUNITY EDITION INSTALLATION GUIDE

RELEASE 3.7

<http://jasperforge.org>

© 2010 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.

This is version 0110-JSO37-12 of the *JasperServer Community Edition (CE) Installation Guide*.

TABLE OF CONTENTS

1	Introduction	9
1.1	Conventions	10
1.2	Java Version Supported	10
1.3	JasperServer Distributions	10
1.4	Installer Distribution Support	11
1.4.1	Installer Distribution Components	11
1.4.2	Installing with Existing Components	11
1.4.3	Running Components as Windows Services	11
1.5	WAR File Binary Distribution Support	12
1.6	Release Notes	12
1.7	Prerequisites for Installation	12
1.8	System Requirements	12
1.9	Support for Internationalization	13
2	Installing JasperServer	15
2.1	Installation Steps	15
2.1.1	Welcome	15
2.1.2	Accepting the License Agreement	15
2.1.3	Installation Directory Location	16
2.1.4	Selecting Tomcat Configuration	16
2.1.5	Selecting MySQL Configuration	16
2.1.6	Installing Sample Data	17
2.1.7	Installing iReport	17
2.1.8	Ready to Install	17
2.1.9	Installation Complete Screen	17
2.1.10	Logging Into JasperServer	18
2.2	Post-Installation Steps	18
2.2.1	Updates Made During Installation	18
2.2.2	Installer Output Log File Location	19
2.2.3	Checking your Java JVM Options	19

3	Starting and Stopping JasperServer	21
3.1	Using JasperServer Start/Stop Scripts	21
3.2	Using Start/Stop Scripts Without Bundled Installation	21
3.3	Logging Into JasperServer	22
3.4	Starting the Included iReport	22
3.5	JasperServer Log Files	22
4	Uninstalling JasperServer	23
5	Installing From the WAR File Distribution	25
5.1	Applications Supported by the WAR File Distribution	25
5.2	Obtaining the WAR File Distribution Zip	26
5.3	Unpacking the WAR File Distribution Zip	26
5.4	Introduction to Buildomatic Scripts	26
5.5	Pre-Installation Steps	26
5.5.1	Checking Your Java Installation	26
5.5.2	About Bundled Apache Ant	27
5.5.3	Checking Your Application Server	27
5.5.4	Checking Your Database Server	27
5.6	Configuring the Buildomatic Scripts	27
5.6.1	Creating your Default Master Properties File	28
5.6.2	Regenerating Your Buildomatic Settings	28
5.7	Installing JasperServer	28
5.7.1	Deploying to GlassFish	29
5.7.2	Creating Databases Manually	29
5.8	Setting Java JVM Options	29
5.9	Starting JasperServer	30
5.10	Logging Into JasperServer	30
5.10.1	JasperServer Heartbeat	30
5.11	Troubleshooting your JasperServer Configuration	31
5.11.1	JasperServer Startup Problems	31
5.11.2	Error Running a Report	31
5.12	Running the Import and Export Utilities	31
5.12.1	Running Export from Buildomatic	31
5.12.2	Running Import from Buildomatic	32
5.12.3	Running Import or Export from the Scripts Directory	32
6	Additional Installation Information	33
6.1	Setting JVM Options for Application Servers	33
6.1.1	Tomcat and JBoss JVM Options	33
6.1.2	Tomcat as a Windows Service JVM Options	34
6.1.3	GlassFish JVM Options	35
6.2	Additional Buildomatic Configuration Information	35
6.2.1	Buildomatic: Generated Property Files	36
6.2.2	Buildomatic: SQL Scripts Location	36

6.2.3	Buildomatic: Database Creation Statements Location	36
6.2.4	Buildomatic: JDBC Driver Locations	37
6.2.5	Buildomatic: JasperServer WAR File Location	37
6.2.6	Buildomatic: Sample Data Catalog ZIP Files	37
6.3	Additional Notes on the MySQL Database	37
6.3.1	Manual Creation of the JasperServer Database	38
6.3.2	Manual Import of Default Users	38
6.3.3	Manual Creation of Sample Databases	38
6.4	Notes on the Hibernate Properties File	39
6.5	Installed JDBC Driver Locations	39
6.6	Notes on Database Connection for Tomcat	39
6.7	Notes on Datasource Definition for JBoss	39
6.7.1	Notes on Extra JBoss Configuration Step	40
6.8	Notes on Database Connection for GlassFish	40
6.9	Notes on JasperServer Logging	40
6.10	Report Scheduling Configuration with Quartz	40
6.10.1	Mail Server Configuration Settings	41
6.10.2	Database settings for the Quartz Driver Delegate Class	41
6.10.3	Settings for the Report Scheduler Web URI	41
6.10.4	Settings for the Quartz Table Prefix	42
6.10.5	Settings for Import-Export	42
6.11	Notes on Updating XML/A Connection Definitions	42
7	Upgrade from JasperServer 3.5 to 3.7	45
7.1	Standard Upgrade Steps	45
7.2	Backing Up Your JasperServer 3.5 Instance	46
7.3	Exporting Your 3.5 Repository Data	46
7.3.1	Exporting With Buildomatic Scripts	46
7.3.2	Exporting With js-export Script	46
7.4	Preparing the JasperServer 3.7 WAR File Distribution	47
7.5	Configuring Buildomatic for Your Database and Application Server	47
7.6	Upgrading to JasperServer 3.7	47
7.7	Starting JasperServer 3.7	48
7.8	Logging Into JasperServer 3.7	48
7.8.1	Clearing Your Browser Cache	48
7.8.2	Logging Into JasperServer	48
7.9	Additional Notes on JasperServer Upgrade	48
7.9.1	Handling JasperServer Customizations	48
7.9.2	Clearing the Application Server Work Directory	48
7.9.3	Clearing the Repository Cache Table	49
7.9.4	Updating the XML/A Connections (Optional)	49
7.9.5	Upgrading the Liferay Portal	49
8	Upgrade Using SQL Upgrade Script	51
8.1	Backing Up Your JasperServer 3.5 Instance	51

8.1.1	Backing Up Your JasperServer CE WAR File	51
8.1.2	Backing Up Your JasperServer Database	51
8.2	Preparing the JasperServer 3.7 WAR File Distribution	52
8.3	Configuring Buildomatic for Your Database and Application Server	52
8.4	Upgrading to JasperServer 3.7	52
8.5	Starting JasperServer 3.7	53
8.6	Logging Into JasperServer 3.7	53
8.6.1	Clearing Your Browser Cache	53
8.6.2	Logging Into JasperServer	53
9	Upgrade Notes for JasperServer 3.0 and 3.1	55
10	Changing Password Encryption in JasperServer	57
10.1	Backing Up Your JasperServer Database	57
10.2	Stopping Your Application Server	57
10.3	Running the Repository Export Utility	58
10.4	Specifying Encryption Settings in the JasperServer WAR	58
10.5	Specifying Encryption Settings for the Import Utility	59
10.6	Recreating the JasperServer Database	59
10.6.1	Dropping and Recreating in MySQL	59
10.6.2	Dropping and Recreating in PostgreSQL	60
10.7	Importing Your Repository Data	60
10.8	Starting the Application Server	60
10.9	Logging Into JasperServer	60
11	Configuring the Import-Export Utilities	61
11.1	Import-Export Configuration Files	61
11.2	Changing Your Configuration Settings	62
11.3	Deploying a Database Driver	62
11.4	Running Import or Export	63
Appendix A	Troubleshooting	65
A.1	Installer Freezes	65
A.2	Database Connectivity Errors	66
A.2.1	Testing the Database Connection	66
A.2.2	Configuration File Locations	66
A.2.3	Special Configuration Case under Tomcat	66
A.2.4	Connect to Installed/Bundled Version of MySQL	67
A.2.5	Maximum Packet Size in MySQL	67
A.3	Error Running a Report	67
A.4	Database Error after Changing MySQL Port Number	68
A.5	Case Sensitivity for Table and Column Names	68
A.6	Java Out of Memory Error	68
A.7	Error Running Scheduled Report	68
A.8	Exporting a Repository That Contains UTF-8	68

A.8.1	Error During JasperServer 1.2 Export	68
A.9	JBoss Modifications	69
A.9.1	JBoss 4.2 XML/A Connection Fix	69
A.9.2	JBoss Large INFO Log Message on Analysis Drill-through.	69
A.9.3	JBoss 4.0 Log4j Error on Startup	69
A.9.4	JBoss 5.0.1 and 5.1.x Error	69
A.10	PostgreSQL: Job Scheduling Error	70
A.11	Error Running Buildomatic Scripts	70
A.11.1	Missing Java JDK	70
A.11.2	Forgot to Copy the File ant-contrib-3.7.jar.	70
A.12	Troubleshooting on Solaris	71
A.13	Disabling User Session Persistence in Application Servers	71

1 INTRODUCTION

JasperServer builds on JasperReports as a comprehensive family of Business Intelligence (BI) products, providing robust static and interactive reporting, report server, and data analysis capabilities. These capabilities are available as either stand-alone products, or as part of an integrated end-to-end BI suite utilizing common metadata and providing shared services, such as security, a repository, and scheduling. JasperServer exposes comprehensive public interfaces enabling seamless integration with other applications and the capability to easily add custom functionality.

The heart of the Jaspersoft BI Suite is JasperServer, which provides the ability to:

- Easily create new reports using an intuitive web-based drag and drop Ad Hoc reporting interface.
- Efficiently and securely manage many reports.
- Interact with reports, including entering parameters and drilling on data.
- Arrange reports and web content to create appealing, data-rich dashboards that quickly convey business trends.
- Host separate organizations securely and transparently.

If you want to extend your knowledge of Jaspersoft BI software, our Ultimate Guides document advanced features and configuration. They also include best practice recommendations and numerous examples. The guides are available as downloadable PDFs. Community edition users can purchase individual guides or bundled documentation packs from the Jaspersoft [online store](#).

This chapter contains the following sections:

- **Conventions**
- **Java Version Supported**
- **JasperServer Distributions**
- **Installer Distribution Support**
- **WAR File Binary Distribution Support**
- **Release Notes**
- **Prerequisites for Installation**
- **System Requirements**
- **Support for Internationalization**

1.1 Conventions

For clarity, this document uses the following conventions when referring to file locations, user names, passwords, and other values that are specific to your environment:

Convention	Description
<js-install>	The root directory where JasperServer will be installed. For manual installations, the directory where you unpack the WAR file distribution.
<tomcat>	The directory where Apache Tomcat is installed. If you use the instance of Tomcat that is bundled by the installer, <tomcat> is located in <js-install>.
<jboss>	The directory where JBoss is installed.
<glassfish>	The directory where GlassFish is installed.
<mysql>	The directory where MySQL is installed. If you use the instance of MySQL that is bundled by the installer, <mysql> is located in the <js-install> directory.
<java>	The directory where java is installed.
jasperadmin/jasperadmin	The user ID and password of the default JasperServer administrator.
jasperdb/password	The user name and password for the default database user.

1.2 Java Version Supported

JasperServer supports both Java 1.5 and Java 1.6. Versions earlier than Java 1.5 are not supported.

1.3 JasperServer Distributions

There are two main distribution packages for JasperServer.

Distribution Package	Description
Installer	Runs on Windows and Linux.
WAR File Binary Distribution Zip	Used for manual installation on Windows, Linux, Mac, and other platforms.

The installers have the capability of installing JasperServer, automatically configuring the JasperServer database, and optionally installing sample data that highlight JasperServer features.

The WAR file binary distribution contains the JasperServer web archive file as well as scripts to create and load the JasperServer database. The WAR file distribution supports additional applications that are not supported by the installers.

1.4 Installer Distribution Support

The installers support the following operating systems:

Platform	Versions supported
Windows	XP Vista Windows 7
Linux	Red Hat Enterprise Linux SUSE Ubuntu And additional Linux distributions

1.4.1 Installer Distribution Components

The installer is designed so that it is easy to get JasperServer up and running quickly. JasperServer requires the Java environment, an application server, and database to run. The installer distribution contains bundled versions of these components:

Component	Description
JasperServer Application	WAR file and configuration support scripts.
iReport Report Designer	Latest version of iReport for Netbeans (optional).
Java 1.5 Runtime	Runs web application container (optional).
MySQL Database	Database server. Can use the bundled version or an existing version.
Apache Tomcat	Web application container: Can use the bundled version or an existing version.
JasperServer Documentation	Found in the <js-install>/docs directory.

1.4.2 Installing with Existing Components

You can choose to deploy the bundled application or if you have existing components, the installer can deploy to these components. For instance, if you already have Tomcat on your computer you can choose an “existing” Tomcat. If you would like the installer to install Tomcat for you, you can choose the “bundled” Tomcat. Both Apache Tomcat and the MySQL database can be independently used as bundled or existing instances. For information on the specific versions of third party applications that are supported by the installer refer to the JasperServer release notes for the distribution you are using. The release notes are found in the root of the installation directory.

1.4.3 Running Components as Windows Services

In order for JasperServer to launch automatically whenever you reboot your server host, Apache Tomcat and the MySQL database must be configured as Windows services that launch automatically after system start-up.

If you would like Tomcat and MySQL to run as Windows services, they must be installed separately from JasperServer. First download and install Tomcat and MySQL, and then configure them as Windows services according to your operating system. You must configure both services to startup automatically, and in the dependencies, Tomcat must depend on MySQL. In other words, MySQL must start first and be running before Tomcat can be started. When the configuration is ready, run the JasperServer installer and choose both Tomcat and MySQL as existing applications when prompted.

1.5 WAR File Binary Distribution Support

The WAR file binary distribution is the package you would use to do a manual installation of the JasperServer application. The WAR file supports many more applications than are supported by the installers. By using the WAR file to install JasperServer manually, you can use a database other than MySQL and an application server other than Apache Tomcat.



For a complete list of applications supported by the WAR file distribution, refer to the release notes that are included in the root directory of the distribution.

Contents of the WAR file binary distribution:

Content Item	Description
JasperServer WAR file archive	This contains all of the JasperServer class files and dependent jars.
JasperServer Database Scripts	SQL scripts for each supported database.
JasperServer Standard Sample Data	Sample data that highlights JasperServer features.
JasperServer Extra Samples	Web Service example applications, sample reports, custom data source examples, and other sample files.
JasperServer Documentation	Guides for end users and administrators.

1.6 Release Notes

Release notes are included with each distribution and with each new update to a distribution.

Not all applications are immediately supported when a new JasperServer version is released. For instance, some applications require additional testing beyond what is completed for the initial General Availability (GA) release. To find out exactly what applications are supported with a particular distribution refer to the release notes found in that distribution.

1.7 Prerequisites for Installation

JasperServer relies on third-party products, such as application servers and relational databases. Unless you use the ones included with the JasperServer installer, these third party products must be installed and configured before beginning a JasperServer installation. Refer to the sections below that relate to your preferred application server and database.

1.8 System Requirements

The following table contains the minimum and recommended resources for a full installation, including MySQL and an application server. The values are based on our own testing. You may find that JasperServer can run on systems with fewer resources or slower systems than stated in the minimum resources column. At the same time, it is possible to run out of resources with the recommended configuration. The success of your deployment depends on the intended load of the system, the number of concurrent users, the data sets and whether the databases are installed on the same system as the JasperServer.

Operating System	Resource	Footprint	Minimum	Recommended
Windows	Disk	~600MB	10 GB free	40 GB +
	RAM		512 MB	1 GB +
	Processor		1 GHz (single Pentium)	1.5 GHz + (multi-core Pentium)
MAC (OSX)	Disk	~600MB	10 GB free	40 GB +
	RAM		512 MB	1 GB +
	Processor		1 GHz (single Pentium)	1.5 GHz + (multi-core Pentium)
Linux	Disk	~600MB	10 GB free	40 GB +
	RAM		512 MB	1 GB +
	Processor		1 GHz (single Pentium)	1.5 GHz + (multi-core Pentium)
Solaris	Disk	~600 MB	10 GB free	40 GB +
	RAM		512 MB	1 GB +
	Processor		UltraSparc II	
AIX	Disk	~600 MB	10 GB free	40 GB +
	RAM		512 MB	1 GB +
HP-UX	Disk	~600 MB	10 GB free	40 GB +
	Processor		512 MB	1 GB +

1.9 Support for Internationalization

JasperServer supports the full Unicode character set using UTF-8 encoding. JasperServer also depends on the underlying database and application server to support the UTF-8 character encoding. If you are using the bundled Tomcat and MySQL software, UTF-8 is configured by default.

2 INSTALLING JASPERSERVER

This chapter contains the following sections:

- [Installation Steps](#)
- [Post-Installation Steps](#)

2.1 Installation Steps

When you run the installation executable, you are prompted to specify information about the third party applications that JasperServer relies on. These third party applications are Apache Tomcat and the MySQL database.



When you run the installer against an existing database instance, the database must be running at install time.

To begin, run the installer on the application server host.

In Windows, the installer is an executable file that you can double-click to run. For example, double-click the following:

```
jasperserver-ce-3.7-windows-installer.exe
```

In Linux, the installer is a .bin file; you can run it from the command line or from a graphical environment. To start the installer from the command line, login with an account that has administrative privileges and open a bash shell. At the command line, enter the name of the installer file. For example:

```
./jasperserver-ce-3.7-linux-installer.bin
```

Whether you run the installer from the command line or in a graphical environment, you are prompted for the same information. The following sections describe these prompts, and assume you are in a graphical environment. If you are installing from the command line, use your keyboard to specify the same details. For example, with the license text, instead of clicking **I accept the agreement**, you press **Y** and press **Enter**.

2.1.1 Welcome

The first step introduces the installer and allows you to continue or exit. Click **Next**.

2.1.2 Accepting the License Agreement

You are prompted to read and accept the license agreement. Read the agreement, agree to the terms by clicking **I accept the agreement**, and click **Next**. On the command line, you must page through several screens of text to read the full agreement.

If you do not accept the agreement, you must exit the installer.

2.1.3 Installation Directory Location

You are prompted for the directory where JasperServer is installed referred to as the <js-install> directory. Accept the default or click **Browse** and select a different location, and click **Next**. On the command line, press Enter to accept the default. To choose a different directory location, enter that location at the prompt.

The default <js-install> directory depends on your operating system:

Windows: C:\Program Files\jasperserver-ce-3.7.0

Linux: <USER_HOME>/jasperserver-ce-3.7.0

2.1.4 Selecting Tomcat Configuration

JasperServer requires a web application server in order to run. The JasperServer installer is pre-configured to run with the Apache Tomcat server. There are two options available for your Tomcat configuration.

The first option is to choose a bundled Tomcat. If you choose this option, the installer puts an instance of Tomcat 5 onto your system. Click **Next**. You are prompted for the server port and shutdown port that Tomcat will use. Most users accept the default values that are displayed. Accept the default values or enter alternate values and then click **Next**.

The second option is to choose an existing Tomcat. If you already have an instance of Tomcat on your system, then you can choose this option. Choose the existing Tomcat option and click **Next**. You are prompted for its location. Enter the correct location for Tomcat or click **Browse** to locate and select another location. Click **Next**. You are prompted for Tomcat's server port and shutdown port. Accept the default values or enter alternate values and then click **Next**.

2.1.5 Selecting MySQL Configuration

JasperServer requires a database in order to run. The JasperServer installer is pre-configured to run with the MySQL database. There are two options available for you MySQL configuration.

The first option is to choose a bundled MySQL. If you choose this option, the installer puts an instance of MySQL 5 onto your system. Click **Next**. The default MySQL port 3306 will be used. The installer will also create a MySQL database user with administrator privileges and credentials of jasperdb/password. If the installer finds that port 3306 is already in use, you are prompted to pick an alternate port. In this case, pick an alternative port value and click **Next**.

Values to be entered or set to defaults for the Bundled MySQL configuration:

Parameter	Default Value and Description
Port	3306 - User must choose an alternate port if 3306 is in use.
Database User Name	Hard coded default: jasperdb - The installer creates this user which is used to connect to the jasperserver database
Database User Password	Hard coded default: password - The installer uses this password for the jasperdb account.



Jaspersoft recommends that you change your database user password from the default value to a new, secure value.

The second option is to choose an existing MySQL. If you already have an instance of MySQL running on your system, then you can choose this option. Choose the existing MySQL option and click **Next**. You are prompted for the location of MySQL, and the port to use. Note that the MySQL instance must reside on your local machine (i.e. localhost or 172.0.0.1). Enter the correct location for MySQL or click **Browse** to locate and select another location. Click **Next**. You are prompted for the user name and password of the MySQL administrative user. Enter this user name and password information and click **Enter**.

Values to be entered or set to defaults if installing to an existing installation of MySQL:

Parameter	Default Value and Description
Binary Directory	The directory where the mysql and mysqladmin binaries are located.
Port	The port number that MySQL uses (default is 3306).
IP or Host Name	The value is hard coded to 172.0.0.1. Note that your existing MySQL instance must reside on the local machine.
MySQL Root User Name	User name of the database administrative user.
MySQL Root Password	Password of the database administrative user. *
Database User Name	jasperdb - The installer creates this user which is used to connect to the jasperserver database.
Database User Password	password - The installer uses this default password for the jasperdb account. *

* The installer cannot handle special characters at the end of a password string. Incompatible characters include: & ; \$



Jaspersoft recommends that you change your database user password from the default value to a new, secure value.

2.1.6 Installing Sample Data

JasperServer can be installed with sample data that can help you evaluate its features. Sample data and resources included are the following:

- Sugar CRM data that simulates three years of operations for a fictitious company that relies on the SugarCRM open source application
- Foodmart data that simulates three years of operations for a fictitious company.
- JasperServer repository resources such as Reports, Analysis Views, Ad Hoc Topics, Domains, Data Sources, and Input Controls.
- Jaspersoft strongly recommends that you install this data, unless you are not interested in testing or evaluating with the default sample data. Click **Yes** to install the sample data and click **Next**.

2.1.7 Installing iReport

iReport is the leading GUI-based JasperReports creation tool. It has the capability of communicating directly with a JasperServer instance and can thus retrieve existing JasperReports from a JasperServer instance for editing, uploading or execution.

In the installer, iReport comes pre-configured with a plugin that allows it to communicate with JasperServer via the web services interface.

If you would like to install iReport click **Yes**.

2.1.8 Ready to Install

The components are now ready for installation. Click **Install** or **Next** to continue. Installation can take a number of minutes.

2.1.9 Installation Complete Screen

After the JasperServer files have been installed, you will see the final installation screen. There are several Post-Installation options that you can choose from, each with its own checkbox. Simply click to make your choices and then click **Finish**.

- View Release Notes - If you choose to view the Release Notes, they are displayed in a new window. If you are running from the command line, you can page through the Release Notes by pressing the Enter key.

- Launch JasperServer Now - If you choose to launch JasperServer from the installer, the installer exits and the application server starts. It takes a few moments for the server to start up. When this is complete, the JasperServer login page appears in your system default Browser. For more information on logging in, see section 3.3, “Logging Into JasperServer,” on page 22.



Starting JasperServer from the installer is dependent on your Tomcat and MySQL configuration choices. The JasperServer start/stop scripts only control the bundled applications that you chose to be installed. For more information, see chapter 3, “Starting and Stopping JasperServer,” on page 21.

Also, if you chose to view the Release Notes, JasperServer will not startup until you close the Release Notes.

- Opt-in for JasperServer Heartbeat - The JasperServer heartbeat will help Jaspersoft create better products by improving our understanding of customer installation environments. When the heartbeat is enabled, the server sends anonymous system and version information to Jaspersoft via https. For more information, see section 5.10.1, “JasperServer Heartbeat,” on page 30.

You can later enable or disable the heartbeat by modifying the jasperserver/WEB-INF/applicationContext-logging.xml file.

2.1.10 Logging Into JasperServer

You should now be ready to log into JasperServer. For information on default login credentials, go to section 3.3, “Logging Into JasperServer,” on page 22.

2.2 Post-Installation Steps

2.2.1 Updates Made During Installation

This first sub-section is informational. It lists the standard updates that are made by the installer to your local environment if you install to already existing applications.

Updates to the application server:

If you installed to an existing Tomcat, the following modifications were attempted to the Tomcat environment:

File or Directory	Updates
Windows: bin/setenv.bat Linux: bin/setenv.sh	This file gets newly created. Sets increased Java memory allocation values to JAVA_OPTS. For additional settings, refer to section 5.8, “Setting Java JVM Options,” on page 29.
Tomcat 5: common/lib Tomcat 6: lib	Adds MySQL JDBC driver.

Updates to the MySQL database:

If you installed to an existing MySQL database, new schemas and users are created in your database instance:

MySQL Updates	Description
Database <code>jasperserver</code> created	This is the JasperServer repository database. This database holds all of the JasperServer system information such as users, roles, datasources, report definitions, etc.
Database user <code>jasperdb</code> created	The JasperServer application uses this user to connect to the database.
Sample database <code>foodmart</code> created	(optional) Database created if install sample data option was chosen.
Sample database <code>sugarcrm</code> created	(optional) Database created if install sample data option was chosen.

2.2.2 Installer Output Log File Location

The installer creates a log during installation that records information as the installation progresses. If you encounter any problems when you install JasperServer, it can be helpful to look at the installer log for any potential errors. You can find the installer log in the following locations:

Windows: C:\Documents and Settings\\Local Settings\Temp\bitrock_installer_<number>.log

Linux: /tmp/bitrock_installer_<number>.log

2.2.3 Checking your Java JVM Options

For both the bundled Tomcat and the existing Tomcat, the installer attempts to set Java JVM options to help with memory allocation. You can double-check the values set to see that they are appropriate for your installation.

To check your Java JVM settings refer to section [5.8, “Setting Java JVM Options,” on page 29](#).

3 STARTING AND STOPPING JASPERSERVER

This chapter contains the following sections:

- [Using JasperServer Start/Stop Scripts](#)
- [Using Start/Stop Scripts Without Bundled Installation](#)
- [Logging Into JasperServer](#)
- [Starting the Included iReport](#)
- [JasperServer Log Files](#)

3.1 Using JasperServer Start/Stop Scripts

Before JasperServer is started, the database it depends on must be running. Then, JasperServer is started by starting the application server that JasperServer is deployed to. If you chose to install a bundled Tomcat and a bundled MySQL, then the JasperServer start/stop scripts will allow you to start both applications with a single script.

To start or stop JasperServer, do the following:

- From the Windows Start menu:
Click **Start > All Programs > JasperServer 3.7 > JasperServer Management > Start JasperServer**.
Click **Start > All Programs > JasperServer 3.7 > JasperServer Management > Stop JasperServer**.

- From a command line:

```
Windows: cd <js-install>/bin
          allctl.bat [start | stop]
Linux:   cd <js-install>
          ./jasperctl.sh [start | stop]
```

3.2 Using Start/Stop Scripts Without Bundled Installation

If you used your own existing installation for one of either Tomcat or MySQL you can still use the start/stop scripts mentioned in the previous section. The scripts would only start the bundled application that you chose to have the installer install.

For example, if you have an existing Tomcat and installed the bundled MySQL, the scripts and menus specified in the previous section would only start and stop the MySQL application in addition to JasperServer. If you used your existing versions of Tomcat or MySQL or both then you should use the start and stop scripts provided by those applications.

3.3 Logging Into JasperServer

This section assumes that JasperServer is running. If it isn't, start it as described in the section above.

Log into JasperServer by entering the correct URL in your browser's address field and supplying the correct user name and password. JasperServer supports Mozilla Firefox and Internet Explorer. The URL depends upon your application server. For the bundled Tomcat, use:

```
http://<hostname>:8080/jasperserver
```

where:

- <hostname> is the name or IP address of the computer hosting JasperServer
- 8080 is the default port number for the Apache Tomcat application server. If you used a different port when installing your application server, specify its port number when you connect to JasperServer.

In Windows, you can also launch the JasperServer login page from the desktop of its host by clicking **Start > All Programs > JasperServer 3.7 > JasperServer Login**.

If the login page appears, JasperServer has started properly. You may now login as the administrator:

User Id: jasperadmin

Password: jasperadmin

If you installed the sample data then additional sample end-users are created. These end users are non-administrative users who have less system privileges than an administrative user. End-users:

User Id: joeuser (sample end-user)

Password: joeuser

User Id: demo (special SuperMart Dashboard demonstration end-user)

Password: demo



Once you have completed the evaluation or testing of your JasperServer instance, you should change your administrative password and remove the sample end-users.

3.4 Starting the Included iReport

If you chose to install iReport as part of the JasperServer installation, you may start iReport from the Windows Start menu. To do this, click **Start > All Programs > JasperServer 3.7 > Start iReport**.

3.5 JasperServer Log Files

Log files contain important information about how JasperServer is running.

The JasperServer log file location is:

```
<application-server-path>/jasperserver/WEB-INF/logs/jasperserver.log
```

The log4j.properties file location is:

```
<application-server-path>/jasperserver/WEB-INF/log4j.properties
```



By default, JasperServer only logs errors and warnings.

In addition to the JasperServer log, your application server and database server also log information about JasperServer. For information about the logs written by your application server and your database server, refer the associated documentation.

4 UNINSTALLING JASPERSERVER

In Windows, click **Start > All Programs > JasperServer > Uninstall** to uninstall JasperServer.

Alternatively, you may open the Control Panel and choose the **Add or Remove Software** option. Locate JasperServer in the list of installed software and click **Change/Remove**. You are prompted to remove the software. Indicate **Yes** and follow the on-screen instructions.

Under Linux, the <js-install> directory includes an executable that removes JasperServer from the host. From the command line as the root user (or any user with sufficient privileges), enter:

```
cd <js-install>
./uninstall
```

You are prompted whether to remove JasperServer. On your keyboard, press Y and then press Enter to remove JasperServer from this computer.

After running either uninstaller, you are prompted to take an uninstall survey from Jaspersoft. Survey answers are anonymous and help Jaspersoft improve the products we make. When you click **Yes**, the survey launches on a survey provider website in a new browser window. Select all the reasons that led you to uninstall JasperServer, or enter a short explanation if none match. Thank you for your feedback.

5 INSTALLING FROM THE WAR FILE DISTRIBUTION

In addition to the installer binaries, the JasperServer application is distributed as a stand-alone WAR file distribution. This distribution is packaged as a ZIP file. Customers who do not wish to use the installer or who have target configurations other than those supported by the installer should use the WAR file distribution.

This chapter contains the following sections:

- [Applications Supported by the WAR File Distribution](#)
- [Obtaining the WAR File Distribution Zip](#)
- [Unpacking the WAR File Distribution Zip](#)
- [Introduction to Buildomatic Scripts](#)
- [Pre-Installation Steps](#)
- [Configuring the Buildomatic Scripts](#)
- [Installing JasperServer](#)
- [Setting Java JVM Options](#)
- [Setting Java JVM Options](#)
- [Starting JasperServer](#)
- [Logging Into JasperServer](#)
- [Troubleshooting your JasperServer Configuration](#)
- [Running the Import and Export Utilities](#)

5.1 Applications Supported by the WAR File Distribution

The instructions in this chapter support the following target configurations:

Application Server	Database
Tomcat, JBoss, or GlassFish	MySQL
	PostgreSQL

For information on the specific versions of third party applications that are supported by the WAR file distribution ZIP refer to the release notes for the distribution you are using. The release notes are found in the root of the unpacked distribution ZIP.

5.2 Obtaining the WAR File Distribution Zip

The WAR file distribution comes in a file named `jasperserver-ce-3.7-bin.zip` in the compressed ZIP format. To download the WAR file distribution, go to the downloads area of the JasperForge.org web site.

5.3 Unpacking the WAR File Distribution Zip

Once you have downloaded the WAR file distribution, you need to unpack it in order to access the files it contains.

Choose a top level directory location to unpack the ZIP file to. The ZIP file creates the directory `jasperserver-ce-3.7-bin`.

Unpack to a directory such as Program Files or the root of the hard drive in Windows or your home directory in Linux. The resulting location given in the following table will be referred to as `<js-install>` in this document:

Operating System	Example Location	Referenced As
Windows	<code>C:\Program Files\jasperserver-ce-3.7-bin</code>	<code><js-install></code>
Windows	<code>C:\jasperserver-ce-3.7-bin</code>	<code><js-install></code>
Linux	<code>/home/<user>/jasperserver-ce-3.7-bin</code>	<code><js-install></code>

5.4 Introduction to Buildomatic Scripts

The WAR file distribution contains a set of scripts known as the buildomatic scripts. These scripts handle the configuration and deployment of JasperServer.

These scripts are found in the buildomatic directory. They rely on the Apache Ant build tool and the Java JVM for execution.

The procedures in this section describe the steps necessary for installing JasperServer using the buildomatic scripts.

The buildomatic scripts are found at the following location:

`<js-install>/buildomatic`.

5.5 Pre-Installation Steps

5.5.1 Checking Your Java Installation

JasperServer is a Java application that requires either Java 1.5 or Java 1.6. Earlier Java versions such as Java 1.4 will not work with JasperServer. You should check your installed Java version to see that it is at least Java 1.5.

The buildomatic scripts are based on Apache Ant and they required the Java JDK. Therefore, you will need to verify that you have the Java Development Kit (JDK) installed and not merely the Java Runtime Environment (JRE). The JDK has additional tools and utilities required by Apache Ant.

You should also make sure that you have your `JAVA_HOME` variable set.

To check your Java version from the command line run:

```
java -version
```

5.5.2 About Bundled Apache Ant

The War File Distribution ZIP comes with a bundled version of Apache Ant so you do not need to download or install Ant. The buildomatic scripts come with Windows and Linux batch scripts that are pre-configured to use the bundled version of Apache Ant. The buildomatic scripts are called from the command line in the following manner:

```
Windows:  js-ant <target-name>
```

```
Linux:    ./js-ant <target-name>
```

The bundled Apache Ant has an additional jar that extends Ant functionality. This jar is: ant-contrib.jar. This jar enables conditional logic in Ant.



On Linux and Solaris, the js-ant commands may not be compatible with all shells. If you have errors, use the `bash` shell explicitly. For more information, see section [A.12, “Troubleshooting on Solaris,”](#) on page 71.

5.5.3 Checking Your Application Server

To run JasperServer you will need to have an application server installed. The buildomatic scripts support automatic deployment to the Tomcat, JBoss, and GlassFish application servers.

In the configuration step for the buildomatic scripts, you will need to specify the application server you are using and the application server's home directory.

For Tomcat and JBoss:

When running the buildomatic install scripts, Tomcat and JBoss should be stopped.

For Glassfish:

When running the buildomatic install scripts, Glassfish should be running. To start the GlassFish application server, run a command similar to the following:

```
asadmin start-domain domain1
```

5.5.4 Checking Your Database Server

To run JasperServer you will need to have a database available. The buildomatic scripts support automatic installation to the MySQL and PostgreSQL databases.

In the configuration step for the buildomatic scripts, you will need to supply, at a minimum, the DB username, DB password, and DB hostname information for your database.

5.6 Configuring the Buildomatic Scripts

The buildomatic scripts are found at the following location:

```
<js-install>/buildomatic
```

These scripts are pre-configured to handle most of the configuration details for your application server and your database.

5.6.1 Creating your Default Master Properties File

The buildomatic scripts read a file called `default_master.properties` in order to gather your application server path and your database settings. You must create the default master properties file from one of the database-specific sample files provided.

1. Copy the file for your database:

Database	Master Properties File
MySQL	<code><js-install>/buildomatic/sample_conf/inst-mysql_master.properties</code>
PostgreSQL	<code><js-install>/buildomatic/sample_conf/inst-postgresql_master.properties</code>

2. And rename to:

```
<js-install>/buildomatic/default_master.properties
```

3. `cd <js-install>/buildomatic`

4. Edit the `default_master.properties` file to add the settings that are specific to your database and your application server. The table below gives examples for each supported database. Be sure to replace the `appServerDir` property value with the path to your installed application server.

Database	Sample Property Values
MySQL	<pre>appServerType=tomcat6 (or tomcat5, jboss, glassfish) appServerDir=c:\\apache-tomcat-6.0.20 (for example) dbUsername=root dbPassword=password dbHost=localhost</pre>
PostgreSQL	<pre>appServerType=tomcat6 (or tomcat5, jboss, glassfish) appServerDir=c:\\apache-tomcat-6.0.20 (for example) dbUsername=postgres dbPassword=postgres dbHost=localhost</pre>



When the property `appServerType` is set to `skipAppServerCheck`, buildomatic will skip any application server validation (for example, if you have not installed one yet).

5.6.2 Regenerating Your Buildomatic Settings

If you later make a change to your master properties file, you should clean and regenerate your buildomatic script settings by running Ant with the following targets:

```
js-ant clean-config
js-ant gen-config
```

The first target will clear the `buildomatic/build_conf/default` directory. The second will re-build the configuration settings.



After the `clean-config` target has been run, any subsequent target will automatically re-build the configuration settings; the `gen-config` target is simply a convenience.

5.7 Installing JasperServer

Now that your `default_master.properties` file has been edited, you can start the installation steps. Once you execute the first install target, the buildomatic scripts will automatically configure the necessary properties and store these settings in the `<js-install>/buildomatic/build_conf/default` directory.

Execute the following steps at the command line. After executing each Ant target below, you should look for the message `BUILD SUCCESSFUL`. You may omit any steps that create sample data if you do not want to install the sample reports:

Commands	Description
<code>cd <js-install>/buildomatic</code>	
<code>js-ant create-js-db</code> <code>js-ant create-sugarcrm-db</code> <code>js-ant create-foodmart-db</code>	Creates the JasperServer and sample databases. In case of errors, see 5.7.2, “Creating Databases Manually,” on page 29.
<code>js-ant load-sugarcrm-db</code> <code>js-ant load-foodmart-db</code> <code>js-ant update-foodmart-db</code>	(Optional) Loads sample data into the sample databases.
<code>js-ant init-js-db-ce</code> <code>js-ant import-minimal-ce</code>	Initializes and loads the default organization and users.
<code>js-ant import-sample-data-ce</code>	(Optional) Loads the demos that use the sample data.
<code>js-ant deploy-webapp-ce</code>	Configures and deploys the WAR file to Tomcat or JBoss. For GlassFish, see Deploying to GlassFish below.



On Linux and Solaris, the `js-ant` commands may not be compatible with all shells. If you have errors, use the `bash` shell explicitly. For more information, see section [A.12, “Troubleshooting on Solaris,”](#) on page 71.

5.7.1 Deploying to GlassFish

The GlassFish application server must first be running as described in section [5.5.3, “Checking Your Application Server,”](#) on page 27. Check that GlassFish is running and then run:

```
js-ant deploy-webapp-ce
```

When deploying to GlassFish with the previous command, the ant target will perform the following actions:

- JVM options will be set as described in section [5.8, “Setting Java JVM Options,”](#) on page 29.
- The GlassFish application server will be stopped. You will restart the server in a subsequent step.

5.7.2 Creating Databases Manually

If you encounter an error when running one of the create db targets (`create-sugarcrm-db`, `create-foodmart-db`, or `create-js-db`) you may create the JasperServer database manually using the database administration tool for your particular database type. After creating the database, you can continue with the remaining `buildomatic` steps. Additionally, keep in mind that the `sugarcrm` and `foodmart` databases are optional, sample databases.

For procedures on MySQL, see section [6.3, “Additional Notes on the MySQL Database,”](#) on page 37.

5.8 Setting Java JVM Options

JasperServer requires that your Java JVM runtime options be set to values larger than the typical Java default values.

For Tomcat and JBoss these values can be directly set by editing the shell scripts which start the application server. See section [6.1.1, “Tomcat and JBoss JVM Options,”](#) on page 33 for detailed steps.

For GlassFish, you can set the JVM options using the `asadmin` utility or by editing the `domain.xml` config directly. See section [6.1.3, “GlassFish JVM Options,”](#) on page 35 for detailed steps.

5.9 Starting JasperServer

To run JasperServer start your application server with one of the following commands:

Tomcat: `<tomcat>/bin/startup.bat` (Windows) *or* `<tomcat>/bin/startup.sh` (Linux)

JBoss: `<jboss>/bin/run.bat` (Windows) *or* `<jboss>/bin/run.sh` (Linux)

GlassFish: `asadmin start-domain domain1`



To see the JasperServer application logs, refer to section 6.9, “Notes on JasperServer Logging,” on page 40.

5.10 Logging Into JasperServer

If JasperServer started up cleanly you should be able to login at the following URL

`http://<hostname>:8080/jasperserver`

For example:

`http://localhost:8080/jasperserver`

`http://jasperserver.example.com:8080/jasperserver`

The login page appears after taking some time to compile the necessary JSP file. You may now login as the administrator:

User Id: `jasperadmin`

Password: `jasperadmin`

If you logged in successfully, your JasperServer home page appears.

Refer to the *JasperServer CE User Guide* to begin adding reports and other objects to JasperServer.



Jaspersoft recommends that you change your `jasperadmin` password from the default values to new, secure values.

5.10.1 JasperServer Heartbeat

Upon first login to a newly installed JasperServer, you will be asked whether to opt-in to the JasperServer Heartbeat or not.

To opt-in, click **OK**. To opt-out, click the check box to remove the check and click **OK**.

The JasperServer heartbeat helps Jaspersoft create better products by improving our understanding of customer installation environments. If you choose to enable the heartbeat, at server startup time information like the following will be sent to Jaspersoft via an HTTPS call:

- Operating System type and version
- JVM type and version
- Application Server type and version
- Database type and version
- JasperServer type and version
- Unique, anonymous identifier value

You can also manually enable or disable the heartbeat by modifying the `jasperserver/WEB-INF/applicationContext-logging.xml` file. To disable, set the `enabled` property to `false` like below:

```
<property name="enabled" value="false"/>
```

5.11 Troubleshooting your JasperServer Configuration

5.11.1 JasperServer Startup Problems

When trying to run a new JasperServer instance, the most typical issue that users encounter are problems with the database configuration. These problems are typically related to having incorrect configurations within the JasperServer database configuration files or in the application server configuration files.

For more information on resolving these types of errors, refer to troubleshooting section [A.2, “Database Connectivity Errors,”](#) on page 66.

5.11.2 Error Running a Report

If you have trouble running reports in your new JasperServer Instance, refer to troubleshooting section [A.3, “Error Running a Report,”](#) on page 67.

5.12 Running the Import and Export Utilities

The buildomatic scripts automatically configure the database information needed by the import and export utilities. These utilities are invoked as ant targets located in the following directory:

```
cd <js-install>/buildomatic
```

This section describes the Ant targets and parameter setting you need to specify in order to send the standard options to the import and export commands.

5.12.1 Running Export from Buildomatic

The `export-ce` target for ant has the following syntax:

```
Windows: js-ant export-ce -DexportFile=<filename> -DexportArgs="<export-options>"
```

```
Linux:    ./js-ant export-ce -DexportFile=<filename> -DexportArgs="\<export-options>"
```

The export file format can be a ZIP file or it can be a set of files under a new directory name. If you specify the `.zip` extension for your output filename then a ZIP archive will automatically be created. Otherwise, a directory with files and sub-directories will be created as a non-compressed set of files. The `exportArgs` argument requires double quotes (") and can contain more than one export option, as shown in the following Windows example.

```
js-ant export-help-ce

js-ant export-ce -DexportFile=my-reports.zip
-DexportArgs="--uris /reports"

js-ant export-ce -DexportFile=my-reports-and-users.zip
-DexportArgs="--uris /reports
--users jasperadmin,joeuser"

js-ant export-ce -DexportFile=my-datasources
-DexportArgs="--uris /datasources --roles ROLE_USER"

js-ant export-ce -DexportFile=js-everything.zip -DexportArgs="--everything"
```

On Linux, all double quotes (") must be escaped with a backslash (\). In addition, when giving a list of user ids, it must be enclosed in single quotes ('), as shown in the Linux example below:

```
./js-ant export-help-ce

./js-ant export-ce -DexportFile=my-reports-and-users.zip
-DexportArgs="--uris /reports
--users 'jasperadmin,joeuser'\"
```

5.12.2 Running Import from Buildomatic

The `import-ce` target for ant has the following syntax:

Windows: `js-ant import-ce -DimportFile=<filename> [-DimportArgs="<import-options>"]`

Linux: `./js-ant import-ce -DimportFile=<filename> [-DimportArgs="\<import-options>\"]`

The imported file is handled as a ZIP archive if its name ends in `.zip`, otherwise it will be handled as a directory. The `importArgs` argument is optional, it can contain more than one import option.

The following examples on Windows are typical import commands:

```
js-ant import-help-ce

js-ant import-ce -DimportFile=my-reports.zip

js-ant import-ce -DimportFile=my-datasources -DimportArgs="--update"
```

On Linux, all double quotes (") must be escaped with a backslash (\). The following examples on Linux are typical import commands:

```
./js-ant import-help-ce

./js-ant import-ce -DimportFile=my-reports.zip

./js-ant import-ce -DimportFile=my-datasources.zip -DimportArgs="--update\"
```

5.12.3 Running Import or Export from the Scripts Directory

The Import-export utility can also be run from the `<js-install>/scripts` directory. If you want to run this utility from the `<js-install>/scripts` directory you may need to manually set up the configuration to point to your database.

For information on manually configuring and running the import-export utility from the scripts directory, see chapter 11, [“Configuring the Import-Export Utilities,” on page 61](#).

6 ADDITIONAL INSTALLATION INFORMATION

This chapter contains the following sections:

- **Setting JVM Options for Application Servers**
- **Additional Buildomatic Configuration Information**
- **Additional Buildomatic Configuration Information**
- **Additional Notes on the MySQL Database**
- **Notes on the Hibernate Properties File**
- **Installed JDBC Driver Locations**
- **Notes on Database Connection for Tomcat**
- **Notes on Datasource Definition for JBoss**
- **Notes on Database Connection for GlassFish**
- **Notes on JasperServer Logging**
- **Report Scheduling Configuration with Quartz**
- **Notes on Updating XML/A Connection Definitions**

6.1 Setting JVM Options for Application Servers

JasperServer runs better with certain Java options for the JVM in which its application server is running. The options you need and how you set them depends on your version of Java, your application server, and how it is deployed.

The settings in this section apply specifically to the Sun JVM. Other JVMs may or may not have equivalent settings.

6.1.1 Tomcat and JBoss JVM Options

JasperServer is supported on Java 1.5 and 1.6. If you are using Java 1.6, there are some additional JVM settings to avoid conflicts with JasperServer's AXIS-based web service classes. These conflicts could cause JasperServer web services and the resources which rely on them to fail (such as analysis XML/A connections). Similarly, JBoss 4.2 includes a web service that conflicts with JasperServer's AXIS-based web service classes and requires the same additional settings.

JVM Options on Windows	
Tomcat file	<tomcat>/bin/setclasspath.bat or <tomcat>/bin/setenv.bat
JBoss file	<jboss>/bin/run.bat
Options for Java 1.5 and Java 1.6	set JAVA_OPTS=%JAVA_OPTS% -Xms128m -Xmx512m -XX:PermSize=32m -XX:MaxPermSize=128m -Xss2m -XX:+UseConcMarkSweepGC -XX:+CMSClassUnloadingEnabled -XX:+CMSPermGenSweepingEnabled
Additional options for Java 1.6 or JBoss 4.2	set JAVA_OPTS=%JAVA_OPTS% -Djavax.xml.soap.MessageFactory=org.apache.axis.soap.MessageFactoryImpl -Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis.soap.SOAPConnectionFactoryImpl -Djavax.xml.soap.SOAPFactory=org.apache.axis.soap.SOAPFactoryImpl -Djavax.xml.transform.TransformerFactory=org.apache.xalan.processor.TransformerFactoryImpl

JVM Options on Linux	
Tomcat file	<tomcat>/bin/setclasspath.sh or <tomcat>/bin/setenv.sh
JBoss file	<jboss>/bin/run.sh
Options for Java 1.5 and Java 1.6	export JAVA_OPTS="\$JAVA_OPTS -Xms128m -Xmx512m -XX:PermSize=32m -XX:MaxPermSize=128m -Xss2m -XX:+UseConcMarkSweepGC -XX:+CMSClassUnloadingEnabled -XX:+CMSPermGenSweepingEnabled"
Additional options for Java 1.6 or JBoss 4.2	export JAVA_OPTS="\$JAVA_OPTS -Djavax.xml.soap.MessageFactory=org.apache.axis.soap.MessageFactoryImpl -Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis.soap.SOAPConnectionFactoryImpl -Djavax.xml.soap.SOAPFactory=org.apache.axis.soap.SOAPFactoryImpl -Djavax.xml.transform.TransformerFactory=org.apache.xalan.processor.TransformerFactoryImpl"

Add your JAVA_OPTS setting directly below the following lines:

Add JVM Options Here	
<tomcat>/bin/setclasspath.bat	set JAVA_ENDORSED_DIRS=%BASEDIR%\common\endorsed
<tomcat>/bin/setclasspath.sh	JAVA_ENDORSED_DIRS=" \$BASEDIR " /common/endorsed
<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"
<tomcat>/bin/setenv.bat or <tomcat>/bin/setenv.sh	JAVA_OPTS setting can go anywhere in this user created file.

6.1.2 Tomcat as a Windows Service JVM Options

If Tomcat is running as a Windows service, then you would typically add the Java options for JasperServer to the Java Tab of the Tomcat Properties dialog:

1. Launch the Tomcat configuration application from the Windows Start menu:
Start > Programs > Apache Tomcat > Configure Tomcat
2. In the Apache Tomcat Properties dialog, click the **Java** tab.
3. In the Java Options field, add your JasperServer JAVA_OPTS values according to the table above.
Enter only the options preceded by -X or -D, not set JAVA_OPTS=%JAVA_OPTS%.
4. Click Apply, then click OK.

6.1.3 GlassFish JVM Options

For GlassFish, the JVM settings are identical for Java 1.5 and Java 1.6. The following sections show how to set the JVM options for GlassFish either through the command line or in a configuration file.

6.1.3.1 Setting GlassFish JVM Options with asadmin Command

First make sure your GlassFish instance is up and running, then run the following command (enter as a single line):

```
asadmin create-jvm-options -Xms128m:-Xmx512m:-XX\:PermSize=32m:
-XX\:MaxPermSize=128m:-Xss2m:-XX\:+UseConcMarkSweepGC:
-XX\:+CMSClassUnloadingEnabled:-XX\:+CMSPermGenSweepingEnabled:
-Djavax.xml.soap.MessageFactory=org.apache.axis.soap.MessageFactoryImpl:
-Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis.soap.SOAPConnectionFactoryImpl:
-Djavax.xml.soap.SOAPFactory=org.apache.axis.soap.SOAPFactoryImpl
```

Now, restart the application server with the following commands:

```
asadmin stop-domain domain1
asadmin start-domain domain1
```

When running the `asadmin create-jvm-options` command above, you may see some error messages such as the following:

```
[exec] CLI167 Could not create the following jvm options. Options exist:
[exec] -Xmx512m
[exec] CLI137 Command create-jvm-options failed.
```

This message indicates that one of the options specified was already set in the JVM. The command will succeed for all other JVM options on the command line. No further action is necessary.

6.1.3.2 Setting GlassFish JVM Options by Editing domain.xml

Open the `<glassfish>/domains/domain1/config/domain.xml` configuration file for editing, and add the following lines to the section entitled `java-config`:

```
<jvm-options>-Xms128m -Xmx512m -XX:PermSize=32m -XX:MaxPermSize=128m -Xss2
-XX:+UseConcMarkSweepGC -XX:+CMSClassUnloadingEnabled
-Djavax.xml.soap.MessageFactory=org.apache.axis.soap.MessageFactoryImpl
-Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis.soap.SOAPConnectionFactoryImpl
-Djavax.xml.soap.SOAPFactory=org.apache.axis.soap.SOAPFactoryImpl
</jvm-options>
```

If you are modifying the settings for a running instance of GlassFish, you must restart the application server with the following commands:

```
asadmin stop-domain domain1
asadmin start-domain domain1
```

6.2 Additional Buildomatic Configuration Information

The JasperServer Ant based buildomatic scripts contain support files that allow for the setup and configuration of a number of databases and application servers. Here are some pointers to the locations and content of some of these files.

6.2.1 Buildomatic: Generated Property Files

After you set your database and application server property values, you initiate buildomatic which automatically generates the database and application server configuration files needed to prepare for a JasperServer installation.

You will find the generated property files in the following location:

```
<js-install>/buildomatic/build_conf/default
```

Here are some of the key configuration files:

```
js.jdbc.properties
js.quartz.properties
js-glassfish-ds.xml
js-jboss-ds.xml
maven_settings.xml - (used for source code build)
```

More generated property files:

```
<js-install>/buildomatic/build_conf/default/webapp
```

In this directory you will find config files such as:

```
META-INF/context.xml
WEB-INF/hibernate.properties
WEB-INF/js.quartz.properties
```

The autogenerated files above are removed if you run the buildomatic target: `clean-config`. You can then regenerate them by running the target: `gen-config`. (Also, after running `clean-config`, any subsequent target will regenerate the configuration files.)

6.2.2 Buildomatic: SQL Scripts Location

Buildomatic comes with SQL scripts and other utilities that support a number of databases. Here is where these files are found:

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

Here is an example of some of the key files (same pattern for additional databases):

```
<js-install>/buildomatic/install_resources/sql/mysql/js-create.ddl
<js-install>/buildomatic/install_resources/sql/mysql/js-drop.ddl
<js-install>/buildomatic/install_resources/sql/mysql/quartz.ddl
<js-install>/buildomatic/install_resources/sql/mysql/sugarcrm.zip
```



You can run these scripts manually by copying them to a location where your database client software is located.

6.2.3 Buildomatic: Database Creation Statements Location

For most databases the buildomatic scripts are able to create the metadata repository database used by JasperServer. This is the database where the data defining users, roles, data sources, reports, analysis views, domains, etc are stored. This database is normally named `jasperserver`.

Buildomatic attempts to create the `jasperserver` database via JDBC when the `create-js-db` target is executed.

The scripts and property files used to create the `jasperserver` database are here:

```
<js-install>/buildomatic/conf_source/db/
```

Property script locations:

```
mysql/scripts.properties
postgresql/scripts.properties
```

6.2.4 Buildomatic: JDBC Driver Locations

Buildomatic has default JDBC drivers for each supported database. These JDBC drivers are located here:

MySQL: `<js-install>/buildomatic/conf_source/db/mysql/jdbc/mysql-connector-java-5.1.9.jar`
 PostgreSQL `<js-install>/buildomatic/conf_source/db/postgresql/jdbc/postgresql-8.1-407.jdbc3.jar`

The buildomatic scripts will automatically copy the appropriate JDBC driver to your application server when you run the `deploy-webapp-ce` target. Here are some typical locations where you can expect the JDBC driver to be copied:

Tomcat 5: `<tomcat>/common/lib`
 Tomcat 6: `<tomcat>/lib`
 JBoss: `<jboss>/server/default/lib`
 GlassFish: `<glassfish>/domains/domain1/lib/ext`

6.2.5 Buildomatic: JasperServer WAR File Location

Buildomatic takes the JasperServer WAR file from the root of the `<js-install>` directory:

`<js-install>/jasperserver.war`

When you run the `deploy-webapp-ce` target, buildomatic takes the war archive and unpacks it into your application server. Next, the database configuration files needed by the application server are copied to the appropriate locations. For instance, in the case of Tomcat:

`<js-install>/jasperserver.war -> unpacked and copied to <tomcat>/webapps`
`<js-install>/buildomatic/build_conf/default/webapp/META-INF/context.xml -> copied to
 <tomcat>/webapp/jasperserver/META-INF`
`<js-install>/buildomatic/build_conf/default/webapp/WEB-INF/hibernate.properties -> copied to
 <tomcat>/webapp/jasperserver/WEB-INF/hibernate.properties`
`<js-install>/buildomatic/build_conf/default/webapp/WEB-INF/js.quartz.properties -> copied to
 <tomcat>/webapp/jasperserver/WEB-INF/js.quartz.properties`
`<js-install>/buildomatic/build_conf/db/mysql/jdbc/mysql-connector-java-5.1.10.jar -> copied to
 <tomcat>/lib`

6.2.6 Buildomatic: Sample Data Catalog ZIP Files

Buildomatic includes export files which hold the JasperServer sample data (that have examples of new features). This sample data is loaded when you run the buildomatic target `import-sample-data-ce`, for instance. These export files along with other important export files are located here:

`<js-install>/buildomatic/install_resources/export/`

Here are some key files (same pattern for additional databases):

`js-catalog-minimal-ce.zip`
`js-catalog-mysql-ce.zip`
`js-catalog-postgresql-ce.zip`

6.3 Additional Notes on the MySQL Database

This section provides commands to:

- Manually create and initialize the JasperServer database.
- Manually import the default users and organization.
- Manually create and load the sample databases.

The MySQL client software, `mysql.exe` or `mysql`, can be used to interact with the MySQL database. The example commands below have been tested at Jaspersoft. The commands to be used on your MySQL instance may be different.

These commands are run from the Windows or Linux command line.

6.3.1 Manual Creation of the JasperServer Database

Please check your database user documentation for how to set up a database and how to create a database user.

Run the following commands:

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

mysql -u root -p
mysql>create database jasperserver character set utf8;
mysql>grant all on *.* to jasperdb@localhost identified by 'password';
mysql>flush privileges; (reload privilege tables)
mysql>use jasperserver;
mysql>source js-create.ddl
mysql>source quartz.ddl
mysql>exit
```



If you are going to access MySQL on a remote server you should run an additional grant statement:

```
mysql>grant all on *.* to jasperdb@'%' identified by 'password';
```

6.3.2 Manual Import of Default Users

First, follow the instructions in chapter [11, “Configuring the Import-Export Utilities,” on page 61](#).

Then run the following commands to perform the import:

```
cd <js-install>/scripts
js-import --input-zip ../buildomatic/install_resources/export/js-catalog-minimal-ce.zip
```

The next command loads the demos that use the sample data. Do not perform the next command if you do not create and load the sample databases in the next section.

```
js-import --input-zip ../buildomatic/install_resources/export/js-catalog-mysql-ce.zip
```

6.3.3 Manual Creation of Sample Databases

Run the following commands:

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

mysql -u root -p
mysql>create database sugarcrm;
mysql>create database foodmart;
mysql>use sugarcrm;
mysql>source sugarcrm-mysql.sql;
mysql>use foodmart;
mysql>source foodmart-mysql.sql; (first make sure the file is unzipped)
mysql>source supermart-update.sql;
mysql>exit
```

6.4 Notes on the Hibernate Properties File

Your hibernate.properties settings will be found in the following directory after buildomatic has been run to automatically generate your configuration files:

```
<js-install>/buildomatic/build_conf/default/webapp/WEB-INF/hibernate.properties
```

Within the jasperserver WAR file the hibernate.properties file is found at the following location:

```
<appserver-path>/jasperserver/WEB-INF/hibernate.properties
```

The buildomatic scripts automatically create this configuration file. When you run the buildomatic target `deploy-webapp-ce` this file is copied to JasperServer in your application server.

Here are some example hibernate property values.

```
MySQL: metadata.hibernate.dialect=org.hibernate.dialect.MySQLDialect
```

6.5 Installed JDBC Driver Locations

If you have installed JasperServer using the installer, the buildomatic installation scripts are not included. However, all of the JDBC drivers that are included in buildomatic are also included in the `<js-install>/scripts` directory:

```
MySQL: <js-install>/scripts/drivers/mysql/jdbc/mysql-connector-java-5.1.9.jar
```

```
PostgreSQL postgresql/jdbc/postgresql-8.1-407.jdbc3.jar
```

6.6 Notes on Database Connection for Tomcat

After setting up the buildomatic configuration for your database, the Tomcat context.xml will be automatically created with the appropriate settings for JasperServer.

When the buildomatic target `deploy-webapp-ce` is run, the context.xml will be automatically copied into the jasperserver WAR set of files.

You can view the automatically generated context.xml at the following location:

```
<js-install>/buildomatic/build_conf/default/webapp/META-INF/context.xml
```

The final location of the context.xml is:

```
<tomcat>/webapps/jasperserver/META-INF/context.xml
```

6.7 Notes on Datasource Definition for JBoss

After setting up the buildomatic configuration for your database, the JBoss datasource definition file will be automatically created with the appropriate settings for JasperServer.

When the buildomatic target `deploy-webapp-ce` is run, the js-jboss-ds.xml will be automatically copied into the JBoss instance.

You can view the automatically generated js-jboss-ds.xml at the following location:

```
<js-install>/buildomatic/build_conf/default/js-jboss-ds.xml
```

The final location of the js-jboss-ds.xml is:

```
<jboss>/server/default/deploy/js-jboss-ds.xml
```

6.7.1 Notes on Extra JBoss Configuration Step

Note: When JasperServer is running under JBoss, there are a couple of INFO log messages and an XML/A connection error that might occur depending on the version of JBoss you are running with.

For more information, refer to troubleshooting section [A.9, “JBoss Modifications,” on page 69](#).

6.8 Notes on Database Connection for GlassFish

After setting up the buildomatic configuration for your database, the GlassFish datasource definition file will be automatically created with the appropriate settings for JasperServer.

When the buildomatic target `deploy-webapp-ce` is run, the `js-glassfish-ds.xml` will be automatically deployed to the GlassFish instance.

You can view the automatically generated `js-glassfish-ds.xml` at the following location:

```
<js-install>/buildomatic/build_conf/default/js-glassfish-ds.xml
```

To deploy the datasource definition manually, you can run a command similar to the following:

```
asadmin add-resources "<js-install>/buildomatic/build_conf/default/js-glassfish-ds.xml"
```

6.9 Notes on JasperServer Logging

The JasperServer log output goes to the following locations:

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

JBoss: `<jboss>/server/default/deploy/jasperserver.war/WEB-INF/logs/jasperserver.log`

GlassFish: `<glassfish>/domains/domain1/autodeploy/jasperserver.war/WEB-INF/logs/jasperserver.log`

You can change the logging level for the overall application or for particular classes by modifying the following property file:

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

JBoss: `<jboss>/server/default/deploy/jasperserver.war/WEB-INF/log4j.properties`

GlassFish: `<glassfish>/domains/domain1/autodeploy/jasperserver.war/WEB-INF/log4j.properties`

6.10 Report Scheduling Configuration with Quartz

The JasperServer report scheduling feature is powered by the Quartz scheduler tool. The configuration settings for Quartz based report scheduling is automatically handled by buildomatic.

This section discusses the various settings contained in the `js.quartz.properties` file. In addition, there is a mention of the `applicationContext-report-scheduling.xml` file.

In a deployed JasperServer instance, you will find the `js.quartz.properties` file in the following location:

```
<app-server-path>/jasperserver/WEB-INF/js.quartz.properties
```

For mail server configuration, there is an additional property setting for authentication in the following file:

```
<app-server-path>/webapps/jasperserver/WEB-INF/applicationContext-report-scheduling.xml
```

There are four main configurations to be discussed in this section:

- Mail Server Configuration
- Quartz Driver Delegate Class
- Report Scheduler Web URI
- Quartz Table Prefix

6.10.1 Mail Server Configuration Settings

If you schedule reports or run them in the background, you can specify email addresses to notify when the report completes. In order to use this feature, you must configure JasperServer to contact an email server. To configure these values, you must edit the following configuration files.

Mail Server Configuration Settings		
Configuration File		
...\WEB-INF\js.quartz.properties		
Property	Description	
report.scheduler.mail.sender.host	The name of the computer hosting the mail server.	
report.scheduler.mail.sender.username	The name of the user in the mail server that JasperServer can use.	
report.scheduler.mail.sender.password	The password of the mail server user.	
report.scheduler.mail.sender.from	The address that appears in the From field on email notifications.	
report.scheduler.mail.sender.protocol	The protocol that the mail server uses. JasperServer only supports SMTP. Note: Your entry must be lower case. For example: <code>smtp</code>	
report.scheduler.mail.sender.port	The port number that the mail server uses. For SMTP, the default is typically 25 (values other than 25 may not work in earlier JasperServer versions).	
Configuration File		
...\WEB-INF\applicationContext-report-scheduling.xml		
Property	Bean	Description
javaMailProperties key="mail.smtp.auth"	reportScheduler MailSender	If your mail server requires authentication, change this property from false to true.

6.10.2 Database settings for the Quartz Driver Delegate Class

The Quartz driver delegate class is a class which Quartz uses to interact with the JDBC driver. For the PostgreSQL database it needs a non-default setting.

Note: If you installed using buildomatic these settings are automatically handled.

To manually edit, you would edit the following file:

```
<app-server-path>/jasperserver/WEB-INF/js.quartz.properties
```

Here are settings for supported databases:

Database	Delegate Class Setting
MySQL	quartz.delegateClass=org.quartz.impl.jdbcjobstore.StdJDBCDelegate
PostgreSQL	quartz.delegateClass=org.quartz.impl.jdbcjobstore.PostgreSQLDelegate

6.10.3 Settings for the Report Scheduler Web URI

For the web URI setting, the exact settings depend on what port your application server is running on and the name of your deployed jasperserver web application (i.e. if you do not use the default name `jasperserver`).

Note: If you installed using buildomatic these settings are automatically handled.

To manually edit, you would edit the following file:

<app-server-path>/jasperserver/WEB-INF/js.quartz.properties

Here are some example settings for several application servers:

Application Server	Example Report Scheduler Web URI Setting
Apache Tomcat	report.scheduler.web.deployment.uri=http://localhost:8080/jasperserver
JBoss	report.scheduler.web.deployment.uri=http://localhost:8080/jasperserver
GlassFish	report.scheduler.web.deployment.uri=http://localhost:8080/jasperserver

6.10.4 Settings for the Quartz Table Prefix

If your database is running with a non-default schema, you will need set a Quartz table prefix. This could be the case for Oracle, SQL Server, or DB2.

Note: If you installed using buildomatic these settings are automatically handled.

To manually edit, you would edit the following file:

<app-server-path>/jasperserver/WEB-INF/js.quartz.properties

Here is an example setting for DB2:

quartz.tablePrefix=JSRPSRVR.QRTZ_

6.10.5 Settings for Import-Export

If you are manually configuring the import-export utility (i.e. not using the buildomatic scripts), then depending on the database you are using you will need to make sure your settings are correct for the Quartz driver delegate class property and the Quartz table prefix property.

Note: If you installed using buildomatic these settings are automatically handled (in buildomatic import-export).

For manual import-export config, check following properties file:

<js-install>/scripts/config/js.quartz.properties

Make sure the following properties are correctly set (based on setting described above):

quartz.delegateClass=<your-setting>

quartz.tablePrefix=<your-setting>

6.11 Notes on Updating XML/A Connection Definitions

Sample XML/A connections are included with the JasperServer sample data. If you plan to use XML/A Web Services in your environment, then you may want to check and possibly update the hard coded values in the sample connections.

JasperServer is able to make XML/A connections over the Web Services interface. These HTTP-based connections use a JasperServer user account for authentication. You may have different usernames and passwords than the defaults that get loaded from the sample data load in the sections above. Additionally, your application server hostnames and port values might be different than the default values.

There are two sample analysis views that use this connection:

- Foodmart Sample XMLA Analysis View
- SugarCRM Sample XMLA Analysis View

If you would like to validate and update these resources, do the following:

1. Log into JasperServer as an Administrative user (such as jasperadmin).

2. Navigate to the Repository Management page by selecting the **View > Repository** menu item.
3. Click to expand the **Analysis Components** folder, then the **Analysis Connections** folder. Click to highlight the **Foodmart XMLA Connection** resource, and then click the **Edit** icon.
4. Edit the following information on this screen:
 - ♦ URI (hostname and port)
 - ♦ Login Username
 - ♦ Login Password
5. Click **Next**, then **Save**.
6. Make the same updates for the **SugarCRM XMLA Connection** resource.

7 UPGRADE FROM JASPERSERVER 3.5 TO 3.7

This chapter describes the steps necessary to carry out an upgrade from JasperServer 3.5 to JasperServer 3.7. These steps will use the JasperServer WAR File Distribution ZIP release package and the included buildomatic scripts for the upgrade procedure.

For some JasperServer releases it is possible to upgrade the JasperServer database using SQL upgrade scripts (if the underlying object model is not significantly changed). For release 3.7, this is the case. If you would like to use the SQL upgrade scripts, this procedure is detailed in chapter 8, “[Upgrade Using SQL Upgrade Script](#),” on page 51.

This chapter contains the following sections:

- [Standard Upgrade Steps](#)
- [Backing Up Your JasperServer 3.5 Instance](#)
- [Exporting Your 3.5 Repository Data](#)
- [Preparing the JasperServer 3.7 WAR File Distribution](#)
- [Configuring Buildomatic for Your Database and Application Server](#)
- [Upgrading to JasperServer 3.7](#)
- [Starting JasperServer 3.7](#)
- [Logging Into JasperServer 3.7](#)
- [Additional Notes on JasperServer Upgrade](#)

7.1 Standard Upgrade Steps

This section lists the standard upgrade steps. These general steps are guaranteed to work with each new JasperServer release.

1. Back up your 3.5 JasperServer instance
2. Export your 3.5 repository data
3. Upgrade your instance to 3.7
4. Import your 3.5 repository data

If your instance of JasperServer 3.5 has any custom modifications or extensions, you will need to keep track of these and re-integrate them into your 3.7 instance after the upgrade is complete.

7.2 Backing Up Your JasperServer 3.5 Instance

First you must backup your JasperServer WAR file and your jasperserver database so that they can be restored in case there is a problem with the upgrade. These steps are performed from the command line in a Windows or Linux shell.

The following instructions are for the MySQL database. For other databases, consult your DB administration documentation for back up information.

1. Back up the jasperserver directory in Tomcat to a backup directory:
 - a. `cd <tomcat>`
 - b. `mkdir js-3.5-war-backup`
 - c. `copy <tomcat>/webapps/jasperserver to <tomcat>/js-3.5-war-backup`
 - d. delete the `<tomcat>/webapps/jasperserver` directory
2. Back up the jasperserver database. Go to the location where you originally unpacked the 3.5 WAR file distribution zip:
 - a. `cd <js-install-3.5>` (the location of your original unpacked 3.5 WAR file distribution)
 - b. Run the following command:

Windows: `mysqldump --user=root --password=<password> jasperserver > js-db-3.5-dump.sql`

Linux: `mysqldump --user=root --password=<password> --host=127.0.0.1 jasperserver > js-db-3.5-dump.sql`



If you installed the previous release from the installer, you may specify `--user=jasperdb` in this command. If you receive an error about packet size, see section [A.2.5, “Maximum Packet Size in MySQL,” on page 67](#). Jaspersoft has tested the `mysqldump` utility for backing up and restoring MySQL databases, but there are other MySQL backup mechanisms, some of which may work better for your JasperServer installation.

7.3 Exporting Your 3.5 Repository Data

You will use the JasperServer `js-export` utility to export your 3.5 repository data. You will need to verify that the import-export utility is properly configured in your 3.5 system. If your import-export utility is not already configured, see chapter [11, “Configuring the Import-Export Utilities,” on page 61](#).

7.3.1 Exporting With Buildomatic Scripts

If you used the buildomatic scripts to install JasperServer 3.5, then everything should already be configured for you to export your 3.5 data. Do the following steps:

1. `cd <js-install-3.5>/buildomatic`
2. Run the export:

Windows: `js-ant.bat export-everything-ce -DexportFile=js-3.5-export.zip`

Linux: `./js-ant export-everything-ce -DexportFile=js-3.5-export.zip`

The export uses the `--everything` option by default.

You will later specify the path to this file when you import into 3.7.

7.3.2 Exporting With js-export Script

Go to the JasperServer 3.5 installation location and change to the scripts directory. This is where you will run the `js-export` command. You will note the location of this file so you can point to it when you upgrade to 3.7. Do the following steps:

1. `cd <js-install-3.5>/scripts`

2. Run the export script:

Windows: `js-export.bat --everything --output-zip js-3.5-export.zip`

Linux: `js-export.sh --everything --output-zip js-3.5-export.zip`

You will later specify the path to this file when you import into 3.7.

7.4 Preparing the JasperServer 3.7 WAR File Distribution

We will use the buildomatic scripts included in the 3.7 WAR file distribution ZIP release package in order to carry out the upgrade. Follow the steps in the sections listed below to obtain and unpack the WAR file distribution ZIP file:

1. Follow steps in section [5.2, “Obtaining the WAR File Distribution Zip,” on page 26.](#)
2. Follow steps in section [5.3, “Unpacking the WAR File Distribution Zip,” on page 26.](#)

7.5 Configuring Buildomatic for Your Database and Application Server

This upgrade procedure is based on using the “buildomatic” scripts which are included with the WAR File Distribution ZIP release package.

Follow the configuration steps that match your database and application server in section [5.6, “Configuring the Buildomatic Scripts,” on page 27.](#) Once your buildomatic scripts are configured, proceed to the next section.

7.6 Upgrading to JasperServer 3.7

Now that your buildomatic scripts have been configured, you can complete the upgrade.

Run the following commands:



Make sure you have backed up your jasperserver database before proceeding.

Commands	Description
<code>cd <js-install-3.7>/buildomatic</code>	
<code>js-ant drop-js-db</code> <code>js-ant create-js-db</code> <code>js-ant init-js-db-ce</code>	This will delete your jasperserver db. Make sure it is backed up.
<code>js-ant import-upgrade-ce</code> <code>-DimportFile="<path-and-filename>"</code>	The <code>-DimportFile</code> should point to the <code><path></code> and <code><filename></code> of the <code>js-3.5-export.zip</code> file you created earlier. On Windows, you must use double quotes (") if your path or filename contains spaces. On Linux, you must use double quotes escaped with a backslash (\") in this case.
<code>js-ant import-sample-data-upgrade-ce</code>	(Optional) This step is optional. Loads the 3.7 sample data.
<code>js-ant deploy-webapp-ce</code>	Delete existing 3.5 war file, deploy 3.7 war file.



On MySQL, if you receive an error about packet size, see [A.2.5, “Maximum Packet Size in MySQL,” on page 67.](#)

7.7 Starting JasperServer 3.7

You may now start your Tomcat, JBoss, or GlassFish application server. Your database should already be running.

7.8 Logging Into JasperServer 3.7

If your application server and JasperServer 3.7 were started cleanly, you can now prepare to login.

7.8.1 Clearing Your Browser Cache

Before you log into 3.7, make sure and clear your Browser cache. Javascript files, which enable UI elements of JasperServer, are typically cached by the Browser. The cache should be cleared to ensure that the most current files are used.

Your end users should also be reminded to clear their Browser caches before logging in.

7.8.2 Logging Into JasperServer

Login using the following URL and administrator credentials:

`http://localhost:8080/jasperserver`

User Id: `jasperadmin`

Password: `jasperadmin`



If you updated your sample data in the sections above, your `jasperadmin` password might be reset to `jasperadmin`.

Your JasperServer instance has now been upgraded to 3.7. If there are problems on startup or login refer to troubleshooting section [A.2, “Database Connectivity Errors,” on page 66](#).

7.9 Additional Notes on JasperServer Upgrade

7.9.1 Handling JasperServer Customizations

If you made modifications or customizations to your JasperServer 3.5 application, these configurations are typically found in the `WEB-INF/applicationContext-*.xml` set of files.

Configuration modifications such as client specific security classes or LDAP server configurations, need to be hand copied from the older 3.5 environment and re-integrated into the new 3.7 environment.

7.9.2 Clearing the Application Server Work Directory

Application Servers have work directories where JSP files are compiled and cached and other objects are stored. These can potentially cause errors when updating to a new WAR file. The buildomatic `deploy-webapp` target automatically clears the application server’s cache. However, it is a good practice to double-check the work directory when doing an upgrade. Here would be the commands for Tomcat:

1. Change directory:
`cd <tomcat>/work`
2. Delete all files under the work directory.

7.9.3 Clearing the Repository Cache Table

In the jasperserver database, compiled JasperReports are cached in the `JIREpositoryCache` table for increased efficiency at runtime. In some cases, you may encounter errors running reports after an upgrade. Because the JasperReports JAR is typically updated with each new JasperServer release, old cached items can get out of date and thus cause errors at runtime. If you encounter errors that mention a JasperReports “local class incompatible,” you should check your repository cache table.

To manually clear this table, run a SQL command similar to the following:

```
update JIREpositoryCache set item_reference = null;
delete from JIREpositoryCache;
```



You can clear your jasperserver repository cache manually using the above command (or a similar command).

7.9.4 Updating the XML/A Connections (Optional)

When you upgrade your sample data to 3.7, your XML/A connection sample data will be updated. XML/A connections use JasperServer login accounts for authentication. Because of this, and because you would normally modify your default jasperadmin password as a standard security procedure, your XML/A connection may fail due to a mismatched password.

If you would like to update your XML/A connections, refer to section [6.11, “Notes on Updating XML/A Connection Definitions,” on page 42](#).

7.9.5 Upgrading the Liferay Portal

JasperServer can be configured to run with the Liferay Portal. If your JasperServer is set up to run with Liferay you must do the following steps as part of the upgrade process.

1. You will need to delete the `webapps/Jaspersoft` folder of the application server hosting Liferay. This deletes libraries used in older versions that conflict with libraries in the latest version.
2. Once this folder is deleted, you can deploy the new portlet WAR.

8 UPGRADE USING SQL UPGRADE SCRIPT

With JasperServer 3.7, it is possible to use SQL scripts to upgrade from JasperServer 3.5.

This chapter details a set of steps used to carry out an upgrade using the provided SQL upgrade scripts. Using these upgrade steps, you will change your existing 3.5 jasperserver database to become a 3.7 jasperserver database. The database change will be carried out using a SQL script containing CREATE and ALTER TABLE statements.

This chapter contains the following sections:

- [Backing Up Your JasperServer 3.5 Instance](#)
- [Preparing the JasperServer 3.7 WAR File Distribution](#)
- [Configuring Buildomatic for Your Database and Application Server](#)
- [Upgrading to JasperServer 3.7](#)
- [Starting JasperServer 3.7](#)
- [Logging Into JasperServer 3.7](#)

8.1 Backing Up Your JasperServer 3.5 Instance

First you must backup your JasperServer WAR file and your jasperserver database so that they can be restored in case there is a problem with the upgrade. These steps are performed from the command line in a Windows or Linux shell.

The following instructions are for the MySQL database. For other databases, consult your DB administration documentation for back up information.

8.1.1 Backing Up Your JasperServer CE WAR File

Back up the jasperserver directory in Tomcat to a backup directory.

1. Go to the <tomcat> directory.
2. Make a new directory named js-3.5-war-backup.
3. Copy <tomcat>/webapps/ jasperserver to <tomcat>/js-3.5-war-backup.
4. Delete the <tomcat>/webapps/jasperserver directory.

8.1.2 Backing Up Your JasperServer Database

Go to the location where you originally unpacked the 3.5 WAR file distribution zip.

1. Go to the <js-install-3.5> (the location of your original unpacked 3.5 WAR file distribution)

2. Run the following command:

Windows: `mysqldump --user=root --password=<password> jasperserver > js-db-3.5-dump.sql`

Linux: `mysqldump --user=root --password=<password> --host=127.0.0.1 jasperserver > js-db-3.5-dump.sql`



If you receive an error about packet size, see section [A.2.5, “Maximum Packet Size in MySQL,”](#) on page 67.

8.2 Preparing the JasperServer 3.7 WAR File Distribution

We will use the buildomatic scripts included in the 3.7 WAR file distribution ZIP release package in order to carry out the upgrade. Follow the steps in the sections listed below to obtain and unpack the WAR file distribution ZIP file:

1. Follow steps in section [5.2, “Obtaining the WAR File Distribution Zip,”](#) on page 26.
2. Follow steps in section [5.3, “Unpacking the WAR File Distribution Zip,”](#) on page 26.

8.3 Configuring Buildomatic for Your Database and Application Server

This upgrade procedure is based on using the “buildomatic” scripts which are included with the WAR File Distribution ZIP release package.

Follow the configuration steps that match your database and application server in section [5.6, “Configuring the Buildomatic Scripts,”](#) on page 27. Once your buildomatic scripts are configured, proceed to the next section.

8.4 Upgrading to JasperServer 3.7

Now that your buildomatic scripts have been configured, you can complete the upgrade.

The steps below will do an “in-database” upgrade. Your existing 3.5 jasperserver database will be changed into a 3.7 jasperserver database.

Run the following commands:



Make sure you have backed up your `jasperserver` database before proceeding.

Make sure you have backed up your JasperServer 3.5 WAR file before proceeding.

Commands	Description
<code>cd <js-install-3.7>/buildomatic</code>	
<code>js-ant upgrade-3.5-3.7-ce</code>	Executes SQL script to carry out database upgrade.
<code>js-ant import-sample-data-upgrade-ce</code>	(Optional) This step is optional. Loads the 3.7 sample data.
<code>js-ant deploy-webapp-ce</code>	Delete existing 3.5 war file, deploy 3.7 war file.

Your JasperServer instance has now been upgraded to 3.7. If there are problems on startup or login refer to troubleshooting section [A.2, “Database Connectivity Errors,”](#) on page 66.

8.5 Starting JasperServer 3.7

You may now start your application server.

8.6 Logging Into JasperServer 3.7

If your application server and JasperServer 3.7 were started cleanly, you can now prepare to login.

8.6.1 Clearing Your Browser Cache

Before you log into 3.7, make sure and clear your Browser cache. Javascript files, which enable UI elements of JasperServer, are typically cached by the Browser. The cache should be cleared to ensure that the most current files are used.

Your end users should also be reminded to clear their Browser caches before logging in.

8.6.2 Logging Into JasperServer

You may now log into JasperServer using the same URL you used before the upgrade.

9 UPGRADE NOTES FOR JASPERSERVER 3.0 AND 3.1

If you are upgrading from an older JasperServer version such as 3.0 or 3.1 then you must first upgrade to JasperServer 3.5 before upgrading to 3.7.

The steps to carry out this operation are fully documented in the JasperServer 3.5 *Installation Guide*. You will need to download the JasperServer 3.5 release package to get the relevant files and documentation.

To download the JasperServer 3.5 WAR file distribution zip package, go to the JasperForge.org web site download area.

10 CHANGING PASSWORD ENCRYPTION IN JASPERSERVER

By default, password encryption is enabled in JasperServer and passwords are stored as cipher text in the database. System administrators can change the encryption algorithm, as well as specify the salt key used to initiate the encryption algorithm.

This chapter describes the procedure to enable password encryption if you have a JasperServer instance without encryption turned on. For more information on JasperServer encryption options refer to the *JasperServer Administrator Guide*.

This chapter contains the following sections:

- [Backing Up Your JasperServer Database](#)
- [Stopping Your Application Server](#)
- [Running the Repository Export Utility](#)
- [Specifying Encryption Settings in the JasperServer WAR](#)
- [Specifying Encryption Settings for the Import Utility](#)
- [Recreating the JasperServer Database](#)
- [Importing Your Repository Data](#)
- [Starting the Application Server](#)
- [Logging Into JasperServer](#)

10.1 Backing Up Your JasperServer Database

As a precaution, you must back up your jasperserver database in case there is any problem while configuring encryption.

To back up the default MySQL database, go to the <js-install> directory and run the following command:

```
Windows: mysqldump --user=root --password=<password> jasperserver > js-db-dump.sql
Linux:    mysqldump --user=root --password=<password> --host=127.0.0.1 jasperserver >
         js-db-dump.sql
```



If you installed JasperServer from the installer, you may specify `--user=jasperdb` in this command.

If you receive an error about packet size, see section [A.2.5, “Maximum Packet Size in MySQL,”](#) on page 67.

10.2 Stopping Your Application Server

You can now stop your application server. You should leave your database running.

10.3 Running the Repository Export Utility

The repository export utility writes out all of the JasperServer repository objects to a set of XML and binary format files. The output of the export operation is known as an export catalog.

To create the export catalog, go to the <js-install>/scripts directory and run the following commands. Note that there are two dashes (--) in front of the command options:

Windows: `js-export.bat --everything --output-dir js-backup-catalog`

Linux: `js-export.sh --everything --output-dir js-backup-catalog`

For information on running the export utility, refer to chapter 11, “Configuring the Import-Export Utilities,” on page 61.

10.4 Specifying Encryption Settings in the JasperServer WAR

JasperServer uses the Spring configuration and Acegi security to enable and configure encryption. These options can allow you to have a strong encryption setup. This section is focused on the minimal configuration necessary for enabling encryption.

1. Open the following file for editing:

```
<tomcat>/jasperserver/WEB-INF/ApplicationContext-security.xml
```

2. In the definition of the `daoAuthenticationProvider` bean, there is a commented-out reference to the `passwordEncoder` bean. Look for the section of the XML file that starts with:

```
<bean id="daoAuthenticationProvider"
```

In this bean definition, uncomment the reference to `passwordEncoder`. This causes the `passwordEncoder` logic to be used. After removing the commenting characters the line should look like the following:

```
<property name="passwordEncoder"><ref local="passwordEncoder"/></property>
```

3. Enable encryption in the `passwordEncoder` bean by modifying the `allowEncoding` property. Change the value from `false` to `true` so that it looks like the following:

```
<property name="allowEncoding"><value>true</value></property>
```

4. If the default DESede algorithm is used, the `secretKey` represents the salt key and must be 24 characters. By default, the `keyInPlainText` property is `true`, meaning the key can be in plain text to make it easier to enter, for example:

```
<property name="keyInPlainText"><value>true</value></property>
```

```
<property name="secretKey"><value>jaspersoftInSanFrancisco</value></property>
```



The text `jaspersoftInSanFrancisco` is 24 characters long, therefore the two properties above work with their default values. However, for better security, we recommend that they be changed.

5. The last two properties may be left unchanged. They are set to DESede by default. The default values are the following:

```
<property name="secretKeyAlgorithm"><value>DESede</value></property>
```

```
<property name="cipherTransformation"><value>DESede/CBC/PKCS5Padding</value></property>
```



As described in the JasperServer Administrator Guide, the `secretKey`, `secretKeyAlgorithm`, and `cipherTransformation` property settings must be consistent with each other. For example, different algorithms expect different key lengths.

6. Save and close the file. Encryption is now enabled for the JasperServer application upon the next restart.

The following table summarizes the options described above.

Password Configuration Options		
Configuration File		
...\\WEB-INF\\applicationContext-security.xml		
Property	Bean	Description

Password Configuration Options, continued		
passwordEncoder	daoAuthenticationProvider	Comment this property out to disable the encryption of passwords.
allowEncoding	passwordEncoder	Determines whether JasperServer should encrypt the passwords it writes to the database. Set it to TRUE to use encrypted passwords
secretKey	passwordEncoder	The “salt” key to use as a pass phrase when encrypting passwords. This string is used to encrypt passwords. This value can be a simple string or a numeric representation that can be parsed by Integer.decode(). For example: String: This is my secret key Numeric representation: 0xC8 0x43 0x29 0x49 0xAE 0x25 0x2F 0xA1 0xC1
keyInPlainText	passwordEncoder	Determines whether the secret key is a simple string or a numeric representation. Set this parameter to TRUE if the secretKey is a string; set it to FALSE if the key is a numeric representation.
secretKeyAlgorithm	passwordEncoder	The name of the algorithm to use, such as DESede.
cipherTransformation	passwordEncoder	The name of the transformation, such as DES/CBC/PKCS5Padding.

The secretKey, secretKeyAlgorithm, and cipherTransformation must be consistent with each other. For example, if the secretKeyAlgorithm is DESede, the secretKey must be 24 bytes long. For more information about secretKey, secretKeyAlgorithm, and cipherTransformation, see Sun’s `javax.crypto` documentation.

10.5 Specifying Encryption Settings for the Import Utility

Before starting JasperServer, you must convert the plain text passwords that are currently stored in the repository export catalog that you created in section 10.1, “[Backing Up Your JasperServer Database](#),” on page 57. These plain-text passwords need to be converted to cipher text and reloaded into the database in order to successfully login after the server restarts. To do this, you must add the same encryption settings to the configuration file that is used by the import and export utilities.

1. Open the following configuration file for editing:
`<js-install>/scripts/config/applicationContext-security.xml`
2. This file contains the `passwordEncoder` bean definition, the same as in the JasperServer WAR, only by itself. Perform the same modifications to this file as in the procedure in section 10.4, “[Specifying Encryption Settings in the JasperServer WAR](#),” on page 58.

10.6 Recreating the JasperServer Database

Next, drop your existing `jasperserver` database and recreate an empty `jasperserver` database.

10.6.1 Dropping and Recreating in MySQL

1. Change directory to `<js-install>/buildomatic/install_resources/sql/mysql`.
2. Log into your MySQL client:
`mysql -u root -p`
3. Drop the `jasperserver` database, create a new one, and load the `jasperserver` schema:

```
mysql>drop database jasperserver;
mysql>create database jasperserver character set utf8;
mysql>use jasperserver;
mysql>source js-create.ddl;
```

10.6.2 Dropping and Recreating in PostgreSQL

1. Change directory to <js-install>/buildomatic/install_resources/sql/postgresql
2. Start `psql` using an administrator account such as `postgres`:

```
psql -U postgres
```
3. Drop the `jasperserver` database, create a new one, and load the `jasperserver` schema:

```
drop database jasperserver;
create database jasperserver encoding='utf8';
\c jasperserver
\i js-create.ddl
```

10.7 Importing Your Repository Data

The import utility reloads all of your repository data. As the data is being saved to the repository, the password fields that were plain text are encrypted using the encryption settings you made in the sections above.

To import your backup catalog to the repository:

1. Change directory to <js-install>/scripts.

Run the import utility with the command for your platform. Note that there are two dashes (`--`) in front of the command options.:

```
Windows: js-import.bat --input-dir js-backup-catalog
Linux:   js-import.sh --input-dir js-backup-catalog
```

For information on running the import utility, see chapter 11, “[Configuring the Import-Export Utilities](#),” on page 61.

10.8 Starting the Application Server

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

10.9 Logging Into JasperServer

You can now log into JasperServer.

Enter your user ID and password in the same manner as you did before encryption was turned on. You can check the contents of the `JUser` table in the `jasperserver` database and examine the password column to see that the password is no longer stored in plain text.

11 CONFIGURING THE IMPORT-EXPORT UTILITIES

The import and export utilities let you add resources to or extract resources from the JasperServer repository. Typically, users export data from their previous instance and import it into their new installation when upgrading JasperServer. The import utility is also used at installation time in order to load the JasperServer sample data into the repository.

Please refer to the *JasperServer Administrator Guide* for more information on command options for the import and export utilities.

This chapter contains the following sections:

- **Import-Export Configuration Files**
- **Changing Your Configuration Settings**
- **Deploying a Database Driver**
- **Running Import or Export**

11.1 Import-Export Configuration Files

Among the scripts directory in your installation directory, you will find the following files that make up the main parts of the import-export utility. These are the files to use or modify to make configuration changes.

File or Location	Purpose
scripts/js-import.bat	Import batch script for Windows.
scripts/js-import.sh	Import shell script for Linux
scripts/config/js.jdbc.properties	Database and hibernate dialect settings file
scripts/config/applicationContext-*.xml	Spring configuration files
scripts/lib	All of the JasperServer jar files and JDBC drivers

11.2 Changing Your Configuration Settings

When you install JasperServer from the installer binary, the import and export utilities are automatically configured. However, if you are doing a manual installation from the WAR file distribution you must modify the following configuration file to include your database settings:

```
<js-install>/scripts/config/js.jdbc.properties
```

The following table gives sample settings for each database. Modify the items in bold to match your own installation. You may specify an encrypted password instead of the clear-text password by default.

If your repository contains international characters, you may need to perform additional configuration for the import and export utilities. See section [A.8, “Exporting a Repository That Contains UTF-8,”](#) on page 68.

Database	Sample Property Values
MySQL	<pre>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 or metadata.jdbc.encryptedPassword=encrypted-password</pre>
PostgreSQL	<pre>metadata.hibernate.dialect= com.jaspersoft.hibernate.dialect.PostgresqlNoBlobDialect metadata.jdbc.driverClassName=org.postgresql.Driver metadata.jdbc.url=jdbc:postgresql://localhost:5432/jasperserver metadata.jdbc.username=postgres metadata.jdbc.password=postgres or metadata.jdbc.encryptedPassword=encrypted-postgres</pre>

11.3 Deploying a Database Driver

In order for the import-export utility to run, it will need the proper JDBC driver. This allows a connection to be made to the JasperServer database.

Put the appropriate JDBC driver JAR into the following directory:

```
<js-install>/scripts/lib
```

All Jaspersoft distributed JDBC drivers can be found at these locations:

```
<js-install>/buildomatic/conf_source/db/<db-type>/jdbc
```

```
<js-install>/scripts/drivers/<db-type>/jdbc
```

```
<js-install>/scripts/lib
```

Double-Check JDBC Drivers in scripts/lib:

The `<js-install>/scripts/lib` directory already has JDBC drivers located there for customer convenience. However, if you are using a different driver (a newer one, or one from a different provider), then you will need to make sure and delete any old JDBC driver that goes with your database type.

11.4 Running Import or Export

To see that the import and export utilities are properly configured, you can run the scripts using the `--help` option (with two dashes `--`) that displays the command options:

```
Windows: js-import.bat --help
         js-export.bat --help
```

```
Linux:   js-import.sh --help
         js-export.sh --help
```

If your repository contains international characters, you may need to perform additional configuration for the import and export utilities. See section [A.8, “Exporting a Repository That Contains UTF-8,” on page 68](#).

For complete information on the standard import-export options refer to the *JasperServer Administrator Guide*.

APPENDIX A TROUBLESHOOTING

This appendix contains the following sections:

- **Installer Freezes**
- **Database Connectivity Errors**
- **Error Running a Report**
- **Database Error after Changing MySQL Port Number**
- **Case Sensitivity for Table and Column Names**
- **Java Out of Memory Error**
- **Error Running Scheduled Report**
- **Exporting a Repository That Contains UTF-8**
- **JBoss Modifications**
- **PostgreSQL: Job Scheduling Error**
- **Error Running Buildomatic Scripts**
- **Troubleshooting on Solaris**
- **Disabling User Session Persistence in Application Servers**

A.1 Installer Freezes

If you run the JasperServer installer on any platform and the installer freezes, it is helpful to look at the log file created by the installer. This log file records the status and completion of installer operations. If your installer has had an explicit error, there may be a specific error message in the log. At a minimum, the log file should help narrow where the error has occurred even if there is not a specific error message.

You can find the installer log in the following locations:

Windows: C:\Documents and Settings\\Local Settings\Temp\bitrock_installer_<number>.log

Linux: /tmp/bitrock_installer.log or bitrock_installer_<number>.log

If you have tried multiple installs, make sure you view the most recent install log file.

A.2 Database Connectivity Errors

The most common problems encountered with a new JasperServer instance are database configuration problems. This section contains information that may help resolve such issues.

A.2.1 Testing the Database Connection

The simplest database configuration problem is an incorrect user name or password. If you encounter database problems upon startup or login, check the user name and password by logging directly into your RDBMS as described in the following sections.

You can connect to your database using the database configuration settings that are found in JasperServer. This validates the database hostname, port, user id, and password that are being used.

If you are having trouble logging into JasperServer on the login page, you can check the users and passwords that exist by viewing the contents of the `jasperserver.JIUser` table.

A.2.1.1 Logging In to MySQL

Start MySQL from the command line and try to log in directly using the `jasperdb` user, for example:

```
<mysql>/bin/mysql -u jasperdb -p or  
<mysql>/bin/mysql -u root -p
```

You are prompted for a password for the user you specified on the command line. Enter the appropriate password to login. The default password used in the JasperServer sample configuration scripts is `password`.

A.2.2 Configuration File Locations

JasperServer configuration properties are found in the following files, according to your application server:

```
Tomcat:  <tomcat>/webapps/jasperserver/META-INF/context.xml  
        <tomcat>/webapps/jasperserver/WEB-INF/hibernate.properties  
        <tomcat>/apache-tomcat/webapps/jasperserver/WEB-INF/web.xml  
JBoss:  <jboss>/server/default/deploy/js-mysql-ds.xml or js-postgresql-ds.xml  
        <jboss>/server/default/deploy/jasperserver.war/WEB-INF/hibernate.properties  
        <jboss>/server/default/deploy/jasperserver.war/WEB-INF/web.xml  
        <jboss>/server/default/deploy/jasperserver.war/WEB-INF/jboss-web.xml  
GlassFish: <glassfish>/domains/domain1/autodeploy/jasperserver.war/WEB-INF/hibernate.properties  
          <glassfish>/domains/domain1/autodeploy/jasperserver.war/WEB-INF/js.quartz.properties  
          <glassfish>/domains/domain1/config/domain.xml
```

A.2.3 Special Configuration Case under Tomcat

If you installed JasperServer using the WAR file distribution file and handled the steps manually, Tomcat may have an additional (confusing) database configuration.

The special case occurs when you have deployed the `jasperserver.war` file into the Tomcat `webapps` directory. Valid JasperServer WAR deployments can be based on a single file (`jasperserver.war`) or an “unpacked” WAR file directory (`jasperserver` directory).

If you use a single WAR file for deployment under Tomcat, Tomcat takes the following steps (for instance in Tomcat 5.5):

- Unpack the `jasperserver.war` file into a new directory named `jasperserver`.
- Take the `jasperserver/META-INF/context.xml` file and copy it to a new file:

```
<tomcat>/conf/Catalina/Localhost/jasperserver.xml
```

This database configuration in `<tomcat>/conf` tree overrides the `context.xml` found in your `jasperserver` directory. If you are having database trouble in this scenario, it is recommended that you keep things simple by:

1. Deleting your `<tomcat>/webapps/jasperserver.war` file. This causes the `jasperserver` directory to be used.
2. Deleting your `<tomcat>/Catalina/Localhost/jasperserver.xml`. This causes the `META-INF/context.xml` from your `jasperserver` directory to be used.

A.2.4 Connect to Installed/Bundled Version of MySQL

These steps are for connecting under Linux.

If you have installed JasperServer using the bundled version of MySQL, you may want to connect to MySQL with the `mysql` command line application to examine the database. In order to connect, you will need to specify the socket that MySQL is using, as specified in the file `<install-dir>/jasperctl.sh`.

1. Get the socket file location by using the Linux `ps` (process status) command:

```
ps -ef | grep mysql
```

2. This displays lots of information. Look for the `--socket` value, for example:

```
... /home/devuser/jasperserver-3.7/mysql/bin/mysqld_safe --port=3306 \
--socket=/home/devuser/jasperserver-3.7/mysql/tmp/mysql.sock ...
```

3. Then run a command similar to the following:

```
/home/devuser/jasperserver-3.7/mysql/bin/mysql -u jasperdb -p \
--socket=/home/devuser/jasperserver-3.7/mysql/tmp/mysql.sock
```

A.2.5 Maximum Packet Size in MySQL

If you are upgrading or importing into a MySQL database and your repository contains large objects such as images, you may see an error such as:

```
ERROR 1153 (08S01): Got a packet bigger than 'max_allowed_packet' bytes
```

The default `max_allowed_packet` on the MySQL server is 1M (one Megabyte = 1,048,576 bytes). The most effective fix is to change this value in the server configuration to accommodate the largest resource stored in your repository. The server configuration file is typically named `my.ini` and is located in the MySQL root directory, although this may vary. Change the configuration setting to a larger value, for example:

```
max_allowed_packet = 4M
```

For more information, see <http://dev.mysql.com/doc/refman/5.0/en/packet-too-large.html>.

After changing this value, restart the MySQL server. Then perform the upgrade or import step again.

A.3 Error Running a Report

If you can log into JasperServer but encounter an error when running a report within JasperServer, you can browse the JasperServer repository to identify and resolve the problem.

One common problem with an individual report is the data source being used. To validate a data source connection:

1. Log into JasperServer as a user with administrative permissions and locate the report unit that returns errors.
2. Select the report and click the **Edit** button in the toolbar to identify the data source the report uses. The data source name is found on the fourth edit page.
3. Select this data source in the repository and click the **Edit** button in the toolbar.
4. Review the information specified for this data source.
5. Click the **Test Connection** button in order to validate the connection.
6. Click **Save** or **Cancel** when you are done.
7. Test your report. If it still returns errors, edit the data source again and try checking other values, such as the port used by the database.

A.4 Database Error after Changing MySQL Port Number

The default port for MySQL is 3306. If you entered a different port when you installed MySQL, the JasperServer installer configures them to communicate properly. If the MySQL port number has changed, or if you encounter a problem, check the database configuration files to verify your port number.

If it is incorrect, change it to the correct port number, save the file, and restart the application server. For more information, see section [A.2.2, “Configuration File Locations,” on page 66](#).

A.5 Case Sensitivity for Table and Column Names

Some databases are case-sensitive with respect to table names and will consider “customer” and “Customer” to be two different tables. If JasperServer is using a case-sensitive database, it’s important that the table names specified in query strings in the JRXML file of a saved report match the actual table names found in the database. A mismatch may occur if you are transferring data from one database to another, which may cause the capitalization of table names to change.

Under Windows MySQL, table and column names are *not* case-sensitive.

Under Linux MySQL, table and column names are case-sensitive. Linux MySQL can be configured to be non-case-sensitive by setting the configuration parameter `lower_case_table_names` to 1 in the `my.ini` or `my.cnf` file. For more information search the MySQL documentation for a section about identifier case sensitivity.

Table and column names in PostgreSQL are case-sensitive.

A.6 Java Out of Memory Error

If you encounter a Java out of memory error, it is suggested that you increase your Java heap size setting. See section [5.8, “Setting Java JVM Options,” on page 29](#). It is recommended that you add `-Xms128m -Xmx512m` to your `JAVA_OPTS` setting, but you may increase that to `-Xms512m -Xmx1024m` or larger if your server can support higher settings.

This Java option is set within the application server, so you must set it and then restart your application server.

A.7 Error Running Scheduled Report

If you setup a scheduled report, chose to run it, and chose to save it as HTML or RTF, the report size can potentially get quite large. If you are running MySQL and you get the following error:

```
JDBC exception on Hibernate data access
org.hibernate.exception.GenericJDBCException: could not insert
```

the problem may be the default size of the MySQL “blob” datatype. You can increase the size of this datatype by updating your `my.ini` or `my.cnf` MySQL configuration file with the following setting:

```
max_allowed_packet=32M
```

A.8 Exporting a Repository That Contains UTF-8

The following errors may happen when you have international characters in database objects, for example user ids.

A.8.1 Error During JasperServer 1.2 Export

Upgrading typically requires doing an export operation on your database. If you get a null pointer exception such as the following:

```
java.lang.NullPointerException
```

```
ResourceExporter.exportResource(ResourceExporter.java:258)
```

it may be due to an incorrect character in the file `scripts/ji-export-util/jdbc.properties`. Check the URL in this file; it should look like the following:

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

Note the ampersand `&` character. It is incorrect if it appears as `&`. The `&` is only correct in an HTML or XML context. It is incorrect in a properties file.

A.9 JBoss Modifications

A.9.1 JBoss 4.2 XML/A Connection Fix

JBoss 4.2 includes the JBossWS service as a standard, default feature. JasperServer has web services support for XML/A connections. The web services classes in JasperServer and JBoss can conflict and cause the following error when attempting to utilize a JasperServer XML/A connection:

```
javax.xml.soap.SOAPException: Unable to create message factory for
SOAP: org.jboss.ws.core.soap.MessageFactoryImpl
```

To prevent the web services class conflict, set the special Java JVM options for JBoss 4.2, as described in section [6.1, “Setting JVM Options for Application Servers,”](#) on page 33.

A.9.2 JBoss Large INFO Log Message on Analysis Drill-through

JBoss has an internal mechanism to track and log information on unclosed JDBC connections. JasperServer Analysis leaves a connection open for performance reasons when doing an analysis drill-through. In this case, JBoss puts a large INFO level message into the `server.log`. To silence this INFO message

1. Open the JBoss `log4j` configuration file for editing:

```
<jboss>/server/default/conf/jboss-log4j.xml
```

2. Set the logging level for the `CachedConnectionManager` class to the following value

```
<category name="org.jboss.resource.connectionmanager.CachedConnectionManager">
<priority value="WARN"/>
</category>
```

A.9.3 JBoss 4.0 Log4j Error on Startup

An error occasionally seen in JBoss 4.0 (tested on 4.0.5) has the following exception message:

```
log4j:ERROR "org.jboss.logging.util.OnlyOnceErrorHandler
```

JBoss is normally distributed with the `log4j` facility enabled. `Log4j` is initialized at JBoss startup. JasperServer also includes and uses `log4j`. When JBoss loads the JasperServer WAR file, the `OnlyOnceErrorHandler` exception can occur. This error is not fatal to JasperServer, but can cause confusion when seen in the JBoss server log.

To remove this error, you can delete the JasperServer version of the `log4j.jar` file:

```
<jboss>/server/default/deploy/jasperserver.war/WEB-INF/lib/log4j-3.7.jar
```

A.9.4 JBoss 5.0.1 and 5.1.x Error

With JBoss 5.0.1 and 5.1.x, you may see the following error:

```
org.jboss.xb.binding.JBossXBRuntimeException: Failed to create a new SAX parser
Caused by: java.lang.ClassCastException
```

Cause: There is a class conflict with the `xercesImpl-3.7.jar` in JasperServer.

Action: Delete the following file:

<jboss>/server/default/deploy/jasperserver.war/WEB-INF/lib/xercesImpl-*.jar



When running the buildomatic scripts to deploy to JBoss, the xercesImpl-3.7.jar file is automatically deleted in order to fix this problem.

A.10 PostgreSQL: Job Scheduling Error

If the Quartz settings under the PostgreSQL database have not been updated to specify the driver delegate class specific to PostgreSQL you will get errors when you try and run a scheduled report. The errors would look similar to the following:

```
Error while fetching Quartz runtime information
org.quartz.JobPersistenceException: Couldn't obtain triggers: Bad value for type int
org.postgresql.util.PSQLException: Bad value for type int
```

If you see this error you will need to check your Quartz properties file found at the following location:

<tomcat>/webapps/jasperserver/WEB-INF/js.quartz.properties

You should make sure that the following property does not have the standard driver delegate, but instead has the PostgreSQL specific driver delegate. It should look like the following for PostgreSQL:

```
org.quartz.jobStore.driverDelegateClass = org.quartz.impl.jdbcjobstore.PostgreSQLDelegate
```

A.11 Error Running Buildomatic Scripts

The buildomatic scripts depend on both Java and Apache Ant. There are two common configuration errors when attempting to do an installation using these scripts (if you are not using the included, bundled Apache Ant).

A.11.1 Missing Java JDK

If you have the Java JRE (Java Runtime Environment) instead of the JDK, you will not have the additional utilities that are required. In particular, the error might refer to the tools.jar similar to the following:

```
[exec] [ERROR] BUILD FAILURE
[exec] [INFO] -----
[exec] [INFO] Compilation failure
[exec] Unable to locate the Javac Compiler in:
[exec]   c:\Program Files\Java\jdk1.6.0_10\jre\..\lib\tools.jar
[exec] Please ensure you are using JDK 1.5 or above and
[exec] not a JRE (the com.sun.tools.javac.Main class is required).
[exec] In most cases you can change the location of your Java
[exec] installation by setting the JAVA_HOME environment variable.
```

Solution: Make sure to download and install the Sun Java JDK.



The Sun web site refers to the download as either “Java SE (JDK)” or “Java SE Development Kit (JDK).”

A.11.2 Forgot to Copy the File ant-contrib-3.7.jar

If you are using your own version of Ant and your Ant instance does not have the ant-contrib.jar in the lib directory, you will get an error similar to the following:

```
BUILD FAILED
c:\js-builds\jasperserver\buildomatic\install.xml:6: Problem: failed to create task or type if
```

Solution: copy `<js-install>/buildomatic/extra-jars/ant-contrib.jar` to your `<apache-ant>/lib` directory.

A.12 Troubleshooting on Solaris

When running the bundled Apache Ant scripts on the Solaris platform, you may see the following error:

```
ANT_HOME=../apache-ant: is not an identifier
```

Cause: The bundled Ant scripts are intended for the `bash` shell and may cause this error when run in the Bourne shell (`sh`).

Action: Run all `js-ant` targets in the `bash` shell explicitly, for example:

```
bash js-ant create-js-db
```

A.13 Disabling User Session Persistence in Application Servers

JasperServer stores non-serializable data in its user sessions, which can cause errors after restarting your application server:

```
Exception loading sessions from persistent storage  
Cause: java.io.NotSerializableException ...
```

The errors appear in the JasperServer log when users log in after the application server has been restarted. The errors do not appear to users, and they have no impact on JasperServer operations.

Because JasperServer user sessions are not persistent, you can configure your application server to disable persistence and avoid the error. For example, in Apache-Tomcat 5.5 and 6.0, edit the file `<tomcat>/conf/context.xml` and locate the following lines:

```
<!-- Uncomment this to disable session persistence across Tomcat restarts -->  
<!--  
<Manager pathname="" />  
-->
```

Remove the comment markers from lines 2 and 4 above, then restart Apache-Tomcat for the change to take effect. For other application servers, refer to the product documentation.

