



JASPERSERVER COMMUNITY EDITION

INSTALLATION GUIDE

RELEASE 3.5

jasperforge.org

© 2009 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 0409-JSO35-11 of the *JasperServer Community Edition Installation Guide*.

TABLE OF CONTENTS

1	Introduction	7
1.1	Conventions	7
1.2	Java Version Supported	8
1.3	JasperServer Distributions	8
1.4	Installer Distribution Support	8
1.4.1	Installer Distribution Components	8
1.4.2	Installing with Existing Components	9
1.4.3	Running Components as Windows Services	9
1.5	WAR File Binary Distribution Support	9
1.6	Release Notes	10
1.7	Prerequisites for Installation	10
1.8	System Requirements	10
1.9	Support for Internationalization	11
2	Installing JasperServer	13
2.1	Installation Steps	13
2.1.1	Welcome	13
2.1.2	Accept License Agreement	13
2.1.3	Installation Directory Location	14
2.1.4	Select Tomcat Configuration	14
2.1.5	Select MySQL Configuration	14
2.1.6	Install Sample Data	15
2.1.7	Install iReport	15
2.1.8	Ready to Install	15
2.1.9	Installation Complete Screen	15
2.1.10	Log Into JasperServer	16
2.2	Post-Installation Steps	16
2.2.1	Updates Made During Installation	16
2.2.2	Installer Output Log File	16
2.2.3	Check your Java JVM Options	16

3	Starting and Stopping JasperServer	17
3.1	Using JasperServer Start/Stop Scripts	17
3.2	Using Start/Stop Scripts Without Bundled Installation	17
3.3	Logging In to JasperServer	18
3.4	Starting the Included iReport	18
3.5	JasperServer Log Files	18
4	Uninstalling JasperServer	19
5	Installing From the WAR File Distribution	21
5.1	Introduction	21
5.1.1	Applications Supported by the WAR File Distribution	21
5.1.2	Application Versions Supported	21
5.2	Obtaining the WAR File Distribution Zip	22
5.3	Unpacking the WAR File Distribution Zip	22
5.4	Introduction to Buildomatic Scripts	22
5.5	Pre-Installation Steps	22
5.5.1	Checking Your Java Installation	22
5.5.2	About Bundled Apache Ant	23
5.5.3	Checking Your Application Server	23
5.5.4	Checking Your Database Server	23
5.6	Configuring the Buildomatic Scripts	23
5.6.1	Creating your Default Master Properties File	23
5.6.2	Edit Default Master Properties File	24
5.6.3	Setup for MySQL	24
5.6.4	How To Refresh Buildomatic Settings	24
5.7	Install JasperServer	24
5.7.1	Errors Running Buildomatic create-db Targets	25
5.8	Setting Java JVM Options	25
5.8.1	Set Java JVM Options for Tomcat or JBoss	25
5.8.2	Set Java JVM Options for GlassFish	25
5.9	Starting JasperServer	26
5.10	Logging In to JasperServer	26
5.10.1	JasperServer Heartbeat	26
5.11	Troubleshooting your JasperServer Configuration	27
5.11.1	JasperServer Startup Problems	27
5.11.2	Error Running a Report	27
5.12	Running the Import and Export Utilities	27
5.12.1	Running JS Export from Buildomatic	27
5.12.2	Running JS Import from Buildomatic	28
6	Additional Installation Information	29
6.1	Setting JVM Options for Application Servers	29
6.1.1	Tomcat and JBoss JVM Options	29
6.1.2	Tomcat as a Windows Service JVM Options	30

6.1.3	GlassFish JVM Options	30
6.2	Additional Buildomatic Configuration Information	31
6.2.1	Generated Buildomatic Property Files	31
6.2.2	Buildomatic Database Script (SQL) Files	32
6.2.3	Buildomatic WAR File Location	32
6.2.4	Buildomatic Sample Data Catalog ZIP Files	32
6.3	Additional Notes on MySQL Database	33
6.4	Notes on the Hibernate Properties File	33
6.5	Notes on JDBC Database Drivers	34
6.6	Notes on Database Connection for Tomcat	34
6.7	Notes on Datasource Definition for JBoss	34
6.7.1	Notes on Extra JBoss Configuration Step	35
6.8	Notes on Database Connection for GlassFish	35
6.9	Notes on JasperServer Logging	35
6.10	Notes on Report Scheduling Configuration	35
6.11	Notes on Updating XML/A Connection Definitions	35
7	Upgrade from JasperServer 3.0 or 3.1 to 3.5	37
7.1	Important Upgrade Information	37
7.1.1	Upgrade Scripts	37
7.1.2	General Procedure	37
7.1.3	Handling JasperServer Customizations	38
7.2	Back Up Your JasperServer 3.1 Instance	38
7.3	Export Your 3.1 Repository Data	38
7.3.1	Export Repository Data	38
7.4	Configure Buildomatic for Your Database and Application Server	39
7.5	Copy Your 3.1 Data Export File	39
7.6	Upgrade to JasperServer 3.5	39
7.6.1	Upgrade to 3.5	39
7.7	Starting JasperServer 3.5	40
7.8	Logging In to JasperServer 3.5	40
7.8.1	Clear Browser Cache	40
7.8.2	Login to JasperServer	40
7.9	Alternate Upgrade to JasperServer 3.5	40
7.9.1	Backup You Current 3.1 Instance	40
7.9.2	Update JasperServer WAR File to 3.5	40
7.9.3	Update Database to 3.5	40
7.10	Additional Notes on JasperServer Upgrade	41
7.10.1	Using mysqldump for Database Backup	41
7.10.2	Handling JasperServer Customizations	41
7.10.3	Clearing the Application Server Work Directory	41
7.10.4	Clearing the Repository Cache Table	41
7.10.5	Updating the XML/A Connections (Optional)	42

8	Upgrade Notes for JasperServer 2.1	43
8.1	Upgrading a MySQL Database	43
8.2	Issue Exporting and Importing from 2.0 to 3.0	43
8.2.1	Deleting Any 2.0 MondrianConnection Objects	44
8.2.2	Running a Special Operation to Convert	44
9	Changing Password Encryption in JasperServer	45
9.1	Backing Up Your JasperServer Database	45
9.2	Stopping Your Application Server	46
9.3	Running the Repository Export Utility	46
9.4	Specifying Encryption Settings in the JasperServer WAR	46
9.5	Specifying Encryption Settings for the Import Utility	47
9.6	Recreating the JasperServer Database	47
9.6.1	Dropping and Recreating in MySQL	47
9.6.2	Dropping and Recreating in PostgreSQL	47
9.7	Importing Your Repository Data	47
9.8	Starting the Application Server	48
9.9	Logging In to JasperServer	48
10	Configuring the Import-Export Utilities	49
10.1	Import-Export Configuration Files	49
10.2	Changing Your Configuration Settings	50
10.3	Deploying a Database Driver	50
10.4	Running Import or Export	50
Appendix A	Troubleshooting	51
A.1	Installer Freezes	51
A.2	Database Connectivity Errors	51
A.2.1	Testing the Database Connection	52
A.2.2	Configuration File Locations	52
A.2.3	Special Configuration Case under Tomcat	52
A.2.4	Connect to Installed/Bundled Version of MySQL	53
A.3	Error Running a Report	53
A.4	Database Error after Changing MySQL Port Number	53
A.5	Case Sensitivity for Table and Column Names	53
A.6	Java Out of Memory Error	54
A.7	Error Running Scheduled Report	54
A.8	JBoss Modifications	54
A.8.1	JBoss 4.2 XML/A Connection Fix	54
A.8.2	JBoss Large INFO Log Message on Analysis Drill-through	55
A.8.3	JBoss 4.0 Log4j Error on Startup	55
A.9	PostgreSQL: Job Scheduling Error	55
A.10	Error Running Buildomatic Scripts	55
A.10.1	Missing Java JDK	56
A.10.2	Forgot to Copy the File ant-contrib-<ver>.jar	56

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.

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
<apache-tomcat>	The directory where Tomcat is installed. If you plan to use the instance of Tomcat that is included in the installer, Tomcat is installed under the <js-install> directory
<jboss>	The directory where JBoss is installed
<glassfish>	The directory where GlassFish is installed
<mysql>	The directory where MySQL is installed. If you plan to use the instance of MySQL that is included in the installer, MySQL is installed under the <js-install> directory
<java>	The directory where java is installed

Convention	Description
<js-install>	The directory where user unpacks the WAR file distribution ZIP
jasperadmin/jasperadmin	The user name and password of the default JasperServer login administrative user
jasperdb or root/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, 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 database create and load scripts. 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
Linux	Red Hat Enterprise Linux SUSE Ubuntu

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)

Component	Description
MySQL Database	Database server: Use the bundled version or an existing version
Apache Tomcat	Web application container: 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 using bundled applications or if you have existing applications, the installer can deploy to these. For instance, if you already have Tomcat on your server you can choose to install to your “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 complete 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-<ver>-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-<ver>-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 Accept 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-<ver>

Linux: <USER_HOME>/jasperserver-<ver>

2.1.4 Select 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 Select 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:

Port	3306 (default value, user chooses alternate port if 3306 is in use).
Database User Name	jasperdb. The installer creates this user to connect to the jasperserver database
Database User Password	password. The installer uses this default 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:

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. Your MySQL must be 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.



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

2.1.6 Install 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, 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 Install iReport

iReport is the leading GUI-based JasperReports creation tool. It can communicate with a JasperServer instance to 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 In to JasperServer](#),” on page 18.



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 17.

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 26.

You can later enable or disable the heartbeat by modifying the jasperserver/WEB-INF/applicationContext-logging.xml file. For additional information on enabling and disabling the heartbeat component refer to <http://www.jaspersoft.com/heartbeat>.

2.1.10 Log Into JasperServer

You should now be ready to login to JasperServer. For information on default login credentials, go to section [3.3, “Logging In to JasperServer,” on page 18](#).

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.

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

File or Directory	Notes
Windows: bin/setclasspath.bat Linux: bin/setclasspath.sh	Modify JAVA_OPTS to add -Djs.license.directory
Windows: bin/setenv.bat Linux: bin/setenv.sh	Added this file. Sets increased Java memory allocation values to JAVA_OPTS. For additional settings, refer to section 5.8, “Setting Java JVM Options,” on page 25 .
Tomcat 5: common/lib Tomcat 6: lib	Add MySQL JDBC driver.

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

MySQL Updates	Notes
Database <code>jasperserver</code> created	This holds the JasperServer Application metadata and customer business data such as Reports, Analysis Views, Data Sources, Permissions, Roles, and Users.
Database user <code>jasperdb</code> created	JasperServer connects to the database using this user.
Sample database <code>foodmart</code> created	Database created if install sample data option was chosen.
Sample database <code>sugarcrm</code> created	Database created if install sample data option was chosen.

2.2.2 Installer Output Log File

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\<username>\Local Settings\Temp\bitrock_installer_<number>.log`

Linux: `/tmp/bitrock_installer_<number>.log`

2.2.3 Check 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 25](#).

3 STARTING AND STOPPING JASPERSERVER

This chapter contains the following sections:

- [Using JasperServer Start/Stop Scripts](#)
- [Using Start/Stop Scripts Without Bundled Installation](#)
- [Logging In to 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 Pro > JasperServer Management > Start JasperServer**.
Click **Start > All Programs > JasperServer Pro > JasperServer Management > Stop JasperServer**.

- From a command line:

```
Windows: cd <js-install>/bin
          ./allclt.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 In to 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 Pro > JasperServer**.

If the login page appears, JasperServer has started properly. You may now login with the following username and password:

Username: jasperadmin - (Administrative user)

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:

Username: joeuser - (sample standard end-user)

Password: joeuser



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 Pro > 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 Pro > 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.

5 INSTALLING FROM THE WAR FILE DISTRIBUTION

This chapter contains the following sections:

- [Introduction](#)
- [Obtaining the WAR File Distribution Zip](#)
- [Unpacking the WAR File Distribution Zip](#)
- [Introduction to Buildomatic Scripts](#)
- [Pre-Installation Steps](#)
- [Configuring the Buildomatic Scripts](#)
- [Install JasperServer](#)
- [Setting Java JVM Options](#)
- [Setting Java JVM Options](#)
- [Starting JasperServer](#)
- [Logging In to JasperServer](#)
- [Troubleshooting your JasperServer Configuration](#)

5.1 Introduction

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.

5.1.1 Applications Supported by the WAR File Distribution

The instructions in this section and the following sections support the following configurations:

Target Configuration	Instructions Located In
Tomcat, JBoss, or GlassFish with MySQL	This chapter

5.1.2 Application Versions Supported

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-<ver>-bin.zip` in the compressed ZIP format. To download the WAR file distribution, navigate to the jasperforge.org downloads area.

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 a directory with the following naming structure:

```
jasperserver-<ver>-bin
```

Unpack to a directory such as Program Files 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	C:\jasperserver-<ver>-bin	<js-install>
Linux	/home/user/jasperserver-<ver>-bin	<js-install>

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. A bundled version of Apache Ant is included with this distribution package to simplify the installation procedures.

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

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 a Windows and a Linux batch script that is 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 jars is: `ant-contrib-<ver>.jar`. The `ant-contrib` jar enables conditional logic in Ant.

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 supply the path to your 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 database.

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:

```
<js-install>/buildomatic/sample_conf/mysql_master.properties
```

2. And rename to:

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

5.6.2 Edit Default Master Properties File

Now that you created the default_master.properties file, you must edit the file and add the settings that are specific to your database and application server.

5.6.3 Setup for MySQL

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

Edit default_master.properties:

Settings	Examples
appServerDir=<your setting>	appServerDir=c:\\apache-tomcat-6.0.18
dbUsername=<your setting>	dbUsername=root
dbPassword=<your setting>	dbPassword=password
dbHost=<your setting>	dbHost=localhost

5.6.4 How To Refresh 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 (“gen-config” is simply a convenience).

5.7 Install JasperServer

Now that your default_master.properties file has been edited, you can now 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 following directory:

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

After executing each Ant target below, you should look for the message BUILD SUCCESSFUL. If you encounter errors creating databases, see Section [5.7.1, “Errors Running Buildomatic create-db Targets,” on page 25](#).

Execute the following steps at the command line:

Commands	Description
cd <js-install>/buildomatic	loading sample data is optional
js-ant create-sugarcrm-db	loading sample data is optional
js-ant load-sugarcrm-db	loading sample data is optional
js-ant create-foodmart-db	loading sample data is optional
js-ant load-foodmart-db	loading sample data is optional

Commands	Description
<code>js-ant update-foodmart-db</code>	loading sample data is optional
<code>js-ant create-js-db</code>	
<code>js-ant init-js-db-ce</code>	
<code>js-ant import-minimal-ce</code>	
<code>js-ant import-sample-data-ce</code>	loading sample data is optional
<code>js-ant deploy-webapp-ce</code>	Tomcat and JBoss
	For GlassFish, see below

GlassFish:

The GlassFish application server must first be running as described in section 5.5.3, “[Checking Your Application Server,” on page 23](#)). 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 25](#).
- The GlassFish application server will be stopped. You will restart the server in a subsequent step.

5.7.1 Errors Running Buildomatic create-db Targets

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 information on manual creation for some database types see Section 6.3, “[Additional Notes on MySQL Database,” on page 33](#)

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. For GlassFish, you can set the JVM options using the asadmin utility or by editing the domain.xml config directly.

5.8.1 Set Java JVM Options for Tomcat or JBoss

See section 6.1.1, “[Tomcat and JBoss JVM Options,” on page 29](#) for detailed steps.

5.8.2 Set Java JVM Options for GlassFish

See section 6.1.3, “[GlassFish JVM Options,” on page 30](#) 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* `<apache-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 35 and look at `<app-server-deploy>/jasperserver/WEB-INF/logs/jasperserver.log`

5.10 Logging In to JasperServer

If JasperServer started up cleanly you should be able to login.

Login by going to the following URL:

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

Example:

`http://localhost:8080/jasperserver`

`http://my.jasperserver.host:8080/jasperserver`

The login page should appear after taking some time to compile the necessary JSP file.

Use the following credentials to login to the system:

Username	Password	Description
jasperadmin	jasperadmin	Administrative User

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

Refer to the *JasperServer 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"/>
```

For additional information on the heartbeat see <http://www.jaspersoft.com/heartbeat>.

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 51.

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 53.

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 JS Export from Buildomatic

The export target for ant has the following syntax:

```
js-ant export-ce -DexportFile=export-file-name -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 “.zip” for your output file name then a ZIP will automatically be created. Otherwise, a directory with files and sub-directories will be created (ie non-compressed set of files).

The exportArgs argument can contain more than one export option.

Examples:

The following examples are typical export commands:

```
js-ant export-help
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.zip
                  -DexportArgs="--uris /datasources --roles ROLE_USER"
js-ant export-ce -DexportFile=js-everything.zip -DexportArgs="--everything"
js-ant export-everything-ce -DexportFile=js-everything.zip
```

5.12.2 Running JS Import from Buildomatic

The import target for ant has the following syntax:

```
js-ant import-ce -DimportFile=import-file-name [-DimportArgs="import options"]
```

The import-file-name is handled as a ZIP file 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.

Examples:

The following examples are typical import commands:

```
js-ant import-help  
js-ant import-ce -DimportFile=my-reports.zip  
js-ant import-ce -DimportFile=my-datasources.zip -DimportArgs="--update"
```

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 MySQL Database](#)
- [Notes on the Hibernate Properties File](#)
- [Notes on JDBC Database Drivers](#)
- [Notes on Database Connection for Tomcat](#)
- [Notes on Datasource Definition for JBoss](#)
- [Notes on Database Connection for GlassFish](#)
- [Notes on JasperServer Logging](#)
- [Notes on Report Scheduling Configuration](#)
- [Notes on Updating XML/A Connection Definitions](#)

6.1 Setting JVM Options for Application Servers

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

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).



If you are running JBoss 4.2 under Java 1.5, you will need to set some specific JVM options so that the web services analysis XML/A feature of JasperServer works. For more information, see troubleshooting section [A.8.1, “JBoss 4.2 XML/A Connection Fix,”](#) on page 54.

On Windows:

Tomcat file	<apache-tomcat>/bin/setclasspath.bat or <apache-tomcat>/bin/setenv.bat
JBoss file	<jboss>/bin/run.bat

Settings 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 settings for Java 1.6 with web services	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
Helpful sample directory	<js-install>/scripts/java-settings - (contains text files for cut/paste)

On Linux:

Tomcat file	<apache-tomcat>/bin/setclasspath.sh or <apache-tomcat>/bin/setenv.sh
JBoss file	<jboss>/bin/run.sh
JAVA_OPTS setting 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 settings for Java 1.6 with web services	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"
Helpful sample directory	<js-install>/scripts/java-settings - (contains text files for cut/paste)

6.1.2 Tomcat as a Windows Service JVM Options

If Tomcat is running as a Windows service, then you would typically add the JasperServer Java Options to the Java Tab of the Tomcat Properties dialog. This dialog can be accessed from the standard Windows menus. For instance:

Start > Programs > Apache Tomcat 6.0 > Configure Tomcat

In the Apache Tomcat Properties dialog:

Click the Java Tab

In the Java Options box:

Add your JasperServer JAVA_OPTS values

Click Apply, then OK

6.1.3 GlassFish JVM Options

For GlassFish, the JVM settings are identical for Java 1.5 and Java 1.6.

Setting 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.

Setting 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</jvm-options>
<jvm-options>-Xss2m -XX:+UseConcMarkSweepGC -XX:+CMSClassUnloadingEnabled</jvm-options>
<jvm-options>-Djavax.xml.soap.MessageFactory=org.apache.axis.soap.MessageFactoryImpl</jvm-options>
<jvm-options>-Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis.soap.SOAPConnectionFactoryImpl
  </jvm-options>
<jvm-options>-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 contents of these files.

6.2.1 Generated Buildomatic 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 run JasperServer.

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-glassfish-ds.xml
js-jboss-ds.xml
js.jdbc.properties
js.quartz.base.properties
```

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`.

6.2.2 Buildomatic Database Script (SQL) Files

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:

- mysql/js-create.ddl
- mysql/quartz.ddl
- mysql/sugarcrm.zip
- mysql/supermart-update.sql

- postgresql/js-create.ddl
- postgresql/quartz.ddl
- postgresql/sugarcrm.zip
- postgresql/supermart-update.sql

6.2.3 Buildomatic 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`

6.2.4 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:

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

6.3 Additional Notes on MySQL Database

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.

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.

Change to the following directory:

```
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';
```

6.3.0.1 Manual Creation of Sample Databases

Change to the following directory:

```
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` 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
PostgreSQL: metadata.hibernate.dialect=com.jaspersoft.hibernate.dialect.PostgresqlNoBlob
           Dialect
```

6.5 Notes on JDBC Database Drivers

The buildomatic scripts will automatically copy the appropriate JDBC driver to your application server when you run the `deploy-webapp-ce` target.

The JDBC drivers stored in buildomatic are found, for instance, in the following locations:

```
<js-install>/buildomatic/conf_source/db/mysql/jdbc/<driver-name>.jar
<js-install>/buildomatic/conf_source/db/postgresql/jdbc/<driver-name>.jar
```

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
```

6.6 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-ce` 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.8, “JBoss Modifications,” on page 54](#).

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 Notes on Report Scheduling Configuration

The report scheduling feature of JasperServer allows reports to be run at pre-configured time intervals. The `js.quartz.properties` file that holds the scheduling configuration properties has some special settings depending on the particular database.

All database scheduling properties are automatically handled by buildomatic.

You can view the automatically generated `js.quartz.properties` file at the following location:

```
<js-install>/buildomatic/build_conf/default/js.quartz.properties
```

The final location of the `js.quartz.properties` file is:

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

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.

If you have JasperAnalysis enabled (via your license), 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. Login to 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 3.0 OR 3.1 TO 3.5

This chapter contains the following sections:

- **Important Upgrade Information**
- **Back Up Your JasperServer 3.1 Instance**
- **Export Your 3.1 Repository Data**
- **You will later copy the js-3.1-export.zip file to your <js-install-3.5> location.**
- **Configure Buildomatic for Your Database and Application Server**
- **Copy Your 3.1 Data Export File**
- **Upgrade to JasperServer 3.5**
- **Starting JasperServer 3.5**
- **Logging In to JasperServer 3.5**
- **Alternate Upgrade to JasperServer 3.5**
- **Additional Notes on JasperServer Upgrade**

7.1 Important Upgrade Information

This procedure describes the steps necessary to carry out an upgrade from JasperServer 3.0 or 3.1 to JasperServer 3.5. These steps will use the JasperServer WAR File Distribution release package and the included buildomatic scripts for the upgrade procedure.

7.1.1 Upgrade Scripts

The main upgrade procedure will use the import-export utility in order to move your data from 3.0 or 3.1 to 3.5.



The database schema did not change from 3.0 to 3.1. Therefore, the steps described in this section can be used to upgrade from 3.0 or 3.1 directly to JasperServer 3.5.

7.1.2 General Procedure

In order to carry out the upgrade procedure, you will be doing the following main steps:

1. Back up your 3.1 JasperServer instance
2. Export your 3.1 repository data
3. Upgrade your instance to 3.5

4. Import your 3.1 repository data

7.1.3 Handling JasperServer Customizations

If your instance of JasperServer 3.1 has any custom modifications or extensions, you will need to keep track of these and re-integrate them into your 3.5 instance after the upgrade is complete.

7.2 Back Up Your JasperServer 3.1 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.

1. Back up the jasperserver directory in Tomcat to a backup directory:
 - a. `cd <apache-tomcat>`
 - b. `mkdir js-3.1-war-backup`
 - c. `copy <apache-tomcat>/webapps/ jasperserver to <apache-tomcat>/js-3.1-war-backup`
 - d. delete the `<apache-tomcat>/webapps/jasperserver` directory
2. Back up the jasperserver database: MySQL. Go to the location where you originally unpacked the 3.0 or 3.1 WAR file distribution zip:
 - a. `cd <js-install-3.1>` (i.e., the location of your original unpacked 3.1 WAR file distribution)
 - b. Run the following command:

Windows: `mysqldump --user=jasperdb --password=<password> jasperserver > js-db-3.1-dump.sql`

Linux: `mysqldump --user=jasperdb --password=<password> --host=127.0.0.1 jasperserver > js-db-3.1-dump.sql`

Note You can use the jasperdb user or the root user to carry out this operation. For more information on running the `mysqldump` command, refer to section [7.10.1, “Using mysqldump for Database Backup,” on page 41](#).

3. Backup the jasperserver database: All Databases. For any particular database, consult your DB administration documentation for back up information.

7.3 Export Your 3.1 Repository Data

You will use the JasperServer `js-export` utility to export your 3.0 or 3.1 repository data. You will need to verify that the import-export utility is properly configured in your 3.0 or 3.1 system. If your import-export utility is not already configured, see chapter [10, “Configuring the Import-Export Utilities,” on page 49](#).

7.3.1 Export Repository Data

Go to the JasperServer 3.0 or 3.1 installation location and change to the “scripts” directory. This is where you will run the `js-export` command. You will save the resulting file and later copy it to your 3.5 install location:

1. `cd <js-install-3.1>/scripts`
2. Run the export script:

Windows: `js-export.bat --everything --output-zip js-3.1-export.zip`

Linux: `js-export.sh --everything --output-zip js-3.1-export.zip`

You will later copy the `js-3.1-export.zip` file to your `<js-install-3.5>` location.

7.4 Configure 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.

1. Follow the configuration steps that match your database and application server in section 5.6, “Configuring the Buildomatic Scripts,” on page 23.

Once your buildomatic scripts are configured, proceed to the section next section.

7.5 Copy Your 3.1 Data Export File

The 3.1 export file created previously should be copied to your 3.5 install location:

1. `cd <js-install-3.5>/buildomatic`
2. Copy the export file:
 Windows: `copy <path-to-js-install-3.1>\scripts\js-3.1-export.zip`
 Linux: `cp <path-to-js-install-3.1>/scripts/js-3.1-export.zip .`

7.6 Upgrade to JasperServer 3.5

Now that your buildomatic scripts have been configured and your 3.1 export data file has been copied, you can complete the upgrade.

7.6.1 Upgrade to 3.5

Run the following commands:



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

Commands	Description
<code>cd <js-install-3.5>/buildomatic</code>	
<code>js-ant drop-js-db</code>	This will delete your jasperserver db. Make sure it is backed up.
<code>js-ant create-js-db</code>	
<code>js-ant init-js-db-ce</code>	
<code>js-ant import-minimal-ce</code>	
<code>js-ant import-upgrade-ce</code>	Will look for <code>js-3.1-export.zip</code> in the current directory. Use <code>-DimportFile=<path-to-file>/js-3.1-export.zip</code> if your export file is not in the current directory.
<code>js-ant import-sample-data-upgrade-ce</code>	This step is optional. Loads the 3.5 sample data.
<code>js-ant deploy-webapp-ce</code>	

7.7 Starting JasperServer 3.5

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

7.8 Logging In to JasperServer 3.5

If your application server and JasperServer 3.5 were started cleanly, you can now prepare to login.

7.8.1 Clear Browser Cache

Before you login to 3.5, make sure and clear your Browser cache. Javascript files, which enable elements of the JasperServer UI, 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 Login to JasperServer

Login using the following URL, username and password:

URL: `http://localhost:8080/jasperserver`

Username	Password	Description
jasperadmin	jasperadmin	Administrative user



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 3.5. If there are problems on startup or login refer to troubleshooting section [A.2, “Database Connectivity Errors,” on page 51](#).

7.9 Alternate Upgrade to JasperServer 3.5

The Community Edition of JasperServer 3.0 or 3.1 can be upgraded to 3.5 by updating the database “in place” using a SQL upgrade script.

7.9.1 Backup Your Current 3.1 Instance

Follow the procedures described above to back up your JasperServer war file and your JasperServer database.

7.9.2 Update JasperServer WAR File to 3.5

Using the buildomatic target “deploy-webapp-ce”, you can update your jasperserver.war file to 3.5. See the section above [7.4, “Configure Buildomatic for Your Database and Application Server,” on page 39](#) for more details.

Or, you can update your application server manually by configuring and copying the `<js-install>/jasperserver.war` to your application server deploy directory.

7.9.3 Update Database to 3.5

First make sure you have backed up your jasperserver database.

Upgrade your MySQL 3.0 or 3.1 database by running the following commands:

```
cd <js-install>/scripts/upgrade
mysql -u root -u (or DB user jasperdb)
```



```
mysql>use jasperserver;
mysql>source upgrade-mysql-3.1.0-3.5.0.sql
mysql>exit
```

7.10 Additional Notes on JasperServer Upgrade

7.10.1 Using mysqldump for Database Backup

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.

The following command is an example of using `mysqldump` to back up the JasperServer database:

```
Windows: mysqldump --user=jasperdb --password=<your-password>
          jasperserver > js-db-3.1-dump.sql

Linux:   mysqldump --user=jasperdb --password=<your-password> --host=127.0.0.1
          jasperserver > js-db-3.1-dump.sql
```

For MySQL: If the repository contains file resources larger than one megabyte, you may encounter the following message when restoring the database: “ERROR 1153 (08S01): Got a packet bigger than 'max_allowed_packet' bytes”. This error requires adjustment of your MySQL server configuration.



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

7.10.2 Handling JasperServer Customizations

If you made modifications or customizations to your JasperServer 3.0 or 3.1 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 3.0 or 3.1 environment and re-integrated into the new 3.5 environment.

7.10.3 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-ce`” 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 <apache-tomcat>/work`
2. Delete all files under the work directory.

7.10.4 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).

7.10.5 Updating the XML/A Connections (Optional)

When you upgrade your sample data to 3.5, 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 35.

8 UPGRADE NOTES FOR JASPERSERVER 2.1

If you are upgrading from an older JasperServer version such as 2.1 then you must run additional database upgrade scripts.

In the upgrade steps specified in the previous section, before you run the steps in section 7.7, “Starting JasperServer 3.5,” on page 40, you should first run the steps below:

This chapter contains the following sections:

- **Upgrading a MySQL Database**
- **Issue Exporting and Importing from 2.0 to 3.0**
- **Issue Exporting and Importing from 2.0 to 3.0**

8.1 Upgrading a MySQL Database

To upgrade the JasperServer database:

1. `cd <js-install-3.0>/scripts/upgrade`
2. `mysql> use jasperserver;`
3. `mysql> source upgrade-mysql-2.1.0-3.0.0.sql;`

To upgrade the sample database:

1. `cd <js-install-3.0>/scripts/upgrade/sample`
2. `mysql> use foodmart;`
3. `mysql> source upgrade-mysql-sample-foodmart-2.1.0.sql;`

8.2 Issue Exporting and Importing from 2.0 to 3.0

If you start with JasperServer 2.0 and go directly to 3.0 (or 2.1), there is a bug regarding `SecureMondrianConnections` when you run the import operation.

The underlying problem is that Analysis data security was added to the 2.1 release and a new resource object `SecureMondrianConnect` was created to handle this functionality. When the old, non-secure `MondrianConnection` object is attempted to be upgraded to the 2.1 `SecureMondrianConnection` object the operation fails.

For the 3.0 release, a work around has been added to help with this problem.

There are two ways to work around this problem.

8.2.1 Deleting Any 2.0 MondrianConnection Objects

You can delete any `MondrianConnections` that are in your 2.0 system. If you are not using `MondrianConnections` as part of your business data, you may still have these resources as part of the JasperServer sample data. These are typically found in the repository under the `/analysis/connections` folder.

Then, you can continue your upgrade procedures as normal.

8.2.2 Running a Special Operation to Convert

Do the following steps:

1. Export your 2.0 repository data.
2. Go to the location of the newly installed JasperServer 3.0:
`cd <js-install-3.0>/scripts`
3. Make sure your 3.0 import-export is configured for your database settings.
4. Open the following ant file for editing:
`build-convert-export.xml`.
5. Look for the `<import-dir>` property and set it to the full path of the 2.0 export directory that you just created, for example:
`<property name="import-dir" value="C:\Program Files\jasperserver-2.0\my-2.0-resources" />`
6. Then, run the ant script that will convert your 2.0 export file so that the `SecureMondrianConnection` issue is fixed, and will import the contents of the converted export file into your 3.0 system:
`ant -f build-convert-export.xml convert-repository-to-3.0-from-export`

9 CHANGING PASSWORD ENCRYPTION IN JASPERSERVER

When password encryption is enabled, passwords in the database are stored as cipher text. System administrators can choose the algorithm that JasperServer will use, as well as specify the salt key used to initiate the encryption algorithm.



For JasperServer 3.5, password encryption is turned on by default.

This section describes the procedure to enable password encryption if you have a JasperServer instance without encryption turned on.

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 In to JasperServer](#)

9.1 Backing Up Your JasperServer Database

As a precaution, you must backup your jasperserver database in case there is any problem while enabling encryption.

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

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

You can use the `jasperdb` user or the `root` user to carry out this operation. For more information on running the `mysqldump` command, refer to section [7.10.1, “Using mysqldump for Database Backup,” on page 41](#).

For other databases, refer to your product documentation for details.

9.2 Stopping Your Application Server

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

9.3 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 [10, “Configuring the Import-Export Utilities,” on page 49](#).

9.4 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:

`<apache-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>
```



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.

9.5 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 9.1, “[Backing Up Your JasperServer Database,” on page 45](#). 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 9.4, “[Specifying Encryption Settings in the JasperServer WAR,” on page 46](#).

9.6 Recreating the JasperServer Database

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

9.6.1 Dropping and Recreating in MySQL

1. Change directory to `<js-install>/scripts/mysql`.
2. Login to 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 jasperserverCreate-mysql.ddl;
```

9.6.2 Dropping and Recreating in PostgreSQL

1. Change directory to `<js-install>/scripts/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 jasperserverCreate-postgresql.ddl
```

9.7 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, refer to [10, “Configuring the Import-Export Utilities,” on page 49](#).

9.8 Starting the Application Server

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

9.9 Logging In to JasperServer

You can now login to JasperServer.

Enter your username 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.

10 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.

The JasperServer buildomatic scripts have an import-export setup that can be autoconfigured for your database settings. For more information running import-export from buildomatic, see section 5.12, “Running the Import and Export Utilities,” on page 27

This chapter contains the following sections:

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

10.1 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

10.2 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.

If your repository contains international characters, you may need to perform additional configuration for the import and export utilities. See section [A.8, “JBoss Modifications,” on page 54](#).

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=jasperdb 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>

10.3 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.

If JDBC driver JAR files are put in the following directory, they are automatically included on the classpath when the import or export command is run using the shell scripts:

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

Drivers can be found here:

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

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

10.4 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, “JBoss Modifications,” on page 54](#).

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**
- **JBoss Modifications**
- **PostgreSQL: Job Scheduling Error**
- **Error Running Buildomatic Scripts**

A.1 Installer Freezes

If you run the JasperServer installer on any platform and the installer “freezes” or “hangs,” 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\<username>\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, username, 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` (`jasperadmin` in 2.1 and earlier).

A.2.2 Configuration File Locations

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

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

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:
`<apache-tomcat>/conf/Catalina/Localhost/jasperserver.xml`

This database configuration in `<apache-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 `<apache-tomcat>/webapps/jasperserver.war` file. This causes the `jasperserver` directory to be used.
2. Deleting your `<apache-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-3.0/mysql/bin/mysqld_safe --port=3306 \
--socket=/home/devuser/jasperserver-3.0/mysql/tmp/mysql.sock ...
```

3. Then run a command similar to the following:

```
/home/devuser/jasperserver-3.0/mysql/bin/mysql -u jasperdb -p \
--socket=/home/devuser/jasperserver-3.0/mysql/tmp/mysql.sock
```

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 52](#).

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 [w, “Setting Java JVM Options,” on page 21](#). It is recommended that you add `-Xms128m -Xmx512m` to your `JAVA_OPTS` setting, but you may increase that to `-Xms512m -Xmx1024m` if memory errors persist.

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 JBoss Modifications

A.8.1 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
```

There are two workarounds for this problem. One would be to remove the JBoss web services service archive. The other is to set special Java JVM options.

A.8.1.1 Removing the JBoss Web Services

If you are not using JBoss web services, you can simply remove the JBoss web service archive to remove the possibility of any conflicts. To do this, remove the following service archive:

```
<jboss>/server/default/deploy/jbossws.sar
```

A.8.1.2 Setting Java JVM Options

If you are using Java 1.5 or Java 1.6, you can prevent the web services class conflict by setting special Java JVM Options as described in section [5.8, “Setting Java JVM Options,” on page 25](#).

A.8.2 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.8.3 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-<ver>.jar
```

A.9 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:

```
<apache-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.10 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.10.1 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 website refers to the download as either “Java SE (JDK)” or “Java SE Development Kit (JDK).”

A.10.2 Forgot to Copy the File ant-contrib-<ver>.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

Cause: The name is undefined.

Action: Check the spelling.

Action: Check that any custom tasks/types have been declared.

Action: Check that any <presetdef>/<macrodef> declarations have taken place.

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

GLOSSARY

Ad Hoc Editor

JasperServer's integrated report designer. Starting from a collection of fields predefined in a Topic or selected from a Domain, the Ad Hoc Editor lets you drag and drop report elements to draft, preview, and finalize reports. Like JRXML reports, Ad Hoc reports can be run, printed, and scheduled within JasperServer. In addition, Ad Hoc reports may be reopened in the Ad Hoc Editor, further modified, and saved.

Analysis Client Connection

A definition for retrieving an analysis view. An analysis client connection is either a direct Java connection (Mondrian connection) or an XML-based API connection (XMLA connection).

Analysis Schema

A metadata definition of a multidimensional database. In JasperAnalysis, schemas are stored in the repository as XML file resources.

Analysis View

A view of multidimensional data that is based on an analysis client connection and an MDX query. It is the entry point to analysis operations, such as slice and dice, drill down, and drill through.

Calculated Field

In a Domain, a field whose value is calculated from a user-written formula that may include any number of fields, operators, and constants. A calculated field is defined in the Domain Designer dialog, and it becomes one of the items to which the Domain's security file and locale bundles can apply.

CRM

Customer Relationship Management. The practice of managing every facet of a company's interactions with its clientele. CRM applications help businesses track and support their customers.

CrossJoin

An MDX function that combines two or more dimensions into a single axis (column or row).

Cube

The basis of most analysis applications, a cube is a data structure that contains three or more dimensions that categorize the cube's quantitative data. When you navigate the data displayed in an analysis view, you are exploring a cube.

Custom Field

In the Ad Hoc Editor, a field that is created through menu items as a simple function of one or two available fields, including other custom fields. When a custom field becomes too complex or needs to be used in many reports, it is best to define it as a calculated field in a Domain.

Dashboard

A collection of reports, input controls, graphics, labels, and web content displayed in a single, integrated view. Dashboards often present a high level view of your data, but input controls can parameterize the data to display. For example, you can narrow down the data to a specific date range. Embedded web content, such as other web-based applications or maps, make dashboards more interactive and functional.

Derived Table

In a Domain, a derived table is defined by an additional query whose result becomes another set of items available in the Domain. For example, with a JDBC data source, you can write an SQL query that includes complex functions for selecting data. You can use the items in a derived table for other operations on the Domain, such as joining tables, defining a calculated field, or filtering. The items in a derived table can also be referenced in the Domain's security file and locale bundles.

Data Policy

In JasperServer, a setting that determines how JasperServer should process and cache data used by Ad Hoc reports. Select your data policies by clicking **Manage > Ad Hoc Options**.

Data Source

Defines the connection properties that JasperServer needs to access data. JasperServer transmits queries to data sources and obtains datasets in return for use in filling reports and previewing Ad Hoc reports. JasperServer supports JDBC, JNDI, and Bean data sources; custom data sources can be defined as well.

Dataset

A collection of data arranged in columns and rows. Datasets are equivalent to relational results sets and the `JRDataSource` type in JasperReports.

Datatype

In JasperServer, a datatype is used to characterize a value entered through an input control. A datatype must be of type text, number, date, or date-time. It can include constraints on the value of the input, for example maximum and minimum values. As such, a JasperServer datatype is more structured than a datatype in most programming languages.

Denormalize

A process for creating table joins that speeds up data retrieval at the cost of having duplicate row values between some columns.

Dice

An OLAP operation to select columns.

Dimension

A categorization of the data in a cube. For example, a cube that stores data about sales figures might include dimensions such as time, product, region, and customer's industry.

Domain

A virtual view of a data source that presents the data in business terms, allows for localization, and provides data-level security. A Domain is not a view of the database in relational terms, but it implements the same functionality within JasperServer. The design of a Domain specifies tables in the database, join clauses, calculated fields, display names, and default properties, all of which define items and sets of items for creating Ad Hoc reports.

Domain Topic

A Topic that is created from a Domain by the Choose Ad Hoc Data wizard. A Domain Topic is based on the data source and items in a Domain, but it allows further filtering, user input, and selection of items. Unlike a JRXML-based Topic, a Domain Topic can be edited in JasperServer by users with the appropriate permissions.

Drill

To click on an element of an analysis view to change the data that is displayed:

- Drill down. An OLAP operation that exposes more detailed information down the hierarchy levels by delving deeper into the hierarchy and updating the contents of the navigation table.
- Drill through. An OLAP operation that displays detailed transactional data for a given aggregate measure. Click a fact to open a new table beneath the main navigation table; the new table displays the low-level data that constitutes the data that was clicked.
- Drill up. An OLAP operation for returning the parent hierarchy level to view to summary information.

Eclipse

An open source Integrated Development Environment (IDE) for Java and other programming languages, such as C/C++.

ETL

Extract, Transform, Load. A process that retrieves data from transactional systems, and filters and aggregates the data to create a multidimensional database.

Fact

The specific value or aggregate value of a measure for a particular member of a dimension. Facts are typically numeric.

Field

A field is equivalent to a column in the relational database model. Fields originate in the structure of the data source, but you may define calculated fields in a Domain or custom fields in the Ad Hoc Editor. Any type of field, along with its display name and default formatting properties, is called an item and may be used in the Ad Hoc editor.

Frame

A dashboard element that displays reports or custom URLs. Frames can be mapped to input controls if their content can accept parameters.

Group

In a report, a group is a set of data rows that have an identical value in a designated field.

- In a table, the value appears in a header and footer around the rows of the group, while the other fields appear as columns.
- In a chart, the field chosen to define the group becomes the independent variable on the X axis, while the other fields of each group are used to compute the dependent value on the Y axis.

Hierarchy Level

In analysis, a member of a dimension containing a group of members.

Input Control

A button, check box, drop-down list, text field, or calendar icon that allows users to enter a value when running a report or viewing a dashboard that accepts input parameters. For JRXML reports, input controls and their associated datatypes must be defined as repository objects and explicitly associated with the report. For Domain-based reports that prompt for filter values, the input controls are defined internally. When either type of report is used in a dashboard, its input controls are available to be added as special content.

Item

When designing a Domain or creating a Topic based on a Domain, an item is the representation of a database field or a calculated field along with its display name and formatting properties defined in the Domain. Items can be grouped in sets and are available for use in the creation of Ad Hoc reports.

JavaBean

A reusable Java component that can be dropped into an application container to provide standard functionality.

JDBC

Java Database Connectivity. A standard interface that Java applications use to access databases.

JNDI

Java Naming and Directory Interface. A standard interface that Java applications use to access naming and directory services.

Join Tree

In Domains, a collection of joined tables from the actual data source. A join is the relational operation that associates the rows of one table with the rows of another table based on a common value in given field of each table. Only the fields in a same join tree or calculated from the fields in a same join tree may appear together in a report.

JPivot

An open source graphical user interface for OLAP operations. For more information, visit <http://jpivot.sourceforge.net/>.

MDX

Multidimensional Expression Language. A language for querying multidimensional objects, such as OLAP (On Line Analytical Processing) cubes, and returning cube data for analytical processing. An MDX query is the query that determines the data displayed in an analysis view.

Measure

Depending on the context:

- In a report, a formula that calculates the values displayed in a table's columns, a crosstab's data values, or a chart's dependent variable (such as the slices in a pie).
- In an analysis view, a formula that calculates the facts that constitute the quantitative data in a cube.

Mondrian

A Java-based, open source multidimensional database application.

Mondrian Connection

An analysis client connection that consists of an analysis schema and a data source used to populate an analysis view.

Mondrian Schema Editor

An open source Eclipse plugin for creating Mondrian analysis schemas.

Mondrian XMLA Source

A server-side XMLA source definition of a remote client-side XMLA connection used to populate an analysis view using the XMLA standard.

MySQL

An open source relational database management system. For information, visit <http://www.mysql.com/>.

Navigation Table

The main table in an analysis view that displays measures and dimensions as columns and rows.

Object

In JasperServer, anything residing in the repository, such as an image, file, font, data source, topic, domain, report element, saved report, report output, dashboard, or analysis view. The folders that contain repository objects are also objects. Administrators set user and role-based access privileges on repository objects to establish a security policy.

OLAP

On Line Analytical Processing. Provides multidimensional views of data that help users analyze current and past performance and model future scenarios.

Organization

A set of users that share resources and repository objects in JasperServer. An organization has its own user accounts, roles, and root folder in the repository to securely isolate it from other organizations that may be hosted on the same instance of JasperServer.

Organization Admin

Also called the organization administrator. A user in an organization with the privileges to manage the organization's user accounts and roles, repository permissions, and repository content. An organization admin can also create sub-organizations and manage all of their accounts, roles, and repository objects. The default organization admin in each organization is the `jasperadmin` account.


Outlier

A fact that seems incongruous when compared to other member's facts. For example, a very low sales figure or a very high number of helpdesk tickets. Such outliers may indicate a problem (or an important achievement) in your business. JasperAnalysis excels at revealing outliers.

Parameter

Named values that are passed to the engine at report-filling time to control the data returned or the appearance and formatting of the report. A report parameter is defined by its name and type. In JasperServer, parameters can be mapped to input controls that users can interact with.

Pivot

To rotate a crosstab such that its row groups become column groups and its column groups become rows. In the Ad Hoc Editor, pivot the crosstab by clicking .

Pivot Table

A table with two physical dimensions (for example, X and Y axis) for organizing information containing more than two logical dimensions (for example, PRODUCT, CUSTOMER, TIME, and LOCATION), such that each physical dimension is capable of representing one or more logical dimensions, where the values described by the dimensions are aggregated using a function such as SUM.

Pivot tables are used in JasperAnalysis.

Properties

Settings associated with an object. The settings determine certain features of the object, such as its color and label. Properties are normally editable. In Java, properties can be set in files listing objects and their settings.

Repository

The tree structure of folders that contain all saved reports, dashboards, analysis views, and resources. Users access the repository through the JasperServer web interface or through iReport. Applications can access the repository through the web service API. Administrators use the import and export utilities to back up the repository contents.

Role

A security feature of JasperServer. Administrators create named roles, assign them to user accounts, and then set access permissions to repository objects based on those roles. JasperServer also makes certain functionality available to users based on their roles, which determines certain menu options displayed to those users.

Schema

A logical model that determines how data is stored. For example, the schema in a relational database is a description of the relationships between tables, views, and indexes. In JasperAnalysis, an OLAP schema is the logical model of the data that appears in an analysis view; they are uploaded to the repository as resources. For Domains, schemas are represented in XML design files.

Set

In Domains and Domain Topics, a named collection of items grouped together for ease of use in the Ad Hoc Editor. A set can be based on the fields in a table or entirely defined by the Domain creator, but all items in a set must originate in the same join tree. The order of items in a set is preserved.

Slice

An OLAP operation for filtering data rows.

SQL

Structured Query Language. A standard language used to access and manipulate data and schemas in a relational database.

System Admin

Also called the system administrator. A user who has unlimited access to manage all organizations, users, roles, repository permissions, and repository objects across the entire JasperServer instance. The system admin can create root-level organizations and manage all server settings. The default system admin is the `superuser` account.

Topic

A JRXML file created externally and uploaded to JasperServer as a basis for Ad Hoc reports. Topics are created by business analysts to specify a data source and a list of fields with which business users can create reports in the Ad Hoc Editor. Topics are stored in the Ad Hoc Components folder of the repository and displayed when a user launches the Ad Hoc Editor.

Transactional Data

Data that describe measurable aspects of an event, such as a retail transaction, relevant to your business. Transactional data are often stored in relational databases, with one row for each event and a table column or field for each measure.

User

Depending on the context:

- A person who interacts with JasperServer to fulfill a goal. There are generally three categories of users: administrators who install and configure JasperServer, database experts or business analysts who create data sources and Domains, and business users who create and view reports and dashboards.
- A user account created for a specific person or purpose. The account associates the login name with user's full name, password, and email address. Roles are assigned to user accounts to determine access to objects in the repository.

WCF

Web Component Framework. A low-level GUI component of JPivot. For more information, see <http://jpivot.sourceforge.net/wcf/index.html>.

XML

eXtensible Markup language. A standard for defining, transferring, and interpreting data for use across any number of XML-enabled applications.

XML/A

XML for Analysis. An XML standard that uses Simple Object Access protocol (SOAP) to access remote data sources. For more information, see <http://www.xmla.org/>

XML/A Connection

A type of analysis client connection that consists of Simple Object Access Protocol (SOAP) definitions used to populate an analysis view.