JasperReports® Server

Upgrade Guide

Version 9.0.0 | January 2024

Contents

Introduction

JasperReports® Server builds on JasperReports® Library 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. The products utilize common metadata and provide shared services, such as security, a repository, and scheduling. The server exposes comprehensive public interfaces. This enables seamless integration with other applications and the capability to add custom functionality with ease.

This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you are licensed to use, or to upgrade your license, contact Jaspersoft.

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

- Easily create reports based on views designed in an intuitive, web-based, drag and drop Ad Hoc Editor.
- Efficiently and securely manage many reports.
- Interact with reports, including sorting, changing formatting, entering parameters, and drilling on data.
- Schedule reports for distribution through email and storage in the repository.
- Arrange reports and web content to create appealing, data-rich Jaspersoft Dashboards that quickly convey business trends.

For users interested in multi-dimensional modeling, we offer Jaspersoft® OLAP, which runs as part of the server.

While the Ad Hoc Editor lets users create simple reports, more complex reports can be created outside of the server. You can either use Jaspersoft® Studio or manually write JRXML code to create a report that can be run in the server. We recommend that you use Jaspersoft Studio unless you have a thorough understanding of the JasperReports file structure.

You can use the following sources of information to learn about JasperReports Server:

- Our core documentation describes how to install, administer, and use JasperReports Server and Jaspersoft Studio. Core documentation is available in PDF format on the Product Documentation website. You can also access PDF and HTML versions of these guides from the Documentation section of the Jaspersoft Community website.
- Our Ultimate Guides document advanced features and configuration. They also include best practice recommendations and numerous examples. You can access PDF and HTML versions of these guides online from the Documentation section of the Jaspersoft Community website.
- The Jaspersoft Community site covers topics for:
    - Developers
    - System administrators
    - Business users
    - Data integration users
- Our free samples, which are installed with JasperReports Library, Jaspersoft Studio, and JasperReports Server, are available and documented online. Please visit our GitHub repository.
- If you have a subscription to our professional support offerings, please contact our Technical Support team when you have questions or run into difficulties. They're available on the web at https://www.jaspersoft.com/support.

JasperReports Server is a component of both a community project and commercial offerings. Each of these integrates the standard features such as security, scheduling, web services interface, and much more for running and sharing reports. Commercial editions provide additional features for hosting large BI deployments, including:

- Ad Hoc views and reports
- advanced charts
- dashboards
- domains
- auditing
- multi-organization architecture

Server Upgrade Distributions

Distribution Package	Description
Overlay Upgrade zip	Available only with the Commercial version of JasperReports Server. Supports upgrade to 9.0.0 from version 8.0 or later. Supports only the Apache Tomcat application server. Supports all certified repository databases. Supports upgrade and rollback of upgrade changes. Provides assistance with identifying customized files in your environment. Supports Windows, Linux, Mac, and other platforms. File name is: **js-jrs_9.0.0_overlay.zip**
WAR File Distribution Zip	Supports upgrade from version 8.0 or later. Supports all certified application servers. Supports all certified repository databases. Supports Windows, Linux, Mac, and other platforms. File name is: **js-jrs_9.0.0_bin.zip**

Upgrade Paths

Your current version determines your upgrade path:

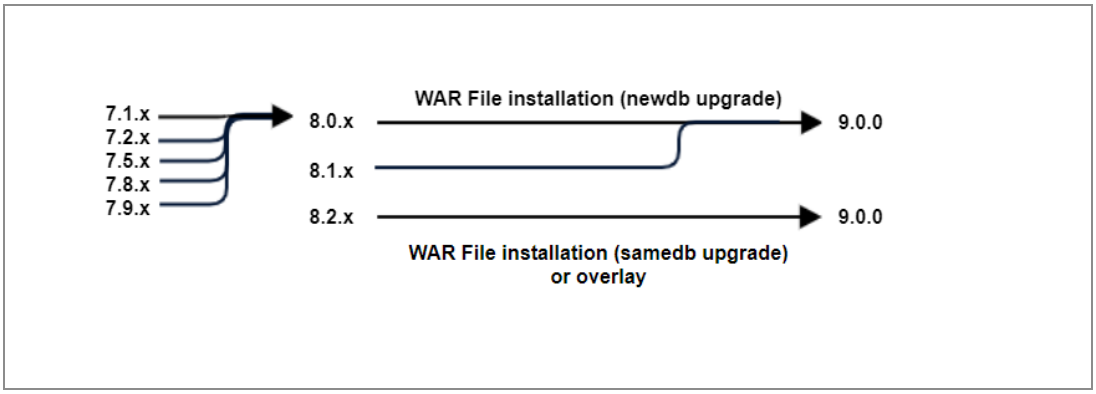

Figure 1: Paths for Upgrading to Version 9.0

If you are upgrading from 8.2, use the instructions in Upgrading from 8.2.x to 9.0. If you are starting from 8.0.x to 8.1.x, use the instructions in Upgrading from 8.0.x - 8.1.x to 9.0.

If you are using the JasperReports Server Commercial edition installed with the WAR file and the Apache Tomcat application server, you can use the overlay upgrade, described in Overlay Upgrade.

For upgrading other versions, see the table below. Versions prior to 6 are no longer supported and must be upgraded to version 6.3 first. You may also need to updgrade in several steps through an intermediate version. In the following table:

- samedb = Follow the steps eqivalent to Upgrading from 8.2.x to 9.0.
- newdb = Follow the steps equivalent to Upgrading from 8.0.x - 8.1.x to 9.0.

To > From:	7.1.x	7.2.x	7.5.x	7.8.x	7.9.x	7.9.x	8.1.x	8.2.0	9.0
7.1.x		samedb	newdb	newdb	newdb	newdb	newdb	newdb	newdb
7.2.x			samedb	newdb	newdb	newdb	newdb	newdb	newdb
7.5.x				samedb	newdb	newdb	newdb	newdb	newdb
7.8.x					newdb samedb	newdb	newdb	newdb	newdb
7.9.x						newdb samedb	newdb	newdb	newdb

To > From:	7.1.x	7.2.x	7.5.x	7.8.x	7.9.x	7.9.x	8.1.x	8.2.0	9.0
8.0.x							newdb samedb	newdb	newdb
8.1.x								newdb samedb	newdb
8.2.x									newdb samedb

About Bundled Apache Ant

Apache Ant version 1.10.10 is bundled with the War File Distribution ZIP and the Overlay Upgrade ZIP. The Ant scripts used for upgrade come with Windows and Linux batch scripts pre-configured to use the bundled version of Apache Ant.

We recommend Apache Ant version 1.10. If you want to run your own version of Apache Ant, version 1.9 or later is required.

The bundled Apache Ant includes an additional jar. This jar (ant-contrib.jar) enables conditional logic in Ant. If you are running your own Ant you should copy the ant-contrib.jar to your <Ant_HOME>/lib folder.

On Linux and Solaris, the Ant commands may not be compatible with all shells. If you get errors, use the `bash` shell explicitly. For more information, see the information on the bash shell in the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

Overlay Upgrade

This chapter describes the overlay process for upgrading to JasperReports Server 9.0.0 and contains the following sections:

Introduction to the Overlay Upgrade

The overlay upgrade procedure is available only for the JasperReports Server Commercial edition installed with the WAR file and only with the Apache Tomcat application server.

- The overlay upgrade supports only the Apache Tomcat application server.
- The overlay upgrade supports only JasperReports Server installations using the WAR file. The binary installer is not supported.

- The overlay upgrade is not possible if you configured custom encryption keys in your previous server.
- Only the certified repository databases are supported.

The overlay upgrade supports upgrading from JasperReports Server versions 8.0 and later to JasperReports Server 9.0.

Although the overlay upgrade does offer a rollback feature, you should always back up your database and application before upgrading.

 This section uses 8.2 to 9.0 upgrade as example.

Upgrade Steps Overview

These are the general steps used in this section:

1. Plan your upgrade.

2. Back up your current JasperReports Server instance.

(The overlay tool automatically takes back up of your war file and ask if you have backed up your database.)

3. Download and unpack the new JasperReports Server overlay upgrade 9.0package zip file.

4. Run the upgrade steps.

The overlay upgrade procedure helps you to identify any modifications or extensions you have made to your JasperReports Server instance.

 It is always best practice to back up your application and database before upgrading.

Plan Your Upgrade

See Planning Your Upgrade to determine if any customizations you've made to your existing version of JasperReports Server are affected by changes to the updated version.

Back Up Your JasperReports Server Instance

First back up your JasperReports Server WAR file and `Jasperserver` database so you can restore them if necessary. Perform these steps from the command line in a Windows or Linux shell.

Back up your JasperReports Server War File

Procedure

1. Create a folder where you can save your `jasperserver-pro` war file, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.
2. Copy `<tomcat>/webapps/jasperserver-pro` to `<path>/JS_BACKUP`.

Back up your Jasperserver Database

Procedure

1. Create a folder (if you did not do so in the step above) where you can save your `Jasperserver` database, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.
2. Run the following commands for PostgreSQL:
    - PostgreSQL

```
cd <path>/JS_BACKUP
pg_dump --username=postgres  jasperserver  >  js-db-dump.sql
```

Back up your JasperReports Server Keystore

1. Create a folder (if you did not do so already) where you can save your server's keystore, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.

2. As the user who originally installed the server, copy `$HOME/.jrsks` and `$HOME/.jrsksp` to `<path>/JS_BACKUP`. Remember that these files contain sensitive keys for your data, so they must always be transmitted and stored securely.

Unpack the Overlay Upgrade Package

The overlay upgrade package comes in a file named:`js-jrs_9.0.0_overlay.zip`.

1. Download the overlay upgrade package from Jaspersoft Technical Support (https://www.jaspersoft.com/support) or contact your sales representative.

2. Extract all files from `js-jrs_9.0.0_overlay.zip`. Create or choose a destination folder, such as `C:\JS_OVERLAY`on Windows,`/home/<user>/JS_OVERLAY` on Linux, or `/Users/<user>/JS_OVERLAY`on Mac.

The overlay upgrade uses paths that exceed the 260-character limit on Windows. To extract the package,Enable NTFS long paths (Windows 10 only) or use a third-party file archive such as 7-Zip.

3. The overlay upgrade package unpacks into a folder named:

`overlay`

This document refers to this folder location as:

`<overlay-folder>`

Check for JDBC Driver (Oracle, SQL Server, DB2)

JasperReports Serveruses the JDBC drivers for the Oracle, SQL Server, and DB2 commercial databases. If you want to use a different JDBC driver, you need to copy it to the correct location. If you use Oracle or DB2, you must also use your existing version of the db.template.properties file. See Working With JDBC Drivers for more information.

Configure the Properties in the default_ master.properties File

Before running the overlay upgrade, copy the `default_master.properties` file from the existing JRS buildomatic directory to the Overlay buildomatic directory. Configure the properties specific to the installation type:

- For Compact upgrade: No additional configuration is required in the `default_ master.properties` file.
- For Split upgrade: Edit the `default_master.properties` file to configure the settings as described inAdditional Buildomatic Configuration for Split Installation Upgrade.

Run the Overlay Upgrade

The overlay upgrade works only with the Tomcat application server. It supports only the certified repository databases. You can perform the overlay upgrade whether you have local customizations or not.

Procedure

1. Stop the Tomcat application server.
2. Make sure that your database is running.
3. Make sure that the user running the overlay commands is the same user that installed the server.
4. Run the following commands:

   ```
   cd <overlay-folder>
   ```

   Windows:`overlay install`

   Linux:`./overlay install`
5. You are prompted to specify a path to a working folder:
   - You can accept the default or specify an alternate folder.
   - Press `enter` to accept the default `../overlayWorkspace`.
6. You are prompted to take a back up of your `Jasperserver` database. If you have already backed up your database, choose "y" to continue. If you have not yet backed up your database, choose "n" to exit the overlay and create a backup.
7. You are prompted to shut down your Tomcat instance:
   - You can stop Tomcat now if you have not already done so.
   - Choose "y" for `yes` to continue.
8. If you are prompted to create a keystore, this means that the server's original keystore was not found in the user's home directory. Proceed with caution:
   - In general, it is recommended to exit the overlay procedure and make sure that the keystore is in the proper location, then rerun the overlay as described below.
   - Alternatively, update the current location of the keystore in the

`keystore.init.properties` file at the following locations:

- .../WEB-INF/classes/keystore.init.properties
- .../buildomatic/keystore.init.properties
- .../buildomatic/conf_source/iePro/keystore.init.properties

- If you continue and create a keystore, then the overlay proceeds but your repository is corrupted and users are unable to log in. In this case, you need to export manually the server's repository with a custom key, then import the key before importing the repository, as described inEncryption Keys.

9. You are prompted to specify a path to your `master.properties` file:

   Specify the path for your `default_master.properties` file, which is present in the Overlay buildomatic directory.

10. For final verification, the overlay prompts you for the path to your application server:
    - If you have not moved it, it is located in the path to: `<tomcat>`
    - Press `enter` to accept the default if it is correct.
11. The overlay begins updating your system:
    - Your `jasperserver-pro` war file is automatically backed up.
    - Potential customizations in your environment is analyzed.

    You are prompted to review the report on customizations if you choose to:
    - Choose "y" for `yes` to continue with the upgrade.
    - The`Jasperserver`database will be upgraded.
    - The `jasperserver-pro` war file will be upgraded.
    - The core data resources will be upgraded in the `Jasperserver` repository database.

When the overlay upgrade has finished, start Tomcat, and log in to test the upgraded JasperReports Server.

If the upgrade was successful, you see BUILD SUCCESSFUL on the command line.

For the Split upgrade, after the upgrade is done, to transfer the data (Audit, Access, and Log monitoring data) to the `audit` database from the `Jasperserver` database, run the following command:

- Windows: `transfer-audit-data.bat`

- Linux and Mac OSX: `./transfer-audit-data.sh`

The data is transferred to the `audit` database and the tables are deleted from the `Jasperserver` database. Rerun the command if there is any interruption in the data transfer process, it resumes the transfer process from where it was interrupted in the previous run.

Rerun the Overlay Upgrade

If you exit the `overlay install` for any reason, you can rerun the overlay by simply running the same command:

```
overlay install
```

By default, the overlay runs in resume mode (`resumeMode=true`). This means that your answers to previous prompts are remembered.

If you want to rerun the `overlay` "from scratch", run the following command:

```
overlay install -DresumeMode=false
```

For more information on the `overlay` options, run:

overlay help

Rollback Procedure

If you encounter an error with the overlay upgrade, use the following rollback procedure:

1. Stop Tomcat.
2. Run the following command:

   ```
   overlay rollback
   ```

3. Specify the path to the working folder:

   The default is `../overlayWorkspace`

4. The tool asks if you have rolled back your JasperReports Server database:

   The default is no

   

   You are manually required to restore your database.

5. When the tool has finished, restore your database (see below), start Tomcat, and test JasperReports Server.

To restore your JasperReports Server Database

1. Go to the directory location where you saved the backup of your `Jasperserver` database.

For example, `C:\JS_BACKUP` or `/opt/JS_BACKUP`.

2. Run the following commands for PostgreSQL:

```
cd /opt/JS_BACKUP
pg_restore --username=postgres  jasperserver  <  js-db-dump.sql
```

To restore your JasperReports Server Keystore

1. Go to the directory location where you saved the backup of your JasperReports Serverkeystore.

For example, `C:\JS_BACKUP` or `/opt/JS_BACKUP`.

2. Copy the .jrsks and .jrsksp files from the `C:\JS_BACKUP` folder back to the `$HOME` folder, where they were originally backed up from.

If the backed up .jrsks and .jrsksp files are not copied from the `C:\JS_BACKUP` folder back to the `$HOME` folder, the JasperReports Server login page does not open and gives an error.

Starting and Logging into JasperReports Server 9.0

Start your application server. Your database should already be running.

Clearing Your Browser Cache

Before you log in, make sure you and your end users clear the browser cache. JavaScript files, which enable the UI elements of JasperReports Server, are typically cached by the browser. Clear the cache to ensure that the newer files are used.

Logging into JasperReports Server

Log in using the following URL, user IDs, and passwords:

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

User ID	Password	Description
`superuser`	`<your-password>`	System-wide administrator
`jasperadmin`	`<your-password>`	Administrator for the default organization

Your JasperReports Server instance has now been upgraded to 9.0. If you have startup or login problems, refer to the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

Additional Tasks to Complete the Upgrade

Installing JasperReports Server automatically generates encryption keys that reside on the file system. These keys are stored in a dedicated Jaspersoft keystore. Make sure that this keystore is properly secured and backed up, as described in the *JasperReports Server Security Guide*.

Perform these tasks with the application server shutdown.

Handling JasperReports Server Customizations

If you made modifications to the original JasperReports Server application, you need to copy manually configuration changes, like client-specific security classes or LDAP server configurations, from your previous environment and integrate them with your upgraded environment. These configurations are typically found in the files at the `WEB-INF/` location, for example, applicationContext-*.xml, *.properties, *.js, *.jar, and so on.

Clearing the Application Server Work Folder

Application servers have work folders where JasperReports Server files are compiled and cached and other objects are stored. When you update the WAR file or license, the buildomatic `deploy-webapp-pro` target should automatically clear the application server's `work` directory, but it is a good practice to double-check. A permission problem, or some other problem, could prevent the clearing of the work folder.

To clear the work folder in Tomcat

1. Change the directory to `<tomcat>/work`.

2. Delete all the files and folders in this directory.

Clearing the Application Server Temp Folder

JasperReports Server uses caching to speed operations within the application. Caching files are created and stored in the application server, usually in a `temp` folder. Clear this `temp` folder to avoid any post-upgrade conflicts.

To clear the temp folder in Apache Tomcat

1. Change the directory to `<tomcat>/temp`.

2. Delete all the files and folders in this directory.

Clearing the Repository Cache Database Table

In the `Jasperserver` database, compiled JasperReports Library resources are cached in the `JIRepositoryCache` table for increased efficiency at runtime. Because the JasperReports Library JAR is typically updated with each new release, old cached items can get out of

date and cause errors at runtime. If you encounter errors that mention a JasperReports Library "local class incompatible", check your repository cache table. In summary, you can clear your `Jasperserver` database cache table as part of this upgrade process whether there are errors or not.

To clear the repository cache database table manually, run a SQL command similar to the one shown below:

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

Running Overlay Upgrade a Second Time

If you run the overlay upgrade a second time, the overlay logic asks if you want to resume the last run of the overlay, so that your previous answers to questions are remembered and reused.

The overlay procedure asks:

```
"We have detected that the overlay install was already run. Do you want
to resume your last run? The default is 'y' ([y], n):"
```

- Choose "y" for yes if you do not want to change any information previously given to the overlay.
- Choose "n" for no if you would like to enter new or different information.

One reason for entering "n" for no would be if you did not give a valid path to your default_master.properties file the first time you run the overlay.

Upgrading from 8.2.x to 9.0

This chapter describes the recommended procedure for upgrading to JasperReports Server 9.0 from version 8.2.x. The examples show you how to upgrade using the js-upgrade shell script.

This chapter contains the following sections:

Upgrade Steps Overview

These are the general steps used in this section:

1. Identify your customizations.

2. Back up your current JasperReports Server instance.

3. Download and set up the new 9.0 JasperReports Server WAR file distribution zip.

4. Run the js-upgrade script as described in Upgrading to JasperReports Server 9.0.

If your current instance of JasperReports Server has modifications or extensions, monitor these and reintegrate them into your 9.0 instance after upgrading.

Upgrading with Customizations

If your current instance of JasperReports Server has modifications or extensions, keep track of these and re-integrate them into your 9.0 instance after upgrading. See Planning Your Upgrade to determine if any customizations you've made to your existing version of JasperReports Server are affected by changes to the updated version.

Back Up Your JasperReports Server Instance

First back up your JasperReports Server WAR file and `Jasperserver` database so you can restore them if necessary. Perform these steps from the command line in a Windows or Linux shell.

This backup example is for Tomcat with the PostgreSQL or MySQL database. For other databases, consult your DB administration documentation for backup information.

Back up your JasperReports Server War File

Procedure

1. Create a folder where you can save your `jasperserver-pro` war file, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.
2. Copy `<tomcat>/webapps/jasperserver-pro` to `<path>/JS_BACKUP`.

Back up your Jasperserver Database

Procedure

1. Create a folder (if you did not do so in the step above) where you can save your `Jasperserver` database, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.
2. Run the following commands for PostgreSQL or MySQL:
   - PostgreSQL

     ```
     cd <path>/JS_BACKUP
     pg_dump --username=postgres  jasperserver  >  js-db-dump.sql
     ```

   - MySQL

```
cd <path>/JS_BACKUP
```

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

For MySQL, If you receive an error about packet size, see the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

Back up your JasperReports Server Keystore

1. Create a folder (if you did not do so already) where you can save your server's keystore, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.

2. As the user who originally installed the server, copy `$HOME/.jrsks` and `$HOME/.jrsksp` to `<path>/JS_BACKUP`. Remember that these files contain sensitive keys for your data, so they must always be transmitted and stored securely.

Preparing the JasperReports Server 9.0 WAR File Distribution

Use the buildomatic `js-upgrade` scripts included in the 9.0 WAR file distribution ZIP release package to carry out the upgrade. The WAR file distribution comes in a compressed ZIP file named `js-jrs_9.0.0_bin.zip`.

Follow these steps to obtain and unpack the WAR file distribution ZIP file:

1. Download the WAR file distribution from Jaspersoft Technical Support (https://www.jaspersoft.com/support) or contact your sales representative.

2. Extract all files from `js-jrs_9.0.0_bin.zip`. Choose a destination, such as a `C:\Jaspersoft` folder on Windows, `/home/<user>` on Linux, or `/Users/<user>` on Mac.

After you unpack the WAR File Distribution, the resulting location will be known as:

`<js-install-9.0>`

Configuring Buildomatic for Your Database and Application Server

This upgrade procedure uses the `js-upgrade-samedb` shell script.

For Unix, the bash shell is required for the js-upgrade scripts. If you are installing to a non-Linux Unix platform such as IBM AIX, FreeBSD or Solaris, you need to download and install the bash shell. See the Troubleshooting appendix of the *JasperReports Server Installation Guide* for more information.

This section shows example configurations for the PostgreSQL, MySQL, and Oracle databases. Other databases are similar.

Example Buildomatic Configuration

The `default_master.properties` file handles the upgrade configuration. We provide a sample configuration file for each database. You must specify your database credentials and application server location, and rename the file to `default_master.properties`.

PostgreSQL Example

To configure `default_master.properties` for PostgreSQL:

1. Locate the `postgresql_master.properties` sample configuration file.

Database	Master Properties File
PostgreSQL	<js-install-9.0>/buildomatic/sample_conf/postgresql_master.properties

2. Copy the file to `<js-install-9.0>/buildomatic`.

3. Rename the file `default_master.properties`.

4. Edit `default_master.properties` for your database and application server.

Database	Sample Property Values

PostgreSQL	`appServerType=tomcat (or wildfly, etc.)`
	`appServerDir=c:\\Apache Software Foundation\\Tomcat 9.0 (for example)`
	`dbUsername=postgres`
	`dbPassword=postgres`
	`dbHost=localhost`

For the Split upgrade, configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.

MySQL Example

To configure `default_master.properties` for MySQL:

1. Locate the `mysql_master.properties` sample configuration file:

Database	Master Properties File
MySQL	<js-install-9.0>/buildomatic/sample_conf/mysql_master.properties

2. Copy the file to `<js-install-9.0>/buildomatic`.

3. Rename the file `default_master.properties`.

4. Edit `default_master.properties` for your database and application server.

Database	Sample Property Values
MySQL	`appServerType=tomcat (or wildfly, etc.)`

Database	Sample Property Values

	`appServerDir=c:\\Apache Software Foundation\\Tomcat 9.0 (for example)`
	`dbUsername=root`
	`dbPassword=password`
	`dbHost=localhost`

For the Split upgrade, configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.

Oracle Example

To configure `default_master.properties` for Oracle:

1. Locate the `oracle_master.properties` sample configuration file:

Database	Master Properties File
Oracle	<js-install-9.0>/buildomatic/sample_conf/oracle_master.properties

2. Copy the file to `<js-install-9.0>/buildomatic`.

3. Rename the file to `default_master.properties`.

4. Edit `default_master.properties` for your database and application server.

Database	Sample Property Values

Oracle	`appServerType=tomcat (or wildfly, etc.)`
	`appServerDir=c:\\Apache Software Foundation\\Tomcat 9.0 (for example)`
	`dbUsername=jasperserver`
	`dbPassword=password`
	`sysUsername=system`
	`sysPassword=password`
	`dbHost=localhost`

For the Split upgrade, configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.

Using Vendor's Drivers for Commercial Databases

JasperReports Server doesn't include JDBC drivers for the following commercial databases: Oracle, SQL Server, or DB2. If you want to use one of those databases, you need to obtain one of the available JDBC drivers and copy it into the correct location, and then edit default_master.properties before running the upgrade steps. See Working With JDBC Drivers for more information.

Upgrading to JasperReports Server 9.0

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

Make sure you have taken backup of your `Jasperserver` database before proceeding.

Make sure you have taken backup of your old JasperReports Server WAR file before proceeding.

1. Stop your application server.

2. Start your database server.

3. Make sure that the user running the upgrade commands is the same user that installed the server.

4. Run the following commands:

Commands	Description
`cd <js-install-9.0>/buildomatic`	
`js-upgrade-samedb.bat`	(Windows) Upgrade jasperserver-pro war file, upgrade Jasperserver database to 9.0, add 9.0 repository resources into the database.

Commands	Description
`./js-upgrade-samedb.sh`	(Linux) Upgrade jasperserver-pro war files, upgrade Jasperserver database to 9.0, add 9.0 repository resources into the database.

If you are prompted to create a keystore, this means that the server's original keystore was not found in the user's home directory. Proceed with caution:

- In general, it is recommended to exit the upgrade procedure and make sure that the keystore is in the proper location, then rerun the upgrade.
- If you continue and create a keystore, then the upgrade proceeds but your repository is corrupted and users are unable to log in. In this case, you need to export manually the server's repository with a custom key, then import the key before importing the repository, as described in Encryption Keys.

For the Split upgrade, after the upgrade is done, to transfer the data (Audit, Access, and Log monitoring data) to the `audit` database from the `Jasperserver` database, run the following command:

- Windows: `transfer-audit-data.bat`
- Linux and Mac OSX: `./transfer-audit-data.sh`

The data is transferred to the `audit` database and the tables are deleted from the `Jasperserver` database. Rerun the command if there is any interruption in the data transfer process, it resumes the transfer process from where it was interrupted in the previous run.

js-upgrade Test Mode

Use the `test` option to run the `js-upgrade` script in test mode. For example, on Windows, enter:

```
cd <js-install-9.0>/buildomatic
```

```
js-upgrade-samedb.bat test
```

In test mode, the js-upgrade scripts check your default_master.properties settings and validate your application server location and its ability to connect to your database. Test mode can help you debug issues like an incorrect database password without altering your system.

Output Log Location

The js-upgrade script creates an output log that captures both standard and error output. If problems occur during script execution, or you just want to remember which options you chose, open the output log file located here:

`<js-install-9.0>/buildomatic/logs/js-upgrade-<date>-<number>.log`

Errors

If you encounter errors running the `js-upgrade` script, first look at the output log to see if you can spot the errors. For help, refer to the Troubleshooting appendix of the *JasperReports Server Installation Guide*. The information in this appendix applies to both `js-upgrade` scripts and `js-install` scripts.

If you need to modify values in your `default_master.properties` file, you can simply edit the file. When you run the `js-upgrade` script again, it uses the new values.

Starting and Logging into JasperReports Server 9.0

Start your application server. Your database should already be running.

Clearing Your Browser Cache

Before you log in, make sure you and your end users clear the browser cache. JavaScript files, which enable the UI elements of JasperReports Server, are typically cached by the browser. Clear the cache to ensure that the newer files are used.

Logging into JasperReports Server

Log in using the following URL, user IDs, and passwords:

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

User ID	Password	Description
`superuser`	`<your-password>`	System-wide administrator
`jasperadmin`	`<your-password>`	Administrator for the default organization

Your JasperReports Server instance has now been upgraded to 9.0. If you have startup or login problems, refer to the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

Additional Tasks to Complete the Upgrade

Perform these tasks with the application server shutdown.

Handling JasperReports Server Customizations

If you made modifications to the original JasperReports Server application, you need to copy manually configuration changes, like client-specific security classes or LDAP server configurations, from your previous environment and integrate them with your upgraded environment. These configurations are typically found in the files at the `WEB-INF/` location, for example, applicationContext-*.xml, *.properties, *.js, *.jar, and so on.

Clearing the Application Server Work Folder

Application servers have work folders where JasperReports Server files are compiled and cached and other objects are stored. When you update the WAR file or license, the buildomatic `deploy-webapp-pro` target should automatically clear the application server's `work` directory, but it is a good practice to double-check. A permission problem, or some other problem, could prevent the clearing of the work folder.

To clear the work folder in Tomcat

1. Change the directory to `<tomcat>/work`.

2. Delete all the files and folders in this directory.

Clearing the Application Server Temp Folder

JasperReports Server uses caching to speed operations within the application. Caching files are created and stored in the application server, usually in a `temp` folder. Clear this `temp` folder to avoid any post-upgrade conflicts. Typically, the `temp` folder used by an application server corresponds to the path referenced by the `java.io.tmpdir` Java system property. For Apache Tomcat the `temp` folder is `<tomcat>/temp`.

To clear the temp folder in Apache Tomcat

1. Change the directory to `<tomcat>/temp`.

2. Delete all the files and folders in this directory.

Clearing the Repository Cache Database Table

In the `Jasperserver` database, compiled JasperReports Library resources are cached in the `JIRepositoryCache` table for increased efficiency at runtime. Because the JasperReports Library JAR is typically updated with each new release, old cached items can get out of date and cause errors at runtime. If you encounter errors that mention a JasperReports Library "local class incompatible", check your repository cache table. In summary, you can clear your `Jasperserver` database cache table as part of this upgrade process whether there are errors or not.

To clear the repository cache database table manually, run a SQL command similar to the one shown below:

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

Upgrading from 8.0.x - 8.1.x to 9.0

This chapter describes the recommended procedure for upgrading from the latest version of JasperReports Server 8.0 through 8.1.x to JasperReports Server 9.0. If you are upgrading from version 8.2.x to 9.0, we recommend the procedure in Upgrading from 8.2.x to 9.0.

If you are upgrading from an earlier version of JasperReports Server, you need to go through an intermediate version before upgrading to 9.0. See Upgrading JasperReports Server 6.4.x or Earlier] for more information.

This upgrade procedure uses the JasperReports Server WAR File Distribution ZIP release package and the included buildomatic scripts. Our examples are for upgrading from version 8.0.x.

This chapter contains the following sections:

Upgrade Steps Overview

These are the general steps used in this section:

1. Plan your upgrade.

2. Back up your current JasperReports Server instance.

3. Export your existing repository data. For example, export your 8.0.x data.

4. Download and set up the new 9.0 JasperReports Server WAR file distribution zip.

5. Run the js-upgrade script as described in Upgrading to JasperReports Server 9.0.

Upgrading with Customizations

If your current instance of JasperReports Server has modifications or extensions, keep track of these and re-integrate them into your 9.0 instance after upgrading. See Planning Your Upgrade to determine if any customizations you've made to your existing version of JasperReports Server are affected by changes to the updated version.

Back Up Your JasperReports Server Instance

First back up your JasperReports Server WAR file and `Jasperserver` database so you can restore them if necessary. Perform these steps from the command line in a Windows or Linux shell.

This backup example is for Tomcat with the PostgreSQL or MySQL database. For other databases, consult your DB administration documentation for backup information.

Back up your JasperReports Server War File

Procedure

1. Create a folder where you can save your `jasperserver-pro` war file, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.
2. Copy `<tomcat>/webapps/jasperserver-pro` to `<path>/JS_BACKUP`.

Back up your Jasperserver Database

Procedure

1. Create a folder (if you did not do so in the step above) where you can save your `Jasperserver` database, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.
2. Run the following commands for PostgreSQL or MySQL:
    - PostgreSQL

```
cd <path>/JS_BACKUP
pg_dump --username=postgres  jasperserver  >  js-db-dump.sql
```

- MySQL

```
cd <path>/JS_BACKUP
```

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

For MySQL, If you receive an error about packet size, see the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

Back up your JasperReports Server Keystore

1. Create a folder (if you did not do so already) where you can save your server's keystore, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.

2. As the user who originally installed the server, copy `$HOME/.jrsks` and `$HOME/.jrsksp` to `<path>/JS_BACKUP`. Remember that these files contain sensitive keys for your data, so they must always be transmitted and stored securely.

Exporting Current Repository Data

To export using the `js-export.bat/.sh` script, navigate to the buildomatic folder, for example, `<js-install-8.0.x>/buildomatic`. If you are using the PostgreSQL database, the js-export script should already be configured to run. If you are using a different database,

or you have changed database passwords, you may need to update the `js-export` configuration.

Run the following commands:

1. Navigate to the buildomatic directory:

```
cd <js-install-8.0>/buildomatic
```

2. Run the `js-export` script:

Operating System	Command
Windows:	`js-export.bat --everything --output-zip js-8.0-export.zip`
Linux:	`./js-export.sh --everything --output-zip js-8.0-export.zip`

Note the location of the export file so that you can use it during the 9.0 upgrade process.

Preparing the JasperReports Server 9.0 WAR File Distribution

Use the buildomatic `js-upgrade` scripts included in the 9.0 WAR file distribution ZIP release package to carry out the upgrade. The WAR file distribution comes in a compressed ZIP file named `js-jrs_9.0.0_bin.zip`.

Follow these steps to obtain and unpack the WAR file distribution ZIP file:

1. Download the WAR file distribution from Jaspersoft Technical Support (https://www.jaspersoft.com/support) or contact your sales representative.

2. Extract all files from `js-jrs_9.0.0_bin.zip`. Choose a destination, such as a `C:\Jaspersoft` folder on Windows, `/home/<user>` on Linux, or `/Users/<user>` on Mac.

After you unpack the WAR File Distribution, the resulting location will be known as:

<js-install-9.0>

Configuring Buildomatic for Your Database and Application Server

This upgrade procedure uses the `js-upgrade-newdb` shell script.

For Unix, the bash shell is required for the js-upgrade scripts. If you are installing to a non-Linux Unix platform such as IBM AIX, FreeBSD or Solaris, you need to download and install the bash shell. See the Troubleshooting appendix of the *JasperReports Server Installation Guide* for more information.

This section shows example configurations for the PostgreSQL, MySQL, and Oracle databases. Other databases are similar.

Example Buildomatic Configuration

The `default_master.properties` file handles the upgrade configuration. We provide a sample configuration file for each database. You must specify your database credentials and application server location, and rename the file to `default_master.properties`.

PostgreSQL Example

To configure `default_master.properties` for PostgreSQL:

1. Locate the `postgresql_master.properties` sample configuration file.

Database	Master Properties File
PostgreSQL	<js-install-9.0>/buildomatic/sample_conf/postgresql_master.properties

2. Copy the file to `<js-install-9.0>/buildomatic`.

3. Rename the file `default_master.properties`.

4. Edit `default_master.properties` for your database and application server.

Database	Sample Property Values

PostgreSQL	`appServerType=tomcat (or wildfly, etc.)`
	`appServerDir=c:\\Apache Software Foundation\\Tomcat 9.0 (for example)`
	`dbUsername=postgres`
	`dbPassword=postgres`
	`dbHost=localhost`

For the Split upgrade, configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.

MySQL Example

To configure `default_master.properties` for MySQL:

1. Locate the `mysql_master.properties` sample configuration file:

Database	Master Properties File
MySQL	<js-install-9.0>/buildomatic/sample_conf/mysql_master.properties

2. Copy the file to `<js-install-9.0>/buildomatic`.

3. Rename the file `default_master.properties`.

4. Edit `default_master.properties` for your database and application server.

Database	Sample Property Values

MySQL	`appServerType=tomcat (or wildfly, etc.)`
	`appServerDir=c:\\Apache Software Foundation\\Tomcat 9.0 (for example)`
	`dbUsername=root`
	`dbPassword=password`
	`dbHost=localhost`

For the Split upgrade, configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.

Oracle Example

To configure `default_master.properties` for Oracle:

1. Locate the `oracle_master.properties` sample configuration file:

Database	Master Properties File
Oracle	<js-install-9.0>/buildomatic/sample_conf/oracle_master.properties

2. Copy the file to `<js-install-9.0>/buildomatic`.

3. Rename the file to `default_master.properties`.

4. Edit `default_master.properties` for your database and application server.

Database	Sample Property Values

Oracle	`appServerType=tomcat (or wildfly, etc.)`
	`appServerDir=c:\\Apache Software Foundation\\Tomcat 9.0 (for example)`
	`dbUsername=jasperserver`
	`dbPassword=password`
	`sysUsername=system`
	`sysPassword=password`
	`dbHost=localhost`

For the Split upgrade, configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.

Using Vendor's Drivers for Commercial Databases

JasperReports Server doesn't include JDBC drivers for the following commercial databases: Oracle, SQL Server, or DB2. If you want to use one of those databases, you need to obtain one of the available JDBC drivers and copy it into the correct location, and then edit

default_master.properties before running the upgrade steps. See Working With JDBC Drivers for more information.

Upgrading to JasperReports Server 9.0

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

Make sure you have taken backup of your `Jasperserver` database before proceeding.

Make sure you have taken backup of your old JasperReports Server WAR file before proceeding.

1. Stop your application server.

2. Start your database server.

3. Make sure that the user running the upgrade commands is the same user that installed the server.

4. Run the following commands:

Commands	Description
`cd <js-install-9.0>/buildomatic`	Change to buildomatic directory
`js-upgrade-newdb.bat <path>\js-8.0-export.zip`	(Windows) Upgrade jasperserver-pro war file, drop, and recreate the database, import data files from the previous version.
`./js-upgrade-newdb.sh <path>/js-8.0-export.zip`	(Linux) Upgrade jasperserver-pro war file, drop, and recreate the database, importing data files from the previous version.

On MySQL, if you receive an error about packet size, see the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

If you have auditing enabled, see the section about including audit events in the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

If the upgrade is Split, the Access, Audit, and Monitoring events are imported to the `audit` database during the import process.

If you are prompted to create a keystore, this means that the server's original keystore was not found in the user's home directory. Proceed with caution:

- In general, it is recommended to exit the upgrade procedure and make sure that the keystore is in the proper location, then rerun the upgrade.
- If you continue and create a keystore, then the upgrade proceeds but your repository is corrupted and users are unable to log in. In this case, you need to export manually the server's repository with a custom key, then import the key before importing the repository, as described in Encryption Keys.

js-upgrade Test Mode

Use the `test` option to run the `js-upgrade` script in test mode. For example, on Windows, enter:

```
cd <js-install-9.0>/buildomatic
```

```
js-upgrade-newdb.bat test <path>/js-8.0.x-export.zip
```

In test mode, the js-upgrade scripts check your default_master.properties settings and validate your application server location and its ability to connect to your database. Test mode can help you debug issues like an incorrect database password without altering your system.

Output Log Location

The js-upgrade script creates an output log that captures both standard and error output. If problems occur during script execution, or you just want to remember which options you

chose, open the output log file located here:

`<js-install-9.0>/buildomatic/logs/js-upgrade-<date>-<number>.log`

Errors

If you encounter errors running the `js-upgrade` script, first look at the output log to see if you can spot the errors. For help, refer to the Troubleshooting appendix of the *JasperReports Server Installation Guide*. The information in this appendix applies to both `js-upgrade` scripts and `js-install` scripts.

If you need to modify values in your `default_master.properties` file, you can simply edit the file. When you run the `js-upgrade` script again, it uses the new values.

Starting and Logging into JasperReports Server 9.0

Start your application server. Your database should already be running.

Clearing Your Browser Cache

Before you log in, make sure you and your end users clear the browser cache. JavaScript files, which enable the UI elements of JasperReports Server, are typically cached by the browser. Clear the cache to ensure that the newer files are used.

Logging into JasperReports Server

Log in using the following URL, user IDs, and passwords:

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

User ID	Password	Description
`superuser`	`<your-password>`	System-wide administrator
`jasperadmin`	`<your-password>`	Administrator for the default organization

Your JasperReports Server instance has now been upgraded to 9.0. If you have startup or login problems, refer to the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

Additional Tasks to Complete the Upgrade

Installing JasperReports Server automatically generates encryption keys that reside on the file system. These keys are stored in a dedicated Jaspersoft keystore. Make sure that this keystore is properly secured and backed up, as described in the *JasperReports Server Security Guide*.

Perform these tasks with the application server shutdown.

Handling JasperReports Server Customizations

If you made modifications to the original JasperReports Server application, you need to copy manually configuration changes, like client-specific security classes or LDAP server configurations, from your previous environment and integrate them with your upgraded environment. These configurations are typically found in the files at the `WEB-INF/` location, for example, applicationContext-*.xml, *.properties, *.js, *.jar, and so on.

Clearing the Application Server Work Folder

Application servers have work folders where JasperReports Server files are compiled and cached and other objects are stored. When you update the WAR file or license, the buildomatic `deploy-webapp-pro` target should automatically clear the application server's `work` directory, but it is a good practice to double-check. A permission problem, or some other problem, could prevent the clearing of the work folder.

To clear the work folder in Tomcat

1. Change the directory to `<tomcat>/work`.

2. Delete all the files and folders in this directory.

Clearing the Application Server Temp Folder

JasperReports Server uses caching to speed operations within the application. Caching files are created and stored in the application server, usually in a `temp` folder. Clear this `temp` folder to avoid any post-upgrade conflicts. Typically, the `temp` folder used by an application server corresponds to the path referenced by the `java.io.tmpdir` Java system property. For Apache Tomcat the `temp` folder is `<tomcat>/temp`.

To clear the temp folder in Apache Tomcat

1. Change the directory to `<tomcat>/temp`.

2. Delete all the files and folders in this directory.

Clearing the Repository Cache Database Table

In the `Jasperserver` database, compiled JasperReports Library resources are cached in the `JIRepositoryCache` table for increased efficiency at runtime. Because the JasperReports Library JAR is typically updated with each new release, old cached items can get out of date and cause errors at runtime. If you encounter errors that mention a JasperReports Library "local class incompatible", check your repository cache table. In summary, you can clear your `Jasperserver` database cache table as part of this upgrade process whether there are errors or not.

To clear the repository cache database table manually, run a SQL command similar to the one shown below:

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

Old Manual Upgrade Steps

This section describes the older, manual upgrade steps used before we implemented the `js-upgrade` shell scripts in release 4.0. They are provided here mainly as a reference for internal use.

We recommend using the js-upgrade shell scripts described in the beginning of this chapter instead of these manual commands.

Commands	Description
`cd <js-install-9.0>/buildomatic`	

`js-ant drop-js-db`
`js-ant create-js-db`
`js-ant init-js-db-pro`
	Deletes and recreates your `jasperserver` db. Make sure that your original database is backed up.

`js-ant import-minimal-pro`	

Windows:
`js-ant import-upgrade -DimportFile="<path-and-filename>" -DimportArgs="--include-server-settings --secret-key='0x1b 0xd4 0xa6 ...'"`
Linux and Mac OSX:
`js-ant import-upgrade -DimportFile=\"<path-and-filename>\" -DimportArgs=\"--include-server-settings --secret-key=\'0x1b 0xd4 0xa6 ...\'\"`
	The `-DimportFile` should point to the <path> and <filename> of the js-8.0.x-export.zip file you created earlier.
	`--include-server-settings --secret-key` specifies the key to use for the import. Use the same key that you imported into the keystore.
	On Windows, you must use double quotation marks (") if your path or filename contains spaces. On Linux and Mac OSX, you must use double quotation marks, escaped with a backslash (\") in this case. On Linux and Mac OSX, you must also escape any single quotation marks with a backslash.

Commands	Description
	Note: "import-upgrade" imports the resources from the 8.0.x instance in a "non-update" mode (so that core resources from 9.0 stay unchanged). Additionally, the "update-core-users" option is applied so that the superuser and jasperadmin users have the same password as set in the 8.0.x instance.
`js-ant import-sample-data-upgrade-pro`	(Optional) This step is optional. It loads the new sample data. The old sample data is overwritten, so you may need to redo certain changes such as configuring the sample data sources for your database.
`js-ant deploy-webapp-pro`	Deletes the existing older war file, deploys the new war file.

Migrating from Compact 9.0 to Split 9.0

This chapter describes the recommended procedure for upgrading to JasperReports Server 9.0 Split installation from JasperReports Server 9.0 Compact installation.

This chapter contains the following sections:

Migrating From Compact to Split (samedb)

To migrate from Compact installation to Split installation (samedb):

1. Configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.
2. Run the following command to migrate from Compact to Split:
   - Windows: `js-migrate-to-split-samedb.bat`
   - Linux and Mac OSX: `./js-migrate-to-split-samedb.sh`

   The `audit` database is created with empty tables.
3. Run the following command to transfer the data (Audit, Access, and Log monitoring data) to the `audit` database from the `jasperserver` database:
   - Windows: `transfer-audit-data.bat`
   - Linux and Mac OSX: `./transfer-audit-data.sh`

   The data is transferred to the `audit` database and the tables are deleted from the `jasperserver` database. Rerun the command if there is any interruption in the data transfer process, it will resume the transfer process from where it was interrupted in the previous run.

Migrating From Compact to Split (newdb)

Prerequisite: Export all the data as described in Exporting Current Repository Data.

To migrate from Compact installation to Split installation (newdb):

1. Configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.
2. Run the following command to migrate from Compact to Split and import the resources:
    - Windows:
        - `js-migrate-to-split-newdb.bat js-<ver>-export.zip` (Migrate without the Audit, Access, and Log Monitoring data)
        - `js-migrate-to-split-newdb.bat js-<ver>-export.zip include-access-events include-audit-events include-monitoring-events` (Migrate with the Audit, Access, and Log Monitoring data)
    - Linux and Mac OSX:
        - `./js-migrate-to-split-newdb.sh js-<ver>-export.zip` (Migrate without the Audit, Access, and Log Monitoring data)
        - `./js-migrate-to-split-newdb.sh js-<ver>-export.zip include-access-events include-audit-events include-monitoring-events` (Migrate with the Audit, Access, and Log Monitoring data)

The `jasperserver` database and `audit` database are created and the resources are imported from the zip file provided as the input.

Upgrading JasperReports Server 6.4.x or Earlier

Upgrading from 6.4.x or Earlier

If you are running JasperReports Server version 6.4.x or earlier, your upgrade requires multiple steps.

If you are running JasperReports Server 6.0 through 6.4.x:

1. Upgrade to the latest version of 7.1.x.
2. Upgrade from 7.1.x to version 8.2.
3. Upgrade from 8.2 to version 9.0.

The steps for upgrade to 6.3.x, 6.4.x, or 7.1.x are documented in the *JasperServer Installation Guide* for that release. Download the JasperReports Server WAR file distribution zip package for the release that you want to get the relevant files and documentation. The Installation Guide is in the docs folder.

You can download the JasperReports Server WAR file distribution zip package from Jaspersoft Technical Support (https://www.jaspersoft.com/support)or contact your sales representative.

If you are running a JasperServer version earlier than 3.7, first upgrade to 3.7.0, then to 6.3.x, then 7.x to 8.x and then to 9.0.

Best Practices for Upgrading on Windows

The two methods for installing JasperReports Server are:

1. Installing with the Binary Installer and Bundled Components

The binary installer is an executable that puts all the components in place to run JasperReports Server. For example, if you take the default installation choices, you get the

Apache Tomcat application server, the PostgreSQL database, and the Java execution environment.

But keep in mind that these components are specially configured to run a specific version of JasperReports Server. This applies to the Windows Start Menu items created to start and stop JasperReports Server.

2. Installing to Pre-existing Components

When installing a “Production" instance of JasperReports Server, you may want to install the main components before you install JasperReports Server. This way you have more control over updating and upgrading components like the application server, database, and Java.

Once you put these components in place, you have two options for installing JasperReports Server:

a. Use the War File ZIP distribution (file name: js-jrs_9.0.0_bin.zip)

You can install JasperReports Server to the existing components using the `js-install.bat` script. You create a default_master.properties file that specifies the location of the application server and database components.

b. Use the Binary Installer, js-jrs_9.0.0_win_x86_64.exe

The installer prompts you for the location of the application server and database components.

If you intend to upgrade your Windows installation with future releases of JasperReports Server, we recommend installing pre-existing components. This reduces any post-upgrade confusion caused by the Windows Start Menu showing the older version of JasperReports Server.

Upgrading from the Community Project

If you are running a Community Project (CP) instance of JasperReports Server and want to upgrade to a commercial version of JasperReports Server, follow the instructions in this chapter.

This upgrade process uses the JasperReports Server commercial WAR File Distribution release package and the included buildomatic scripts.

This CP to commercial upgrade procedure is valid only for upgrade within a major JasperReports Server release, for example 9.0 CP to 9.0 commercial.

This chapter contains the following sections:

General Procedure

The upgrade procedure consists of the following main steps:

1. Back up your JasperReports Server CP instance.

2. Export your CP repository data.

3. Upgrade your instance to JasperReports Server Commercial.

4. Import your CP repository data.

If you customized or extended JasperReports Server CP, you need to keep track of these modifications and integrate them with your JasperReports Server commercial instance after completing the upgrade.

Backing Up Your JasperReports Server CP Instance

Back up the old JasperReports Server CP WAR file and `Jasperserver` database in case a problem occurs with the upgrade. Perform these steps from the command line in a Windows or Linux shell.

These instructions assume you have a Tomcat application server and the PostgreSQL or MySQL database. Other application servers require a similar procedure. If you have another database, consult your DB administration documentation for backup information.

Backing Up Your JasperReports Server CP WAR File

For example, for Apache Tomcat, back up the `jasperserver` directory from the `<tomcat>/webapps` folder:

1. Go to the `<tomcat>` directory.
2. Make a new directory with a name, `js-cp-war-backup`.
3. Copy `<tomcat>/webapps/ jasperserver` to `<tomcat>/js-cp-war-backup`.
4. Delete the `<tomcat>/webapps/jasperserver` directory.

Backing Up Your JasperReports Server Database

Go to the location where you originally unpacked your CP WAR File Distribution zip. (Or create a new local folder to hold your backup file.)

1. Go to the `<js-install-cp>` directory.
2. Run one of the following commands:
    - For PostgreSQL on Windows or Linux:

```
cd <js-install-cp>
pg_dump --username=postgres  jasperserver  >  js-db-cp-
dump.sql
```

- For MySQL on Windows:

```
mysqldump --user=root --password=<password> jasperserver > js-
db-cp-dump.sql
```

- For MySQL on Linux:

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

For MySQL, if you receive an error about packet size, see the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

Backing Up Your Keystore

Back up your JasperReports Server Keystore

1. Create a folder (if you did not do so already) where you can save your server's keystore, for example `C:\JS_BACKUP` or `/opt/JS_BACKUP`.

2. As the user who originally installed the server, copy `$HOME/.jrsks` and `$HOME/.jrsksp` to `<path>/JS_BACKUP`. Remember that these files contain sensitive keys for your data, so they must always be transmitted and stored securely.

Exporting Your CP Repository Data

Before exporting your CP repository data, check to see if you have the `default_master.properties` file in this directory.

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

This file holds settings specific to your JasperReports Server instance, such as your application server location and your database type and location. If you do not have this file, see Example Buildomatic Configuration .

To export your CP repository data

1. Navigate to the buildomatic directory:

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

2. Run buildomatic with the export target:

Windows:	`js-ant.bat export-everything-ce -DexportFile=js-cp-export.zip`
Linux:	`./js-ant export-everything-ce -DexportFile=js-cp-export.zip`

This operation uses the export option `--everything`, which collects all your repository data.

Remember the path to your exported file. You need to specify it when you import to your commercial JasperReports Server repository.

Preparing the JasperReports Server 9.0 WAR File Distribution

Use the buildomatic scripts included in the commercial 9.0 WAR File Distribution release package for the upgrade. Follow these steps to obtain and unpack the commercial 9.0 WAR file distribution ZIP file:

1. The WAR File Distribution comes in a compressed ZIP file named `js-jrs_9.0.0_bin.zip`. Download the WAR File Distribution from Jaspersoft Technical Support (https://www.jaspersoft.com/support) or contact your sales representative.

2. Extract all files from `js-jrs_9.0.0_bin.zip`. Choose a destination, such as `C:\Jaspersoft` on Windows, `/home/<user>` on Linux, or `/Applications` on Mac OSX.

After you unpack the WAR File Distribution Zip, the resulting location is known as:

```
<js-install-pro>
```

Configuring Buildomatic for Your Database and Application Server

This upgrade procedure uses the buildomatic scripts included with the WAR File Distribution ZIP release package.

Example Buildomatic Configuration

The `default_master.properties` file handles the upgrade configuration. We provide a sample configuration file for each database. You must specify your database credentials and your application server location, and rename the file to `default_master.properties`.

PostgreSQL Example

This example uses PostgreSQL (the same general logic applies to other databases).

1. Copy `postgresql_master.properties` from:

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

2. Paste the file to:

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

3. Rename the file to: `default_master.properties`
4. Edit default_master.properties for your database and application server. Sample property values are:
   - `appServerType=tomcat (or wildfly, and so on)`
   - `appServerDir=c:\\Apache Software Foundation\\Tomcat 9.0 (for example)`
   - `dbUsername=postgres`
   - `dbPassword=postgres`
   - `dbHost=localhost`

For the Split upgrade, configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.

MySQL Example

This example uses MySQL (the same general logic applies to other databases).

1. Copy `mysql_master.properties` from:
   `<js-install-pro>/buildomatic/sample_conf`
2. Paste the file to:
   `<js-install-pro>/buildomatic`
3. Rename the file to: `default_master.properties`
4. Edit default_master.properties for your database and application server. Sample property values are:
   - `appServerType=tomcat (or wildfly, and so on)`
   - `appServerDir=c:\\Apache Software Foundation\\Tomcat 9.0 (for example)`
   - `dbUsername=root`
   - `dbPassword=password`
   - `dbHost=localhost`

For the Split upgrade, configure the settings in the `default_master.properties` file as described in Additional Buildomatic Configuration for Split Installation Upgrade.

Upgrading to the Commercial Version of JasperReports Server 9.0

After configuring the default_master.properties file, you can complete the upgrade.

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

Make sure you have backed up your old JasperReports Server WAR file before proceeding.

1. Stop your application server.

2. Start your database server.

3. Make sure that the user running the upgrade commands is the same user that installed the server.

4. Run the following commands:

Commands	Description
`cd <js-install-pro>/buildomatic`	

`js-ant drop-js-db`
`js-ant create-js-db`
`js-ant init-js-db-pro`
	The first command deletes your jasperserver db. Make sure it is backed up. The other commands recreate and initialize the database.

`js-ant import-minimal-pro`	Adds superuser, themes, and default tenant structure.

Windows:
`js-ant import-upgrade -DimportFile="<path>/js-cp-export.zip" -DimportArgs="--include-server-settings --secret-key='0x1b 0xd4 0xa6 ...'"`
Linux and Mac OSX:
`js-ant import-upgrade -DimportFile=\"<path>/js-cp-export.zip\" -DimportArgs=\"--include-server-settings --secret-key=\'0x1b 0xd4 0xa6 ...\'\"`
	The `-DimportFile` argument should point to the js-cp-export.zip file you created earlier.
	`--include-server-settings --secret-key` specifies the key to use for the import. Use the same key that you imported into the keystore.
	On Windows, you must use double quotation marks (") if your path or filename contains spaces. On Linux, you must use double quotation marks escaped with a backslash (\") in this case.

`js-ant import-sample-data-upgrade-pro`	(Optional) Loads the 9.0 commercial sample data.
`js-ant deploy-webapp-cp-to-pro`	Delete the CP war file, and deploy the commercial (pro) war file.

Commands	Description
`js-ant create-audit-db`	(Optional) Creates the audit database. Required only for the Split installation.
`js-ant init-audit-db-pro`	(Optional) Initializes the audit database. Required only for the Split installation.

On MySQL, if you receive an error about packet size, see the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

If you are prompted to create a keystore, this means that the server's original keystore was not found in the user's home directory. Proceed with caution:

- In general, it is recommended to exit the upgrade procedure and make sure that the keystore is in the proper location, then rerun the upgrade.
- If you continue and create a keystore, then the upgrade proceeds but your repository is corrupted and users are unable to log in. In this case, you need to export the server's repository with a custom key as described in Encryption Keys. Then replace the `import-upgrade` commands in the table above with the following ones that specify the `secret-key` value from the export:

Windows	Linux and Mac OSX
`js-ant import-upgrade -DimportFile="<path>/js-cp-export.zip" -DimportArgs="--include-server-settings --secret-key='0xb1 0x44 0x72 ...'"`	`js-ant import-upgrade -DimportFile=\"<path>/js-cp-export.zip\" -DimportArgs=\"--include-server-settings --secret-key=\'0xb1 0x44 0x72 ...\'\"`

Starting and Logging into JasperReports Server 9.0

Before starting the server:

1. Set up the JasperReports Server License.

Copy the `<js-install-pro>/jasperserver.license` file to the `C:\Users\<user>` directory (Windows 7 example).

For information about how to set up the license, see the *JasperReports Server Installation Guide*.

2. Delete any files in the `<tomcat>\temp` folder.

3. Delete any files, directories, or subdirectories in `<tomcat>\work\Catalina\localhost`.

4. Delete any jasperserver*.xml files that might exist in `<tomcat>\conf\Catalina\localhost`.

5. (Optional) Move any existing `<tomcat-install>\logs` files into a backup directory to clean up old CP log data.

For instructions on clearing directories, see Additional Tasks to Complete the Upgrade.

Now start your Tomcat or JBoss application server. Your database should already be running.

Clearing Your Browser Cache

Before you log in, make sure you and your end-users clear the browser cache. JavaScript files, which enable UI elements of JasperReports Server, are typically cached by the browser. Clear the cache to ensure that the newer files are used.

Logging into the Commercial Version of JasperReports Server 9.0

Log in using the following URL, user IDs, and passwords:

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

User ID	Password	Description
`superuser`	`superuser`	System-wide administrator
`jasperadmin`	`jasperadmin`	Administrator for the default organization

Your jasperadmin password might be reset to the default setting by the upgrade operation. For example, the Jasperadmin password might be reset to `jasperadmin`. For security reasons, you should change your Jasperadmin and superuser passwords to non-default values.

Your JasperReports Server instance has now been upgraded from Community Project (CP) to commercial. If startup or login problems occur, refer to the Troubleshooting appendix of the *JasperReports Server Installation Guide*.

Re-Configuring XML/A Connections (Optional)

XML/A connection definitions contain a username and password for connecting the Web Services to the server. A commercial edition of JasperReports Server supports multi-tenancy, which allows multiple organizations on a single instance. The default organization is `organization_1`. Each user (except a superuser) must belong to a specific organization. After upgrading to the commercial JasperReports Server, users belong to the default organization.

You need to update XML/A connection definitions to include the organization the user belongs to.

The XML/A connection also specifies an instance URI. You need to update this URI to the commercial instance. Edit your XML/A connections as shown in the following examples:

- User IDs
    - Change "`jasperadmin`" to "`jasperadmin|organization_1`"
    - Change "`joeuser`" to "`joeuser|organization_1`"
- URI values

Change:

`http://localhost:8080/jasperserver/xmla`

to

`http://localhost:8080/jasperserver-pro/xmla`

Additional Tasks to Complete the Upgrade

Perform these tasks with the application server shutdown.

Handling JasperReports Server Customizations

If you made modifications to the original JasperReports Server application, you need to copy manually configuration changes, like client-specific security classes or LDAP server configurations, from your previous environment and integrate them with your upgraded environment. These configurations are typically found in the files at the `WEB-INF/` location, for example, applicationContext-*.xml, *.properties, *.js, *.jar, and so on.

Clearing the Application Server Work Folder

Application servers have work folders where JasperReports Server files are compiled and cached and other objects are stored. When you update the WAR file or license, the buildomatic `deploy-webapp-pro` target should automatically clear the application server's `work` directory, but it is a good practice to double-check. A permission problem, or some other problem, could prevent the clearing of the work folder.

To clear the work folder in Tomcat

1. Change the directory to `<tomcat>/work`.

2. Delete all the files and folders in this directory.

Clearing the Application Server Temp Folder

JasperReports Server uses caching to speed operations within the application. Caching files are created and stored in the application server, usually in a `temp` folder. Clear this `temp`

folder to avoid any post-upgrade conflicts. Typically, the `temp` folder used by an application server corresponds to the path referenced by the `java.io.tmpdir` Java system property. For Apache Tomcat the `temp` folder is `<tomcat>/temp`.

To clear the temp folder in Apache Tomcat

1. Change the directory to `<tomcat>/temp`.
2. Delete all the files and folders in this directory.

Clearing the Repository Cache Database Table

In the `Jasperserver` database, compiled JasperReports Library resources are cached in the `JIRepositoryCache` table for increased efficiency at runtime. Because the JasperReports Library JAR is typically updated with each new release, old cached items can get out of date and cause errors at runtime. If you encounter errors that mention a JasperReports Library "local class incompatible", check your repository cache table. In summary, you can clear your `Jasperserver` database cache table as part of this upgrade process whether there are errors or not.

To clear the repository cache database table manually, run a SQL command similar to the one shown below:

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

Planning Your Upgrade

Some of the new and enhanced features in JasperReports Server can affect your deployment, and you should plan your upgrade accordingly. Before upgrading make sure to:

- Review this information carefully and determine how the changes described affect your deployment.
- Back up your current JasperReports Server installation, repository, and keystore.
- Run the upgrade script as the same user who originally installed the server, or make sure the server's keystore is available in the home directory of the user running the upgrade script.

The versions and their affected functionality are:

- Changes in 9.0 affect upgrades.
- Changes in 8.2 affect upgrades.
- Changes in 8.1 affect upgrades. Users will not be able to upgrade from 8.0 Compact to 8.1 Split, or from 8.0 Split to 8.1 Compact. Currently, the **js-upgrade-newdb.sh/bat** script does not import the access, audit, monitoring data when upgrading.
- Changes in 8.0 affect the installation and upgrades. The Split installation has been introduced from this release. The Audit, Access, and Monitoring events can be moved to a different audit database using the Split installation or upgrade, which improves the performance of JasperReports server.
- Changes in 7.8 affect PhantomJS/Rhino JavaScript engine. With this release, the supported JavaScript engine is Chrome/Chromium.
- Changes in 7.5 affect Simba and Impala drivers, the MongoDB query language, custom themes, and encryption keys.
- Changes in 7.2 affect legacy dashboards, customizations to the login page, external authentication, and customizations to the Spring Security framework.
- Changes in 7.1 affect customizations to the login page.
- Changes in 6.4 affect the Impala community connector.

- Changes in 6.2.1 affect the Impala community connector.
- Changes in 6.2 affect the default Ad Hoc templates.
- Changes in 6.1 affect themes.

Changes are cumulative, so review all topics that affect you. For example, if you are upgrading from 6.1 to 7.1, you may be affected by changes in 6.1, 6.2, 6.2.1, and 6.4.

For versions of the software earlier than 6.1, see earlier versions of the *JasperReports Server Upgrade Guide*.

This section describes only those changes that can significantly impact your existing deployment. For an overview of new features, improvements, and bug fixes see the release notes in the root directory of the distribution. For information on how to use the new features, see the *JasperReports Server User Guide* or the *JasperReports Server Administrator Guide*.

This chapter contains the following sections:

Changes in 9.0 That May Affect Your Upgrade

UI Customizations Note

If you have done any customizations for your UI (JavaScript and CSS files), you have to first perform the JasperReports Server upgrade, then get the JasperReports Server Source Packages, apply the customizations in the UI source files, rebuild them and publish into the upgraded JasperReports Server.

Progress Driver Removal

As of release 9.0, Progress drivers are removed from JasperReports Server. If you have any resources that use or depend on any of the following Progress Drivers:

- TIautorest-6.0.1.005359.jar
- TIcassandra-6.0.3.jar
- TIdb2-5.1.4.000288.jar
- TIgooglebigquery-6.0.0.001346.jar
- TIhive-5.14.1.jar
- TIimpala-5.14.2.jar
- TImongodb-6.0.2.000510.jar
- TIoracle-6.0.0.000790.jar
- TIredshift-5.14.1.jar / TIredshift-6.0.0.000366.jar
- TIsforce-6.0.0.001533.jar
- TIsparksql-6.0.1.000124.jar
- TIsqlserver-6.0.0.000541.jar

  then you must manually install drivers that are recommended by the database vendor. After installing new drivers, update the resources in JasperReports Server to use the new drivers.

Additional Steps to Migrate JasperReports Server on MS SQL Server, Oracle, DB2

The migration to JasperReports Server 9.0 and a shift from Progress drivers to Database vendor drivers can bring challenges, particularly with the behavior of resources. You may experience instances where, following the upgrade, the resources either stop functioning or display altered behaviors. This discrepancy is often caused by the variance in how Progress JDBC drivers and Native drivers interpret certain data types.

Let us review some patterns of the problems identified during this migration and explore two strategies to address these issues:

1. Update your resources to reflect data type changes:

   One approach to resolve such issues is to update your resources to align with the changes in data types. Progress JDBC drivers may read certain data types differently than native vendors, leading to discrepancies in resource behavior. After identifying such cases in reports, domains and adhoc views, modify affected resources to reflect the new data type conventions.

2. Override the default data type mapping

   For users who prefer to keep the old data type mapping, another solution is to override the default JDBC data type mapping. This involves finding what data types behave differently in your resources and configuring JasperReports Server to use old type mapping for certain database. While this approach maintains consistency with the previous setup, it's crucial to thoroughly test and validate the reports to ensure accuracy and reliability.

SQL Server

Handling SQL Server Time and Float columns in JasperReports Server domains

When using a migrated domain, adhoc view, or adhoc report, you may encounter a message indicating missing fields or columns. This issue occurs due to the transition from the Progress driver, which interpreted Time columns as Timestamp and Float as Float, to the SQL Server native driver, where Time is read as Time and Float as Double. To revert

this change in data types, we have introduced additional configuration beans that enable users to override the default type mapping.

To revert to old behavior:

1. Edit `../WEB-INF/applicationContext-jdbc-metadata.xml`.
2. find `<bean class="com.jaspersoft.jasperserver.api.engine.common.domain.JdbcDriverMetaConfigurationImpl">` which has property `<property name="databaseProductName" value="Microsoft SQL Server"/>`.
3. Uncomment the whole bean.
4. Restart JasperReports Server.

After the change is applied, when using a SQL Server JDBC/JNDI connection in Domains or AdHoc, all Time fields will be interpreted as Timestamp, and all Float columns (with JDBC code 8) will be treated as Double.

Using SQL Server Time field as a parameter or field in a JRXML report

Executing a report unit may result in exceptions like **The data types time and datetime are incompatible in the equal to operator.** This can occur when a Time field, initially treated as Timestamp or DateTime, is referenced in the WHERE clause of an SQL Query, resulting in a type mismatch.

You can resolve this issue using any of the following two options:

- update JRXML and SQL query to reflect type change.
- force the SQL Server JDBC driver to return Time as DateTime by updating the JDBC connection URL and adding `sendTimeAsDatetime=false`.

Oracle

Handling Oracle Float and Double Precision Types in JasperReports Server Domains

When using a migrated domain, adhoc view, or adhoc report, users may encounter a message indicating missing fields or columns. This issue occurs due to the transition from the Progress driver, which interpreted Float or Double Precision columns as Double, to the Oracle native driver, where these types are interpreted as Float. To revert this change in data types, we have introduced additional configuration beans that enable users to override the default types mapping.

To revert to old behavior perform:

1. Edit `../WEB-INF/applicationContext-jdbc-metadata.xml`.
2. Find `"<bean class="com.jaspersoft.jasperserver.api.engine.common.domain.JdbcDriverMetaConfigurationImpl">" which has property "<property name="databaseProductName" value="Oracle"/>"`.
3. Uncomment that whole bean.
4. Restart JasperReports Server.

After change is applied, when using a Oracle JDBC/JNDI connection in Domains or AdHoc, all Float or Double Precision fields will be interpreted as Double.

Using Oracle Numeric types with Precision and Scale in JRXML Report

When using Oracle numeric fields with precision and scale, such as Numeric (10,4), in JRXML reports, discrepancies may occur in values with trailing zeros. For instance, the value '6.1200' in a report might be displayed as '6.12.' This occurs due to the Oracle native driver's behavior of not preserving trailing zeros, which was previously managed by the Progress driver during formatting.

To resolve this issue, it is recommended to control value formatting within the JRXML, than relying on the database driver. To implement this fix, edit the JRXML file and introduce field formatting by adding:

```
<textField pattern="#,###.0000" isBlankWhenNull="true">
```

Using Aggregate Functions for Oracle Date types in JRXML Reports

After the migration of adhoc resources dependent on JRXML topics to the Oracle native driver, you may encounter difference in the results of aggregate functions on date fields. Additionally, the Hide Duplicate Rows adhoc function might return identical (duplicate) values. This discrepancy occurs from the difference in how Oracle database handles date values compared to the Progress driver.

Oracle database lacks a dedicated date type and stores dates with hours, minutes, and seconds. Oracle's DATE type is similar to Timestamp type. So for such date fields, Oracle native driver returns date values with their associated hours, minutes, and seconds, while the Progress driver always sets these time components to zero.

The consequences are noticeable when applying aggregate functions, particularly Range, as different results may be displayed when non-zero hours/minutes/seconds/miliseconds values are stored in the Oracle database. Detecting the differences in milliseconds between two values may be challenging, especially because the Oracle native driver returns the column type as `java.sql.Date`. If in JRXML the same type is set, the JasperReports Server AdHoc functionality restricts changing the format to display hours, seconds, and milliseconds.

There are two methods to align behavior with that of the Progress driver:

1. The initial approach relies on editing the report query. By utilizing the SQL TRUNCATE function for the date field, the Oracle driver will return the date along with zeroed hours, minutes, seconds, and milliseconds. For instance:

   Original Query:

   ```
   SELECT id, date_entered FROM my_table
   ```

   Modified Query:

   ```
   SELECT id, TRUNC(date_entered) FROM my_table
   ```

2. The alternative method is to edit the JRXML file and modifying the required field type to Timestamp.

Handling Oracle Date and Timestamp Fields Referenced in Domains

After migrating to Oracle native driver, some domains may produce issues with complex joins that references Timestamp as strings.

Consider the following example:

```
SELECT *
FROM (
    SELECT
        "EMPLOYEE"."SALARY" AS "EMPLOYEE_SALARY1",
        "STORE"."FIRST_OPENED_DATE" AS "STORE_FIRST_OPENED_DATE"
    FROM "FOODMART"."STORE" "STORE"
    INNER JOIN "FOODMART"."EMPLOYEE" "EMPLOYEE"
        ON ("STORE"."STORE_ID" = "EMPLOYEE"."STORE_ID" AND
"STORE"."FIRST_OPENED_DATE" < '1985-03-04')
)
WHERE ROWNUM <= 1000
```

In this scenario, the Timestamp value `1985-03-04` is treated as a string by the database.

When running reports or views relying on such domain schemas, an Oracle exception `ORA-01843: not a valid month` may occur.

Oracle requires proper conversion of the string to a Date or Timestamp. Without explicit date format and locale provided, the Oracle native driver uses the database system format and locale. This behavior is different from Progress Oracle driver, which consistently uses ANSI format by default.

To resolve this issue, edit the Domain Schema and append `ts` to the date value in complex join condition. For example, `1985-03-04` should be represented as `ts'1985-03-04'`.

Alternatively, you can use the `to_timestamp` function: `to_timestamp('1985-03-04 00:00:00', 'YYYY-MM-DD HH24:MI:SS.FF')`.

DB2

Adjustments in DB2 Error Messages After Migrating to Native Driver

After the migration to the native DB2 driver, you may observe a change in the format of SQL error messages, which now provide fewer details as compared to how Progress DB2 driver handled errors. Instead of receiving error descriptions directly, you will get messages containing SQLCODE and SQLERROR.

Let's compare the error messages for the same SQL error:

	SQL error	SQL State	Vendor code
Progress Driver	[TibcoSoftware][DB2 JDBC Driver] [DB2]PRODUCT1_ID NOT COLUMN OF INSERTED/UPDATED TABLE, OR ANY TABLE IN A FROM CLAUSE	42703	206
DB2 Native Driver	DB2 SQL Error: SQLCODE=-206, SQLSTATE=42703, SQLERRMC=PRODUCT1_ID, DRIVER=4.32.28	42703	206

While this change doesn't impact the functionality, for investigating failing queries you may now need to refer to the official DB2 Java driver documentation and use SQLCODE and SQLSTATE to search for the corresponding error description. The new format, though different, still provides the essential information needed for error analysis.

AdHoc Component in Reports

With the introduction of the AdHoc Component feature, every new AdHoc report generated from an AdHoc view automatically activates this functionality. This integration ensures that any modifications made in the AdHoc view are automatically reflected in its associated AdHoc Report. AdHoc Reports created prior to or imported from earlier JasperReports Server versions will maintain their existing behavior and won't dynamically incorporate

AdHoc view changes. To use the AdHoc Component feature for your AdHoc Reports, it's recommended to recreate reports from the AdHoc views in JasperReports Server 9.0.

The AdHoc Component feature relies on new report templates shipped with JasperReports Server 9.0, located at "/public/templates." It's important to be aware that these templates have the same names as before. During the import of older resources, there is a risk of overwriting these templates by accident. In such cases, although AdHoc reports may be generated and function correctly, they will be based on legacy report templates. To fix this, you can re-import these templates that are shipped as part of our export-minimal-catalogs. The templates and can be imported using the `js-ant import-minimal-pro` command (for details, see the *JasperReports Server Installation Guide*).

If your deployment uses custom AdHoc templates, it is advised to update them to the new format based on the templates found in `/public/templates`.

JasperReports Server and JasperReports Web Studio Integration

The introduction of the latest JasperReports Web Studio brings a requirement to deploy two additional applications crucial for the normal functioning of JasperReports Web Studio. These applications can be deployed either on the same application server as JasperReports Server or remotely.

Consider allocating additional memory for your application server to meet the demand generated by the deployment of two additional applications for JasperReports Web Studio. The amount of required memory depends on the usage pattern of JasperReports Web Studio. Typically, the memory requirements for JasperReports Web Studio are significantly lower than those for the main JasperReports Server application.

Advanced Date Time Calculations

To support new calculations, the following Date grouping functions were renamed:

- Quarter renamed to Quarter and Year (examples, Q1 2024, Q2 2024, Q3 2024, Q4 2024).
- Month renamed to Month and Year (examples, January 2024, February 2024).

Updates to JNDI Data Sources

With the addition of the new JNDI Security feature, it is required to create the new JNDI resources even if the feature is currently disabled. If there are plans to enable this feature, ensure that all resources dependent on `jdbc/jasperserver` transition to `jdbc/jasperserverSystemAnalytics`. Similarly, resources using `jdbc/jasperserverAudit` should migrate to `jdbc/jasperserverAuditAnalytics`.

OAuth with OpenID

The introduction of OAuth support requires migration of configurations for previous deployments of JasperReports Server. If your deployment relied on OAuth configurations customized within `applicationContext-externalAuth-*.xml`, it is required to migrate these configurations to the new format.

For more information, see *JasperReports® Server External Authentication Cookbook*.

Alerting in Report Viewer

The Alerting feature in the Report Viewer operates on the Quartz engine, similar to the Scheduler. To support the increased load from Alert executions on the JasperReports Server system, the configuration of `org.quartz.threadPool.threadCount` is increased from 2 to 4. If your deployment has modified this property, it is recommended to increase it by adding additional threads specifically for Alerting tasks. It's important to note that the introduction of new Alerting tasks may result in additional load on the overall system performance. However, the footprint of Alerting tasks is comparable to Scheduler tasks, because Alert executions perform report executions when triggered, similar to how scheduled jobs do it. It is recommended to adjust the system configuration based on your usage of the Alerting feature to optimize performance.

Improvements in the Existing Column Names of the Schedules Page

The view of the Schedules page has been improved with updates to the following existing column names:

Old Column Names	New Column	Remarks
Job name	Job name/Description	Updated
Resource	Resource/URL	Updated
Owner	User	Updated

Disabled Deletion of "AnonymousUser"

The ability to delete the system user "AnonymousUser" is now disabled. This role is strictly an internal system user that is required for the normal functioning of JasperReports Server.

Container Element for viz.js

Due to security reasons when setting a container for viz.js, that container element cannot be a Script Element or an Element that has children script tags.

Configuring Scheduler for Dashboard

JasperReports Server allows you to enable the scheduler and headless browser to load the dashboard on the server side and export it. You need to set the `deploy.base.local.url` property in the `js.config.properties` file when "detailed" is selected during `Export`.

Newdb Upgrade Note

Currently, the **js-upgrade-newdb.sh/bat** script does not import the access, audit, monitoring data when upgrading.

However, once the upgrade process has completed, you can use the **JRS UI - Import** page to reimport the JRS export file that was passed in with the newdb script and select the checkboxes for including access, audit, and monitoring data. Once this import is completed, then the access, audit, monitoring data exist in the new database/release.

Important Notes about Compact and Split installations

- Users are able to upgrade from 8.2 Compact to 9.0 Compact using samedb and newdb.
- Users are able to upgrade from 8.2 Split to 9.0 Split using samedb and newdb.
- Users will not be able to upgrade:
    - From 8.2 Compact to 9.0 Split.
    - From 8.2 Split to 9.0 Compact.

If users need 9.0 Split installations but they are on 8.2 Compact, the required upgrade path is to:

1. Upgrade 8.2 Compact to 9.0 Compact.
2. Then, migrate from 9.0 Compact to 9.0 Split.

For more information on these installation options, see the *Installation* and *Upgrade* guides.

Changes in 8.2 That May Affect Your Upgrade

- Users are able to upgrade from 8.1 Compact to 8.2 Compact using samedb and newdb.
- Users are able to upgrade from 8.1 Split to 8.2 Split using samedb and newdb.
- Users will not be able to upgrade:
    - From 8.1 Compact to 8.2 Split.
    - From 8.1 Split to 8.2 Compact.

If users need 8.2 Split installations but they are on 8.1 Compact, the required upgrade path is to:

1. Upgrade 8.1 Compact to 8.2 Compact.
2. Then, migrate from 8.2 Compact to 8.2 Split.

For more information on these installation options, see the *Installation* and *Upgrade* guides.

Newdb Upgrade Note

Currently, the **js-upgrade-newdb.sh/bat** script does not import the access, audit, monitoring data when upgrading.

However, once the upgrade process has completed, you can use the **JRS UI - Import** page to reimport the JRS export file that was passed in with the newdb script and select the checkboxes for including access, audit, and monitoring data. Once this import is completed, then the access, audit, monitoring data exist in the new database/release.

UI Customizations Note

If you have done any customizations for your UI (JavaScript and CSS files), you have to first perform the JasperReports Server upgrade, then get the JasperReports Server Source Packages, apply the customizations in the UI source files, rebuild them and publish into the upgraded JasperReports Server.

Simba Driver Removal

As of release 8.0.4, Simba drivers are removed from JasperReports Server.

If you have any resources that use or depend on any of the following Simba Drivers:

•athena-jdbc42 2.0.33.1003

•cassandra-jdbc42 2.0.13.1014

•impala-jdbc42 2.6.26.1031

•neo4j-jdbc42

•spark-jdbc42 2.6.22.1040

then you must manually install publicly available drivers (for example, Athena and Cassandra have the same Simba drivers publicly available). For other drivers, you must obtain the drivers that are recommended by the database vendor. After installing new drivers, update the resources in JasperReports Server to use the new drivers.

Changes in 8.1 That May Affect Your Upgrade

- Users are able to upgrade from 8.0 Compact to 8.1 Compact using samedb and newdb.
- Users are able to upgrade from 8.0 Split to 8.1 Split using samedb and newdb.
- Users will not be able to upgrade:
    - From 8.0 Compact to 8.1 Split.
    - From 8.0 Split to 8.1 Compact.

If users need 8.1 Split installations but they are on 8.0 Compact, the required upgrade path is to:

1. Upgrade 8.0 Compact to 8.1 Compact.
2. Then, migrate from 8.1 Compact to 8.1 Split.

For more information on these installation options, see the *Installation* and *Upgrade* guides.

Newdb Upgrade Note

Currently, the **js-upgrade-newdb.sh/bat** script does not import the access, audit, monitoring data when upgrading.

However, once the upgrade process has completed, you can use the **JRS UI - Import** page to reimport the JRS export file that was passed in with the newdb script and select the checkboxes for including access, audit, and monitoring data. Once this import is completed, then the access, audit, monitoring data exist in the new database/release.

Changes in 8.0 That May Affect Your Upgrade

Split Installation Updates

The Split installation has been introduced from this release. The Audit, Access, and Monitoring events can be moved to a different audit database using the Split installation or

upgrade, which improves the performance of JasperReports server.

You have the option to choose from the following installations:

- Compact installation: The Repository, Audit, Access, and Monitoring tables are created in the repository database.
- Split installation: The Repository tables are created in the repository database. The Audit, Access, and Monitoring tables are created in a different audit database other than the repository database.

The default installation is the Compact installation.

With this release, the **jiaccessevent.user_id** type has been changed from numeric (integer data type) to a field (text data type). Due to this change, after the upgrade to version 8.0, you need to recreate any visualizations or domains that have the **user_id** column to display the data of the column.

Changes in 7.8 That May Affect Your Upgrade

Chrome/Chromium Updates

In the 7.8 release, the JavaScript engine is switched to Chrome/Chromium from PhantomJS/Rhino. JasperReports Server now uses the Chromium JavaScript engine to export reports and dashboard to PDF and other formats. PhantomJS/Rhino support has been removed.

You need to install and configure Chrome/Chromium to export the reports and dashboards to PDF and other output formats.

If you choose to continue the installation without Chrome/Chromium, reports and dashboards cannot be exported to PDF, DOCX, and other output formats.

The configuration properties have been updated to support the Chrome/Chromium configuration.

For information about configuring Chrome/Chromium in JasperReports Server, see the *JasperReports Server Administrator Guide*.

Changes in 7.5 That May Affect Your Upgrade

Driver Updates

In the 7.5 release, the Simba JDBC drivers for Spark and Impala have been updated. By default, the new release supports the new JDBC drivers, and the old drivers cannot be used. You should update your data sources to use the new driver. For more information, see the *JasperReports Server Administrator Guide*.

The drivers have been replaced due to vulnerabilities from third-party libraries. Update your data sources to use the new drivers.

Using the Old Impala Driver

If you want to continue using the Impala driver that was previously available from the community website, modify the install as described below.

Add the following files to the <js-install>/WEB-INF/lib directory:

- Curator-client-2.6.0.jar
- Curator-framework-2.6.0.jar
- Curator-recipes-2.6.0.jar
- Hive-metastore-1.2.2.jar
- Hive-service-1.2.2.jar
- Impala-jdbc4-1.0.44.1055.jar
- Libfb303-0.9.3.jar

If you do not add the files listed, data sources that use the old Impala driver causes errors when running reports that rely on them.

Using the Old Spark Driver

If you want to continue using the Spark driver that was previously available from the community website, modify the install as described below.

Add the following files to the <js-install>/WEB-INF/lib directory:

- Curator-client-2.6.0.jar
- Curator-framework-2.6.0.jar
- Curator-recipes-2.6.0.jar
- Hive-metastore-1.2.2.jar
- Hive-service-1.2.2.jar
- Spark-jdbc4-1.1.1.1001.jar
- Libfb303-0.9.3.jar

If you do not add the files listed, data sources that use the old Spark driver causes errors when running reports that rely on them.

Changes to the Jaspersoft MongoDB Query Language

The Jaspersoft MongoDB Query Language has been updated to reflect changes in the MongoDB driver:

- All aggregate commands must be updated to the new API-driven query syntax.
- All other command-driven queries (queries that use `runCommand`) are deprecated. If you want to use your queries in a future release, you should update them to the new syntax.

See the language reference for more information.

Encryption Keys

JasperReports Server 7.5 streamlines how it manages the encryption keys it uses to protect sensitive data inside and outside of the server. There is no more any need to configure the encryption keys because all keys are generated automatically during the installation and

stored in a central keystore. The keys are used transparently whenever the server stores passwords internally or exports sensitive data. And as long as the same user performs the upgrade, the upgrade scripts have access to the same keys in the keystore.

The new keys are backward compatible with the default keys from previous servers. However, there are possible cases when you need to manage keys during an upgrade. For example, if you do not have access to the user who installed the 7.5 instance, you may not be able to access the keystore anymore.

If you are in this situation, you should plan your upgrade as follows:

1. Before starting, back up your original 7.5 server.

2. Then export everything from your running 7.5 server with the following command:

```
cd <js-install-7.5>/buildomatic
js-export.sh --everything --output-zip js-7.5-export.zip --genkey
```

This encrypts the export with the key that is displayed on the console output:

```
Secret Key: 0xb1 0x44 0x72 0x0a 0xe9 0x5b 0x39 0xf5 0x87 0x5c 0xa9 0x1b
0x99 0x9d 0x14 0x4c
Key Alias (UUID): 9e41cd54-31da-43aa-84c2-638a7d0b47b8
```

3. Proceed with the upgrade and installation of the new server, but without migrating your data.

4. Import the key into your upgraded server with the following command.

```
./js-import.sh --input-key "0xb1 0x44 0x72 0x0a 0xe9 0x5b 0x39
                 0xf5 0x87 0x5c 0xa9 0x1b 0x99 0x9d 0x14 0x4c"
               --keyalias 9e41cd54-31da-43aa-84c2-638a7d0b47b8
               --keyalg AES --keypass NewKeyPassword
```

5. Proceed with the migration of your data to the upgraded server, or manually import your catalog to the upgraded server. As the data is imported, it is decrypted with the given key, and re-encrypted with the server's new keys.

6. Once the server is ready for production, back up your data and the new keystore once again.

For more information and procedures for importing keys, see the *JasperReports Server Security Guide*.

Theme Changes

The look and feel of the JasperReports Server web interface has been redesigned to modernize the application's appearance. To accomplish this, markup and styles have been modified. As a result of these modifications, custom themes developed for the previous interface need to be updated for the new interface. The main changes are in the banner, body, and home page.

The following table lists the changes made to the user interface, except for the changes to the home page. The changes to the home page are extensive. Instead of attempting to update an existing home page, you should reimplement the home page in the new default theme.

If you have not customized the user interface, these changes do not affect you.

Banner

Element	Classname and Modifications	File	Notes

Banner	`.banner`
	
	Changed `background-color`, `font-family` and `height`.
		containers.css	Default value:
			
			`background-color: #062e79`
			`font-family: source_sans_proregular`
			`height: 40px`

Body	`#frame` Changed the `top` value to fit the body of the application between the banner and footer without	containers.css	Default value: `top: 40px`

Element	Classname and Modifications	File	Notes
	overlap.		

Banner Logo	`#logo`
	Changed `width` and `height`.
	Responsive behavior was added to the banner. There is now a breakpoint at which the logo shrinks in size (1100 px) and a breakpoint at which it becomes hidden (980 px).
		theme.css	Default values:
			`height: 23px`
			`width: 200px`
			Breakpoint from 981-1100 px:
			`width: 150px`
			980 px and below:
			`display: none`

Banner Main Navigation home icon	`.menu.primaryNav #main_home .wrap > .icon`
	New sprites for `background-image`: one for standard-resolution displays and one for high-resolution displays.
		containers.css	Default value:
			`background-image: url(images/banner_icons_sprite@1x.png)`
			High-resolution value:
			`background-image: url(images/banner_icons_sprite@2x.png)`

Banner Main Navigation Item text	`.menu.primaryNav .wrap`
	Enlarged `font-size`. Changed `height` and `line-height` to be 1 px shorter than `.banner`.
		containers.css	Default values:
			`font-size: 14px`
			`height: 39px`
			`line-height: 39px`

Banner Main Navigation Item arrow	`.menu.primaryNav .node > .wrap > .icon` New sprites for `background-image`: one for standard-resolution	containers.css	Default values: `background-image: url (images/disclosur`

Element	Classname and Modifications	File	Notes

icon	displays and one for high-resolution displays. Changed `height` of icon container.		`e_icons_sprite@1x.png)`
			height: `16 px`
			
			High-resolution value:
			`background-image: url (images/disclosure_icons_sprite@2x.png)`

Banner Metadata container	#metalinks
	
	Changed `height` to be 1 px shorter than `.banner`. Increased `margin-right` to accommodate search box.
	
	With the addition of responsive behavior, the `margin-right` value changes at certain breakpoints to accommodate a smaller search box.
		theme.css	Default values:
			
			height: `39px`
			margin-right: 270 px
			
			Breakpoint from 821-1100 px:
			`margin-right: 200px`
			
			Breakpoint from 751-820 px:
			`margin-right: 140px`

Banner Metadata text	`#metalinks li` Enlarged `font-size`. Increased `line-height` to vertically center text in the banner.	theme.css	Default values: font-size: `14 px` line-height: `39 px`
Banner Search container	`#globalSearch.control.searchLockup` Increased `width` of container.	controls.css	Default value: `width: 250px` Breakpoint from

Element	Classname and Modifications	File	Notes

	Responsive behavior was added to the banner. There are now breakpoints at which the search container shrinks in width and a breakpoint at which it becomes hidden.		821-1100 px:
			`width: 180px`
			Breakpoint from 751-820 px:
			`width: 100px`
			750 px and below:
			`display: none`

Banner Search input wrapper	`#globalSearch.control.searchLockup > .wrap` Increased `height` of input wrapper.	controls.css	Default values: height: 28 px

Banner Search input	`#globalSearch.control.searchLockup > .wrap > input [type=text]`
	Responsive behavior was added to the banner. There are now breakpoints at which the search input shrinks in width and a breakpoint at which it becomes hidden.
		controls.css	Default value:
			`width: 200px`
			Breakpoint from 821-1100 px:
			`width: 130px`
			Breakpoint from 751-820 px:
			`width: 80px`
			750 px and below:
			`display: none`

Banner Search button icon	`#globalSearch .button.search` New sprites for `background-image`: one for standard-resolution displays and one for high-	controls.css	Default value: `background-image: url(images/search_icons_sprite@1x.png)`

Element	Classname and Modifications	File	Notes
	resolution displays.		High-resolution value: `background-image: url(images/search_icons_sprite@2x.png)`

Ad Hoc Designer

Extensive changes have been made to the look and feel of the Ad Hoc Designer. Although there are too many changes to document fully, the following table lists the basic elements that have changed.

Element	Classname and Modifications	File	Notes
Page Title	`#display > .column.decorated > .content > .header` This element has been removed and replaced with the new `.pageHeader` element.	pages.css	

Data and Filters Panel Headers	`#designer .column.decorated > .content > .header`
	Removed bottom border, changed background-color, and increased `height`.
		pageSpecific.css	Default values:
			`background-color: #d6d5d5`
			`border-bottom: 0`
			`height: 32px`

Data and Filters	`#designer .column.decorated >`	pageSpecific.css	Default values:

Element	Classname and Modifications	File	Notes

Panel Headers Title Text	`.content > .header > .title`
	Changed `color` and `font-family`. Increased `font-size` and `line-height`.
			`color: #333333`
			`font-family: source_sans_proregular`
			`font-size: 15px`
			`line-height: 32px`

Panel Minimize Button	`#designer .button.minimize`
	New sprites for `background-image`: one for standard-resolution displays and one for high-resolution displays. Changed `height` and `width`. Added a `background-color`.
		pageSpecific.css	Default values:
			`background-color: #999999`
			`background-image: url (images/disclosure_indicators_icons_sprite@1x.png)`
			`height: 32px`
			`width: 14px`
			High-resolution value:
			`background-image: url (images/disclosure_indicators_icons_sprite@2x.png)`

Panel Options Button
Panel Section Options Button
	`.header > .button.mutton, #filter-container .title .button.mutton`
	New sprites for `background-image`: one for standard-resolution displays and one for high-resolution displays. Changed `height` and `width`.
		pageSpecific.css	Default values:
			`background-image: url (images/disclosure_indicators_icons_sprite@1x.png)`
			`height: 32px`
			`width: 22px`
			.
			High-resolution value:
			background-image: url (images/disclosure_indicators_icons_sprite@2x.png)

Element	Classname and Modifications	File	Notes

Panel Section Headers	`#designer #availableFields .dimension .header`, `#designer #availableFields .measure .header`, `#level-container .pod-header`, `#filter-container .header`, `#expression-container .header`
	
	Changed `background-color` and `font-family`, increased `height`, and removed the bottom border.
		pageSpecific.css	Default values:
			
			`background-color: #ebebeb`
			`border-bottom: none`
			`font-family: source_sans_proregular`
			`height: 32px`

Toolbar	`#designer .toolbar` Increased `height`.	pageSpecific.css	Default values: `height: 32px`
Toolbar Buttons	`button.capsule` Increased `width`.	buttons.css	Default value: `width: 32px`
Toolbar Buttons with down arrow	`button.capsule.mutton` Increased `width`.	buttons.css	Default value: `width: 36px`
Toolbar Button Icons	`.button.capsule .indicator` New sprites for `background-image`: one for standard-resolution displays and one for high-resolution displays.	pageSpecific.css	Default values: `background-image: url (images/disclosure_indicators_icons_sprite@1x.png)` . High-resolution value:

Element	Classname and Modifications	File	Notes
			`background-image: url (images/disclosure_ indicators_icons_ sprite@2x.png)`

Report Viewer

Changes have been made to the general look and feel of the Report Viewer. The following table lists the basic elements that have changed.

Element	Classname and Modifications	File	Notes
Page Title	`#reportViewer #reportViewFrame > .content > .header` This element has been removed and replaced with the new `.pageHeader` element.	pages.css	
Toolbar	`#reportViewer .toolbar` Increased `height`.	pageSpecific.css	Default value: `height: 32px`
Toolbar Buttons Container	`#reportViewer .toolbar > .buttonSet` Increased `height`.	pageSpecific.css	Default value: `height: 31px`
Toolbar Button Icons	`#designer .toolbar .button .icon` New sprites for `background-image`: one for standard-resolution	pageSpecific.css	Default values: `background-image: url (images/button_ action_icons_ sprite@1x.png)`

Element	Classname and Modifications	File	Notes
	displays and one for high-resolution displays.		. High-resolution value: `background-image: url (images/button_ action_icons_ sprite@2x.png)`

Options Panel Header	`#reportViewer #inputControlsForm > .content > .header`
	Removed bottom border, changed background-color, and increased `height`.
		pageSpecific.css	Default values:
			`background-color: #d6d5d5`
			`border-bottom: 0`
			`height: 32px`

Options Panel Header Title Text	`#reportViewer #inputControlsForm > .content > .header > .title`
	Changed `color` and `font-family`. Increased `font-size` and `line-height`.
		pageSpecific.css	Default values:
			`color: #333333`
			`font-family: source_ sans_proregular`
			`font-size: 15px`
			`line-height: 32px`

Options Panel Minimize Button	`#reportViewer #inputControlsForm .button.minimize`
	New sprites for `background-image`: one for standard-resolution displays and one for high-resolution displays. Changed `height` and `width`. Added a `background-color`.
		pageSpecific.css	Default values:
			`background-color: #999999`
			`background-image: url (images/disclosure_ indicators_icons_ sprite@1x.png)`
			`height: 32px`
			`width: 14px`
			.
			High-resolution value:

Element	Classname and Modifications	File	Notes
			`background-image: url (images/disclosure_ indicators_icons_ sprite@2x.png)`

Dashboard Designer

Extensive changes have been made to the look and feel of the Dashboard Designer. Although there are too many changes to document fully, the following table lists the basic elements that have changed.

Element	Classname and Modifications	File	Notes
Page Title	`.column.decorated > .content > .header` This element has been removed and replaced with the new `.pageHeader` element.	pages.css	

Available Content Panel Header	`.dashboardDesigner .column.decorated > .content > .header`
	
	Removed bottom border, changed background-color, and increased `height`.
		designer.css	Default values:
			
			`background-color: #d6d5d5`
			`border-bottom: 0`
			`height: 32px`

Available Content Panel Header Title	`#display.dashboardDesigner .column.decorated > .content > .header > .title`
	
	Changed `color` and `font-family`. Increased `font-size` and `line-height`.
		designer.css	Default values:
			
			`color: #333333`
			`font-family: source_ sans_proregular`
			`font-size: 15px`
			`line-height: 32px`

Element	Classname and Modifications	File	Notes

Available Content Panel Minimize Button	`.dashboardDesigner .button.minimize`
	
	New sprites for `background-image`: one for standard-resolution displays and one for high-resolution displays. Changed `height` and `width`. Added a `background-color`.
		designer.css	Default values:
			
			`background-color: #999999`
			`background-image: url (images/disclosure_indicators_icons_sprite@1x.png)`
			`height: 32px`
			`width: 14px`
			
			High-resolution value:
			
			`background-image: url (images/disclosure_indicators_icons_sprite@2x.png)`

Available Content Panel Section Headers	`.dashboardDesigner .dashboardSidebar .panel.collapsiblePanel > .header`
	
	Removed bottom border and increased `height`. Changed `background-color` and `font-family`.
		designer.css	Default values:
			
			`background-color: #ebebeb`
			`border-bottom: none`
			`font-family: source_sans_proregular`
			`height: 32px`

Available Content Panel Section Headers Title	`.dashboardDesigner .dashboardSidebar .panel.collapsiblePanel > .header > .title`
	
	Changed `color`. Increased `font-size`, height, and `line-height`.
		designer.css	Default values:
			
			`color: #333333`
			`font-size: 13px`
			`height: 32px`
			`line-height: 33px`

Available Content Panel	`.dashboardDesigner .collapsiblePanel > .header > .buttonIconToggle`	designer.css	Default values: `background-image: url`

Element	Classname and Modifications	File	Notes

Section Headers Toggle Button	New sprites for `background-image`: one for standard-resolution displays and one for high-resolution displays. Increased `height` and `width`.		`(images/disclosure_indicators_icons_sprite@1x.png)`
			`height: 32px`
			`width: 22px`
			.
			High-resolution value:
			background-image: `url (images/disclosure_indicators_icons_sprite@2x.png)`

Available Content Panel Section Options Button	`.header > .button.mutton`
	New sprites for `background-image`: one for standard-resolution displays and one for high-resolution displays. Increased `height` and `width`.
		containers.css	Default values:
			`background-image: url (images/disclosure_indicators_icons_sprite@1x.png)`
			`height: 32px`
			`width: 22px`
			.
			High-resolution value:
			`background-image: url (images/disclosure_indicators_icons_sprite@2x.png)`

Dashboard Canvas	`.dashboardCanvas > .content > .body` Changed `background-color`.	canvas	Default values: `background-color: #ffffff`

Changes in 7.2 That May Affect Your Upgrade

Removal of Legacy Dashboards

JasperReports Server 7.2 removes support for legacy dashboards, created in JasperReports Server version 5.6.2 and earlier. If your JasperReports Server repository contains any legacy dashboards, a warning message appears during the upgrade. If you continue with the upgrade, your legacy dashboards are permanently deleted. You cannot roll back this operation after it is done.

If you have any legacy dashboards you want to keep, you should recreate them as new dashboards before upgrading. For information on creating dashboards using the Dashboard Designer, see the *JasperReports Server User Guide*.

Changes to the Login Page

The layout of the login page changed in JasperReports Server 7.2. There were no changes to the CSS classes, but some default values were changed. If you have customized the login page, test your customizations to ensure they have the desired effect in 7.2, and make any necessary changes. If you have not customized the login page, this change does not affect you.

Spring Security Upgrade

JasperReports Server uses the Spring Security framework to implement security throughout the product. In JasperReports Server 7.2, the Spring Security framework was updated to Spring Security 4.2. For many users, this upgrade has no impact. However, you may need to make some changes if you have implemented the following:

- External authentication–If you have implemented external authentication or Single Sign-on in your server implementation, you need to update your implementation:
    - If you implemented external authentication using one of the sample files included in the project, you need to reimplement your changes in the updated sample files included in JasperReports Server 7.2.

    - If you have implemented a custom external authentication solution, you need to migrate your solution to the new framework.
- Customizations–If you have customized the server using Spring Security classes, you need to migrate your solution to the new framework.

Migrating External Authentication Sample Files

If you have implemented external authentication using one of the sample-applicationContext-<customName>.xml files located in the <js-install>/samples/externalAuth-sample-config directory, migrate your changes to JasperReports Server 7.2 as follows:

1. Prior to upgrade, back up your existing applicationContext-<customName>.xml (for example, applicationContext-externalAuth-LDAP.xml), located in the <js-webapp>/WEB-INF directory of your previous version of JasperReports Server.

2. Update your server installation to JasperReports Server 7.2, as described in the *JasperReports Server Upgrade Guide*.

3. In the new installation, locate the sample file that corresponds to the file you implemented previously. For example, if you implemented applicationContext-externalAuth-LDAP.xml, locate <js-install-7.2>/samples/externalAuth-sample-config/sample-applicationContext-externalAuth-LDAP.xml.

4. Rename the JasperReports Server 7.2 sample file to remove the sample- prefix. For example, rename sample-applicationContext-externalAuth-LDAP.xml to applicationContext-externalAuth-LDAP.xml.

5. Configure the properties in the new sample file to match the properties in your existing sample file. To do this:

a. Locate each bean that you have modified in the previous version.

b. Find the same bean in the JasperReports Server 7.2 sample. The names of the beans have not changed between versions.

c. Copy or reenter the properties you need for your server, taking care not to copy over class names or class packages.

Although the bean names are the same in the JasperReports Server 7.2 sample files, the name and package of the class in many bean definitions have changed. Make sure not to overwrite the new names with the old ones.

d. Save the JasperReports Server 7.2 sample file.

e. Rename the JasperReports Server 7.2 sample file to remove the sample- prefix. For example, rename sample-applicationContext-externalAuth-LDAP.xml to applicationContext-externalAuth-LDAP.xml.

f. Place the modified file in the <js-webapp-7.2>/WEB-INF directory.

Migrating Customizations

The Spring Security codebase was significantly restructured from 3.x to 4.x. Many classnames have changed and other classes were moved to different packages. In addition, many classes were deprecated. At a minimum, you need to update the names and paths of the Spring Security classes you reference in any customizations you have made to JasperReports Server For information on updating your customizations, see the *Spring Security migration guide*:

https://docs.spring.io/spring-security/site/migrate/current/3-to-4/html5/migrate-3-to-4-xml.html

For specific information about migrating from deprecated classes in 4.x, see the Deprecations topic in the same document.

Changes in 7.1 That May Affect Your Upgrade

Changes to the Login Page

The layout of the login page changed in JasperReports Server 7.1. There were no changes to the CSS classes, but some default values were changed. If you have customized the login page, you should make sure your customizations still have the desired effect in 7.1, and make any necessary changes.

If you have not customized the Login page, this change will not affect you.

Changes to Absolute Paths in Reports

Prior to 7.1, you could use absolute paths in reports, for example:

repo:/organizations/organization_1/reports/main_jrxml.jrxml

If you have a reference to an image, a subreport, or other resource that has an absolute path, or if you use a $P{} parameter which later gets resolved as an absolute path, the report will cause an error. You need to update the report and use a path which is visible to a tenant user. Consider using relative path, or the public folder in case if reports needs to work for several tenants.

Changes in 6.4 That May Affect Your Upgrade

Removal of the Impala Connector

JasperReports Server provides new and updated drivers for various databases. In the 6.2 release, the previous connector for Impala that had been available on the Jaspersoft community website was replaced with two new options:

- TIBCO Impala JDBC driver (also called Progress)
- Simba JDBC driver (Cloudera-endorsed JDBC interface)

By default, the new release supports the new JDBC drivers, and the old Impala connector cannot be used. You should update your Impala data sources to use the new drivers. For more information, see the *JasperReports Server Administrator Guide*.

If you wish to continue using the Impala connector that was previously available from the community website, modify the install as described below.

1. Add the following files to the <js-install>/WEB-INF/lib directory:
    - hive-service-0.12.0-cdh5.1.3.jar
    - zookeeper-3.4.5-cdh5.1.3.jar
    - avro-1.7.5-cdh5.1.3.jar
    - commons-compress-1.4.1.jar
    - hadoop-core-1.2.1.jar
    - hive-ant-0.12.0-cdh5.1.3.jar
    - hive-common-0.12.0-cdh5.1.3.jar
    - hive-exec-0.12.0-cdh5.1.3.jar

- hive-jdbc-0.12.0-cdh5.1.3.jar
- jasperserver-hive-connector-bugfix-SNAPSHOT.jar
- js-hive-datasource-1.2.1-cdh5.jar
- paranamer-2.3.jar
- parquet-hadoop-bundle-1.2.5-cdh5.1.3.jar
- xz-1.0.jar

2. Delete the file applicationContext-HiveDatasource.xml from the <js-install>/WEB-INF directory:

If you do not add the files listed, data sources that use the old Impala connector will cause errors when running reports that rely on them.

Changes in 6.2.1 That May Affect Your Upgrade

Removal of the Impala Connector

JasperReports Server provides new and updated drivers for various databases. In this release, the previous connector for Impala that was available on the Jaspersoft community website is replaced with two new options:

- TIBCO Impala JDBC driver (also called Progress)
- Simba JDBC driver (Cloudera-endorsed JDBC interface)

By default, the new release supports the new JDBC drivers, and the old Impala connector cannot be used. You should update your Impala data sources to use the new drivers. For more information, see the *JasperReports Server Administrator Guide*.

If you wish to continue using the Impala connector from the community website, you must replace the following JAR files in the .../WEB-INF/lib directory:

Replace	With This
hive-service-0.13.1.jar	hive-service-0.12.0-cdh5.1.3.jar

Replace	With This
zookeeper-3.4.6.jar	zookeeper-3.4.5-cdh5.1.3.jar

Unless you replace the files listed in the table, data sources that use the old Impala connector will cause errors when running reports that rely on them.

Changes in 6.2 That May Affect Your Upgrade

Renaming of Ad Hoc Templates

Due to minor changes to Ad Hoc templates in 6.2, the default template files have been renamed, for example, actual_size.510.jrxml has been renamed actual_size.620.jrxml. During upgrade, templates with an earlier version number are overwritten by the new template. If you have customized the default template and kept the same file name, your changes will be overwritten. To avoid this, make a copy of your customized template with a unique name and upload it to your template directory (by default, Public > Templates) using Add Resource > File > Style Template.

In general, if you want to customize the Ad Hoc templates, we recommend you rename the existing template and set the new template as a default, rather than overwriting the existing template. See the *JasperReports Server Administrator Guide* for more information.

Changes in 6.1 That May Affect Your Upgrade

Changes to Themes

The look and feel of the JasperReports Server web interface has been redesigned to modernize the application's appearance. To accomplish this, markup and styles have been modified. As a result of these modifications, custom themes developed for the previous interface need to be updated for the new interface.

The following table lists the changes made to the user interface and describes some of the steps necessary to update custom themes in overrides_custom.css. The main changes are in the banner, body, footer, and login page. The changes to the login page are extensive. Instead of attempting to update an existing login page, you should reimplement the login page in the new default theme.

For information on developing new themes, see the *JasperReports Server Administrator Guide* and the *JasperReports Server Ultimate Guide*.

Updating Themes in JasperReports Server 6.1

Element	Classname and Modifications	File	Notes
Banner	`.banner` Give custom value to `height`	containers.css	Default value: height: 32 px
Body	`#frame` Set custom `top` and `bottom` values that position the body of the application between the banner and footer without overlap	containers.css	Default value: top: 32 px bottom: 17 px This value needs to be equal to or greater than the height of `.banner` The bottom position needs to be adjusted only if the height of the footer is changed
Banner Logo	`#logo` Give custom values to height and width that match the dimensions of your logo Adjust margins around the logo if needed	theme.css	Default values: height: 22 px width: 176 px margin-top: 6 px margin-right: 4 px margin-bottom: 0 margin-left: 8 px
Banner Main	`.menu.primaryNav.wrap`	containers.css	height: 31 px line-height: 31 px

Element	Classname and Modifications	File	Notes
Navigation	Set height and line-height to 1 px shorter than `.banner`		
Banner Main Navigation Home icon	`.menu.primaryNav #main_home.wrap > .icon` Set the height to be the same as .banner Set values for width and background-position to fit your image.	containers.css	height: 32 px width: 14 px background-position: 0 -164 px background-position: 0 -163 px (IE8-9)
Banner Main Navigation Item arrow icon	`.menu.primaryNav .node > .wrap > .icon` Set height to your desired value, with the maximum value being the same height measurement as the .banner element. Set background-position and width to a value that properly displays the default or your custom image.	containers.css	height: 32 px background-position: left -79 px width: 11 px

Banner Main Navigation Item arrow icon	`.menu.primaryNav .wrap.over`
	`.menu.primaryNav .wrap.pressed`
	Set background-position to a value that properly displays the default or your custom image.
		containers.css	background-position is not explicitly defined. The value is cascaded from `.menu.primaryNav .node > .wrap > .icon`
			This only needs to be adjusted if you want a different color disclosure indicator for the pressed and over states of the

Element	Classname and Modifications	File	Notes
			main menu links.
Banner Search container	`#globalSearch.searchLockup` Set the margin-top to a desired value that will vertically center it within the banner.	controls.css	margin-top: 5 px
Banner Metadata	`#metalinks li` Set the line-height to the desired value that will vertically center it within the banner.	themes.css	line-height: 20 px
Footer	`#frameFooter` Set the height if you want it to be anything other than the default value.	containers.css	height: 17 px
Login page	Reimplement in a new theme.		

Working With JDBC Drivers

This section describes how to set up your installation to use a driver other than the default driver.

Open Source JDBC Drivers

For open source JDBC drivers, buildomatic is set up to use a single default driver. If you want to use a driver other than the default driver, you can modify the buildomatic property files that determine the default JDBC driver.

The buildomatic JDBC driver property files are set up to point to a specific driver jar. This allows for multiple driver jar files in the same `buildomatic/conf_source/db/<dbType>/jdbc` folder. During the installation procedure only the default driver jar is copied to your application server.

If you want to use a newer JDBC driver version or a different JDBC driver, you can modify the buildomatic properties seen in your `default_master.properties` file.

PostgreSQL Example

The buildomatic/conf_source/db/postgresql/jdbc folder contains the following driver file:

```
postgresql-42.2.20.jar
```

If, for instance, you want to change the default driver used by PostgreSQL from type jdbc4 to jdbc3, edit your default_master.properties file:

Overlay upgrade:	`<overlay-folder>/buildomatic/default_master.properties`
Other upgrade:	`<js-install>/buildomatic/default_master.properties`

Uncomment and change:

```
# maven.jdbc.version=42.2.5
```

To:

```
maven.jdbc.version=9.2-1002.jdbc3
```

When you next run a buildomatic command, such as deploy-webapp-pro, the jdbc3 driver will be copied to your application server.

MySQL Example

The `buildomatic/conf_source/db/mysql/jdbc` folder contains this driver file:

```
mariadb-java-client-2.5.3.jar
```

If, for instance, you want to use a JDBC driver built and distributed by the MySQL project, such as `mysql-connector-java-5.1.43-bin.jar`, you first need to download the driver from the MySQL Connector/J download location:

```
https://dev.mysql.com/downloads/connector/j/
```

Next, change your buildomatic configuration properties to point to this new driver.

Edit your default_master.properties file:

Overlay upgrade:	`<overlay-folder>/buildomatic/default_master.properties`
Other upgrade:	`<js-install>/buildomatic/default_master.properties`

Uncomment and change:

```
# jdbcDriverClass=com.mysql.jdbc.Driver
```

```
# jdbcDataSourceClass=com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSource
```

```
# maven.jdbc.groupId=mysql
```

```
# maven.jdbc.artifactId=mysql-connector-java
```

```
# maven.jdbc.version=5.1.43-bin
```

To:

```
jdbcDriverClass=com.mysql.jdbc.Driver
```

```
jdbcDataSourceClass=com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSourc
e

maven.jdbc.groupId=mysql

maven.jdbc.artifactId=mysql-connector-java

maven.jdbc.version=5.1.43-bin
```

Installing Database Vendor JDBC Drivers

JasperReports Server no longer includes JDBC drivers for the following commercial databases:

- DB2
- Oracle
- SQL Server

You can download the driver supplied by the database vendor as described below. To do this, you must first obtain and install the driver you want, then copy that driver into buildomatic.

Oracle Example

1. Copy your Oracle driver to the following directory:

Overlay upgrade:	`<overlay-folder>/buildomatic/conf_source/db/oracle/jdbc/`
Other upgrade:	`<js-install>/buildomatic/conf_source/db/oracle/jdbc/`

SQL Server Example

1. Copy your SQL Server driver to the following directory:

Overlay	`<overlay_folder>/buildomatic/conf_source/db/sqlserver/jdbc`

upgrade:	
Other upgrade:	`<js_install>/buildomatic/conf_source/db/sqlserver/jdbc`

2. Change to the <js_install>/buildomatic directory and open default_master.properties in a text editor.

3. Go to the Additional Settings section in this file.

4. Go to the first setup item, Setup Standard SQL Server JDBC Driver.

5. Uncomment the required properties and enable your driver. The following example shows how to set up default_master.properties to point to a driver named mssql-jdbc-6.4.0.jre8.jar:

```
# 1) Setup Standard SQLServer JDBC Driver
#
# Uncomment and modify the value to native
jdbcDriverMaker=native
#
# Uncomment and modify the value in order to change the default
# Driver will be found here: <path>/buildomatic/conf_
source/db/sqlserver/native.jdbc
#
maven.jdbc.groupId=sqlserver
maven.jdbc.artifactId=sqljdbc
maven.jdbc.version=6.4.0.jre8
```

6. Save the default_master.properties file.

DB2 Example

1. Copy your DB2 driver to the following directory:

Overlay upgrade:	`<overlay_folder>/buildomatic/conf_source/db/db2/jdbc`
Other upgrade:	`<js_install>/buildomatic/conf_source/db/db2/jdbc`

2. Change to the <js_install>/buildomatic directory and open default_master.properties in a text editor.

3. Go to the Additional Settings section in this file.

4. Go to the first setup item, Setup Standard DB2 JDBC Driver.

5. Uncomment the required properties and enable your driver.

```
# 1) Setup Standard DB2 JDBC Driver
#
# Uncomment and modify the value to native
jdbcDriverMaker=native
#
# Uncomment and modify the value in order to change the default
# Driver will be found here: <path>/buildomatic/conf_
source/db/db2/native.jdbc
#
maven.jdbc.groupId=ibm
maven.jdbc.artifactId=db2jcc
maven.jdbc.version=9.7
```

6. Add the following additional properties, setting the correct values for your installation. For example:

```
db2.driverType=4
db2.fullyMaterializeLobData=true
db2.fullyMaterializeInputStreams=true
db2.progressiveStreaming=2
db2.progressiveLocators=2
dbPort=50000
js.dbName=JSPRSRVR
sugarcrm.dbName=SUGARCRM
foodmart.dbName=FOODMART
```

7. Save the default_master.properties file.

Jaspersoft Documentation and Support Services

For information about this product, you can read the documentation, contact Support, and join Jaspersoft Community.

How to Access Jaspersoft Documentation

Documentation for Jaspersoft products is available on the Product Documentation website, mainly in HTML and PDF formats.

The Product Documentation website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The documentation for this product is available on the JasperReports® Server Product Documentation page.

How to Access Related Third-Party Documentation

When working with JasperReports® Server, you may find it useful to read the documentation of the following third-party products:

How to Contact Support for Jaspersoft Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our product Support website.
- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the product Support website. If you do not have a username, you can request one by clicking **Register** on the website.

How to Join Jaspersoft Community

Jaspersoft Community is the official channel for Jaspersoft customers, partners, and employee subject matter experts to share and access their collective experience. Jaspersoft Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from Jaspersoft products. In addition, users can submit and vote on feature requests from within the Jaspersoft Ideas Portal. For a free registration, go to Jaspersoft Community.

Legal and Third-Party Notices

SOME CLOUD SOFTWARE GROUP, INC. (“CLOUD SG”) SOFTWARE AND CLOUD SERVICES EMBED, BUNDLE, OR OTHERWISE INCLUDE OTHER SOFTWARE, INCLUDING OTHER CLOUD SG SOFTWARE (COLLECTIVELY, “INCLUDED SOFTWARE”). USE OF INCLUDED SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED CLOUD SG SOFTWARE AND/OR CLOUD SERVICES. THE INCLUDED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER CLOUD SG SOFTWARE AND/OR CLOUD SERVICES OR FOR ANY OTHER PURPOSE.

USE OF CLOUD SG SOFTWARE AND CLOUD SERVICES IS SUBJECT TO THE TERMS AND CONDITIONS OF AN AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER AGREEMENT WHICH IS DISPLAYED WHEN ACCESSING, DOWNLOADING, OR INSTALLING THE SOFTWARE OR CLOUD SERVICES (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH LICENSE AGREEMENT OR CLICKWRAP END USER AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE SAME TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

Jaspersoft, JasperReports, Visualize.js, and TIBCO are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only. You acknowledge that all rights to these third party marks are the exclusive property of their respective owners. Please refer to Cloud SG’s Third Party Trademark Notices (https://www.cloud.com/legal) for more information.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: https://scripts.sil.org/OFL

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file for the availability of a specific version of Cloud SG software on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at https://www.tibco.com/patents.