JasperReports® Server

Administrator Guide

Version 9.0.0 | January 2024

Contents

Overview of JasperReports Server Administration

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 utilizing common metadata and provide shared services, such as security, a repository, and scheduling. The server exposes comprehensive public interfaces enabling seamless integration with other applications and the capability to easily add custom functionality.

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

- Easily create new 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 as PDFs in the doc subdirectory of your JasperReports Server installation. You can also access PDF and

HTML versions of these guides online 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 website covers topics for developers, system administrators, business users, and 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.

> This administrator guide describes features that are only available to users who have administrator roles. Many of the configuration procedures also assume you have access to the installed files on the host computer.

- 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.

This chapter contains the following sections:

- Overview of Organizations
- Overview of the Repository
- Overview of Users and Roles
- Overview of Security
- Administrator Login
- Administrator Pages

> 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're licensed to use, or to upgrade your license, contact Jaspersoft.

JasperReports Server is a component of both a community project and commercial offerings. Each integrates the standard features such as security, scheduling, a web services interface, and much more for running and sharing reports. Commercial editions provide additional features, including Ad Hoc charts, dashboards, Domains, auditing, and a multi-

organization architecture for hosting large BI deployments. This guide documents the features of the commercial edition.

Overview of Organizations

The commercial version of JasperReports Server is built on organizations, which are logical entities within the server that have their own users, roles, and branches of the repository. As with any business structure or hierarchy, organizations may have suborganizations, which may in turn have suborganizations, and so on.

In the default JasperReports Server installation, a single organization mimics the simple structure of older versions of JasperReports Server. If you want to deploy multiple organizations, you must be aware of certain design considerations.

Single Default Organization

After a default installation, JasperReports Server contains a single organization in which you can deploy your reports. For example, if you install the sample data, you will see a single organization that holds all sample resources, users, and roles.

Single organizations are designed to handle most business cases and are straightforward to administer. However, even with a single organization, the server still includes a root that represents the parent of the organization. For example, a system admin and an organization admin share administrative duties. Also, many features such as themes, attributes, and repository URIs can be defined at both the root and organization levels.

If your business needs more organizations, you will need to manage several levels of administrators and possibly create shared resources in the repository. The following sections provide use cases and explain the functioning of multiple levels of administrators.

Multiple Organizations

With multiple organizations, several logical entities share a server instance, but remain entirely isolated from each other. There are many use cases for defining multiple organizations in JasperReports Server. For instance:

- An application provider, such as a software-as-a-service (SaaS) company or a computer department, has a hosted application offered to many customers. It

integrates JasperReports Server in its application to offer dashboards, reports, and analysis. A number of common reports and data sources are useful across customers, but there are customer-specific reports, as well. Machines and databases are shared by customers, according to the provider's own architecture, but within the functionality provided by JasperReports Server, each customer is a separate organization. Customers can manage their own users in the hosted application, and the server maps the application's authentication scheme to the correct organization. The organization structure provides the full power of JasperReports Server to each of the provider's customers, while ensuring that their data and reports are secure.

- A company has many departments but wants them to share a common BI infrastructure. Corporate IT needs to deploy and maintain only a single instance of JasperReports Server, while each department is represented by an organization that manages its own users. For security and simplicity, the departments do not share databases, except in the case of sub-departments, such as Accounts Payable being a sub-department of Finance. Users access JasperReports Server directly, logging in with their department name and username. Organization administrators have defined the data sources and Domains specific to the needs of their department's users.

The organization feature is flexible enough to accommodate any combination of these scenarios and many others like it. In all cases, administrators can configure secure environments for any number of organizations, and end users experience a powerful BI platform tailored to their needs.

All organizations or hierarchies of organizations exist independently within a single instance of JasperReports Server. Neighboring organizations are isolated from one another, while parent organizations have full control of their suborganizations. Users can access only the data and resources in their own organizations or suborganizations, and administrators can define roles and set permissions to further control access.

Levels of Administrators

Each organization has an administrator who can manage users, roles, and repository permissions in that organization and any of its suborganizations.

There are essentially three levels of administration:

- The system administrator - Also called system admin. The default login name of the system admin is superuser. He exists at the root level, outside all organizations. The system admin manages the JasperReports Server installation, creates top-level

organizations, and configures server-wide settings. The system admin can create, modify, and delete users, roles, and repository objects in any organization.

- The administrator of a top-level organization - Also called organization admin. The organization admin manages all users, roles, and repository objects in an entire organization, including any suborganizations. The default login name of the organization admin is jasperadmin.

- The administrator of a suborganization - Functionally equivalent to an organization admin, but limited to the suborganization's users, roles, and repository objects. This administrator can be overridden by a top-level organization admin.

Overview of the Repository

The repository is a hierarchical structure of folders where administrators and users create and store resources for running reports. In its appearance and function, the repository resembles a file system with a structure of folders containing files. However, the repository is actually implemented as a database that is private to the server instance. As a result, it lacks a few of the functions of a file system.

Folder Structure

The repository is a tree structure composed of folders and resources. The root contains folders for the sample data installed with the server. The root of the repository tree structure is accessible only to the system admin logged in as superuser. The tree contains the folders for each organization's reporting resources, and folders for certain features of the server.
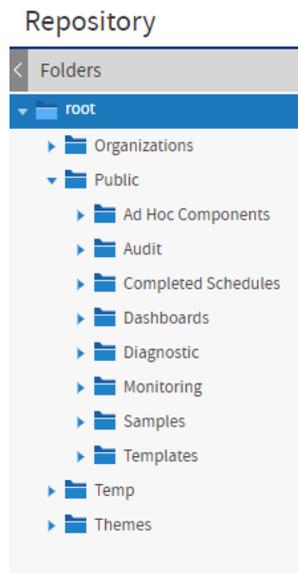
Figure 1: Root of the Repository Showing Default Folders

In general, we recommend that you avoid placing resources directly in the root or organization folder. Instead, use folders for various resource types, as in the sample data. The Themes folder contains files that control the look and feel of the user interface, as described in Themes.

The Organizations folder contains a folder for the default organization, named Organization. If you create more than one organization, each organization has its own folder, contained in a folder named for the organization. Each organization also contains a folder called Organizations where suborganizations are created.

Organizations located in root/Organizations are called top-level organizations. Organizations located within other organizations are called suborganizations.

JasperReports Server automatically restricts every user's access to their own organization's branch of the repository. System admins (superuser) can view and create folders in all organizations, while organization admins (jasperadmin) can view and create folders only in the organizations they administer. The following figure shows the system admin's view starting at root, and the organization admin's view starting in his organization.

System admin (superuser) view:
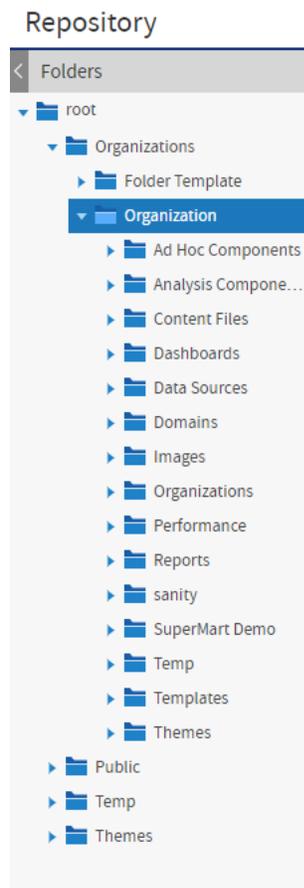
Organization admin (jasperadmin) view:
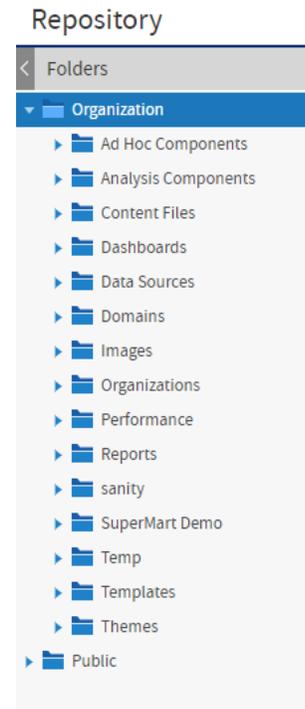
Figure 2: System Admin and Organization Admin Views of the Repository

The Public folder is a special system folder that appears at the root and in every organization folder. All instances are links to the same folder, so that its contents are shared with all organizations. The system admin should manage the Public folder and set permissions so that users can access shared resources (such as data sources, logos, and report templates) but not modify them.

Resources

Resources are stored in the repository and used as input for creating reports and performing analysis. Some resources, such as images, fonts, or JRXML files created in Jaspersoft Studio, are uploaded from files. Others, such as data sources and Domains, are created in JasperReports Server itself. Of course, dashboards, data views, and reports can

also be saved in the repository to be run as often as needed. Output such as PDF or HTML can be saved in the repository as well.

All resources, including folders, have unique IDs, names, and optional descriptions. The ID of a resource, along with the ID of its enclosing folders create the path used to reference that resource. The path of a resource is also called its repository URI (Universal Resource Indicator). The name and description appear in the user interface when you browse or search the repository.

Resources are stored in an internal format not accessible to users or administrators. Any resource can be exported with the js-export utility, but the resulting files are for backup or transfer to another JasperReports Server instance and cannot be modified.

JasperReports Server restricts access to folders and resources based on organizations, permissions, user names, and roles. For more information, see Repository Administration.

The sample data includes dashboards, reports, Domains, data sources, and many of their components, such as input types and image files. Each type of content is stored in a separate folder, making it easy to locate. The Supermart Demo folder contains a complete example of dashboards, reports, and resources for various business scenarios in a fictional grocery store company.

Browsing and Searching

Users and administrators can browse or search the repository, depending on what action they want to perform and how resources are organized. Searching the repository finds specific resources faster. For more information on browsing and searching the repository, see the JasperReports Server User Guide.

- Browsing - On the Home page, click View > Repository.

    The Folders panel on the left lists the folders in the repository and the Repository panel lists the contents of the selected folder. The tool bar in the Repository panel allows administrators to perform actions such as Copy, Cut, Paste, and Delete; select several resources with Control-click or Shift-click to perform actions in bulk.

Figure 3: Browsing the Repository

- Searching - Enter a search term in the search field at the top of any page, or select View > Search Results.

Figure 4: Search Field in the Menu Bar

The search results page displays a search field with the current search term at the top of the Repository panel. The search applies to resource IDs, names, and descriptions. Use the filters in the left-hand panel to refine your search.

Figure 5: Searching the Repository

Overview of Users and Roles

User accounts and role membership provide authentication and authorization for access control in JasperReports Server. When logging in, users enter their username and password to access the server. In a server with multiple organizations, users must also enter their organization ID. Administrators assign named roles to users and then create role-based permissions to further control access to the repository.

Users and roles are associated with the organizations in which they are defined. Users and roles defined in an organization can be granted or denied access to any repository folder or object in the organization or its suborganizations. However, a suborganization administrator has no access to the roles and users in the parent organization, even if they are used in access permission within the suborganization.

User names and role names are unique within an organization, but not necessarily among suborganizations or across all organizations in the server. For example, the default organization administrator is called jasperadmin in every organization. Because login credentials include the organization and user name, JasperReports Server can distinguish each user. In some cases such as web services, a user is identified by the unique string username|organization_ID.

Administrators define permissions directly on the resources and folders in the repository. You can define a level of access, such as read-write, read-only, or no access, and assign each permission based on either a username or a role.

Administering Users and Roles

Administrators perform the following actions to manage users in their organization:

- Create, modify, and delete users.

- Set user account properties such as name, email, and attributes.

- Reset a user password. However, no administrator can ever view a user's existing password.

- Login as any user to test permissions.

- Create, modify, and delete roles.

- Assign roles to users.

- Set access permissions on repository folders and resources.

The system admin (superuser) can perform these actions for any user or any role in any organization in the server. An organization admin (jasperadmin) can perform these actions for users and roles in the same organization or its suborganizations.

Delegated Administration

JasperReports Server enables three levels of delegated administration:

- With multiple organizations, administrators in each organization are limited to actions within their organization.

- The Administer permission allows a user to view and set permissions on a folder or resource. This can allow a power-user to manage a section of the repository, but not to create or manage users.

- Granting ROLE_ADMINISTRATOR, ROLE_SUPERUSER, or both allows a user to see the management interface and create users and roles. This is true delegated administration, whereby a user other than jasperadmin or superuser has administration abilities.

In the case of true delegated administration, three factors determine the scope of a user's administrative privileges:

- ROLE_ADMINISTRATOR – JasperReports Server confers the organization-level privileges to any user with this role. This includes managing users, roles, and permissions, as well as creating resources in the repository. When a user with this role logs in, the server displays the additional menus to access the admin pages and manage repository resources. Any administrator can assign this role to any other user.

- ROLE_SUPERUSER – When a user already has ROLE_ADMINISTRATOR, this additional role grants access to the system configuration functions. Only a system admin can assign this role to another user.

  In a multi-organization environment, ROLE_SUPERUSER should not be given to organization admins or organization users, because this allows access to the Ad Hoc cache shared by all organizations. In the case of a single organization such as in the default installation, you may assign this role to the organization admins to grant access to system settings without granting privileges to create top-level organizations or other system administrators.

- The user's organization – Regardless of roles, an administrator is always limited in scope to the organization in which the user account is created, including any suborganizations thereof. In no case can a user, even with the ROLE_SUPERUSER, ever view or modify any organization, user, role, or folder outside of the organization to which that user belongs.

Any administrator can grant ROLE_ADMINISTRATOR to any user. That user then becomes equivalent to an organization admin of the organization in which he belongs. In order to delegate system administration, the existing system admin must first create other users at the root level, outside of any organization. The system admin can then assign both ROLE_ADMINISTRATOR and ROLE_SUPERUSER to grant them system admin privileges. For further information about these roles, see Repository Permissions.

Overview of Security

JasperReports Server ensures that people can access only the data they are allowed to see. The mechanisms that define organizations, users, roles, and repository resources work together to provide complete access control. Security has many facets covered in this guide and other guides:

Authentication	Authentication is the process of restricting access to identified users. Users must log in with their user ID and password so that they have an identity in JasperReports Server. The server stores user definitions, including encrypted passwords, in a private database. Administrators create, modify, and delete user accounts through the administrator pages, as described in Managing Users.
Password policies	Every company must establish a password policy that weighs its security risks against the convenience of its users. JasperReports Server supports many different password policies such as password expiration, reuse, and strong patterns. This configuration is described in the JasperReports Server Security Guide.
External authentication	External authentication uses centralized identity services such as LDAP (used by Microsoft Active Directory and Novell eDirectory), Central Authentication Service (CAS), or Java Authentication and Authorization Service (JAAS). For more information, see the JasperReports Server External Authentication Cookbook.
Application Security	System admins who install and maintain enterprise software know they must protect their servers against hackers. JasperReports Server protects your data against intruders with many protocols and tools, such as HTTPS, encryption, CSRF prevention, and input validation against cross-site scripting and SQL injection. For these topics and others, see the JasperReports Server Security Guide.
Organizations	Users belong to organizations and are restricted to seeing resources within their organization. Organizations have their own administrators, but they see only the users, roles, and resources from their organization. When JasperReports Server is configured with multiple organizations, they are effectively isolated from each other, although the system admin can share resources through the Public folder. For more information, see Multiple Organizations in the Repository.
Roles	JasperReports Server also implements roles that can be assigned to any number of users. Roles let administrators create groups or classes of users that are granted similar permissions. A user may belong to any number of roles and receive the privileges from each of them. Administrators create, modify, and delete roles through the

	administrator pages, as described in Managing Roles.
Resource permissions	Administrators can define access permissions on every folder and resource in the repository. Permissions are enforced when accessing any resource either directly through the repository interface, indirectly when called from a report, or programmatically through the web services.
	Permissions can be defined for every role and every user, or they can be left undefined so they are inherited from the parent folder. To restrict access or hide resources such as database connections, the administrator can set no-access or execute-only permission. For more information, see Repository Permissions.
Administrator privileges	JasperReports Server distinguishes between administrators and users. Administrators are granted access to the UI for permissions, user management, and sensitive resources such as database connections. Administrators also set the UI appearance with themes and monitor server activity with diagnostics. Regular users are restricted to the folders, reports, and dashboards that admins allow them to access. Most of the features in this guide are not accessible to regular users. See Delegated Administration.
Menus and pages	The menus that appear in JasperReports Server depend on the user's roles. For example, only users with the administrator role can see the Manage menu and access the administrator pages. By modifying the server's configuration, you can modify access to menus, menu items, and individual pages. Refer to the JasperReports Server Source Build Guide and JasperReports Server Ultimate Guide for more information.
Attributes	Attributes are name-value pairs associated with a user, organization, or server. Attributes can be used to restrict or enable a user's access to data in several ways. See Managing Attributes.

Administrators must keep security in mind at all times when managing organizations, user, roles, and resources, because effective security relies on all of them working together.

Administrator Login

Administrators log in on the standard login page, using the following default passwords:

System admin:	user ID superuser and password superuser
Organization admin:	user ID jasperadmin and password jasperadmin

> For security reasons, always change the default administrator passwords immediately after installing JasperReports Server. For instructions, see Editing a User.

For more information about options on the Login page and logging in with multiple organizations, see the JasperReports Server User Guide.

The first time you log in as an administrator, you may be prompted to opt-into the Heartbeat program. You should also set the administrator passwords and email.

Failed login attempts can disable an account. The number of remaining attempts is displayed after a failed login.

It is recommended to create a second system admin (with custom name) in case a superuser gets locked out, you will need to edit the db table to unlock. See **Creating a System Administrator**.

Refer to the Enabling a Locked User topic for information on how to unlock a user account. Refer to the *JasperReports Server Security Guide* for information on how to disable or configure the setting (enabled by default).

JasperReports Server Heartbeat

When you log into JasperReports Server for the first time after installation, administrators may be prompted to opt into the server's Heartbeat program. It reports specific information to Jaspersoft about your implementation: the operating system, JVM, application server, database (type and version), and JasperReports Server edition and version number. By tracking this information, we can build better products that function optimally in your environment. No personal information is collected.

To opt into the program, click OK. To opt out, clear the check box then click OK.

Administrator Email

After logging in for the first time, you should set the email on the superuser and jasperadmin accounts to your email address. In rare cases, the server may notify you by email about issues with your license.

> ⚠️ This is also a good time to change the default passwords on the superuser and jasperadmin accounts.

To set the email and passwords on the administrator accounts, edit the user account information as described in Editing a User.

Administrator Pages

Administrators have access to special pages to manage the server. After logging in, click Admin Menu on the Home page, or select an item from the Manage menu on any page.

System admin:

Organization admin:

Figure 6: Manage Menu for Administrators

The administrator controls are different for system and organization admins, as shown in the previous figure. Organization administrators can manage users, roles, and suborganizations, but only within their organization. System admins can manage top-level organizations, as well as users and roles in any organization. And only system admins have access to the server settings for configuring logs, Jaspersoft OLAP, and Ad Hoc cache and data policies.

The following figure shows the Admin page for system admins. The page is similar for organization admins, but it does not have the Server Settings heading.
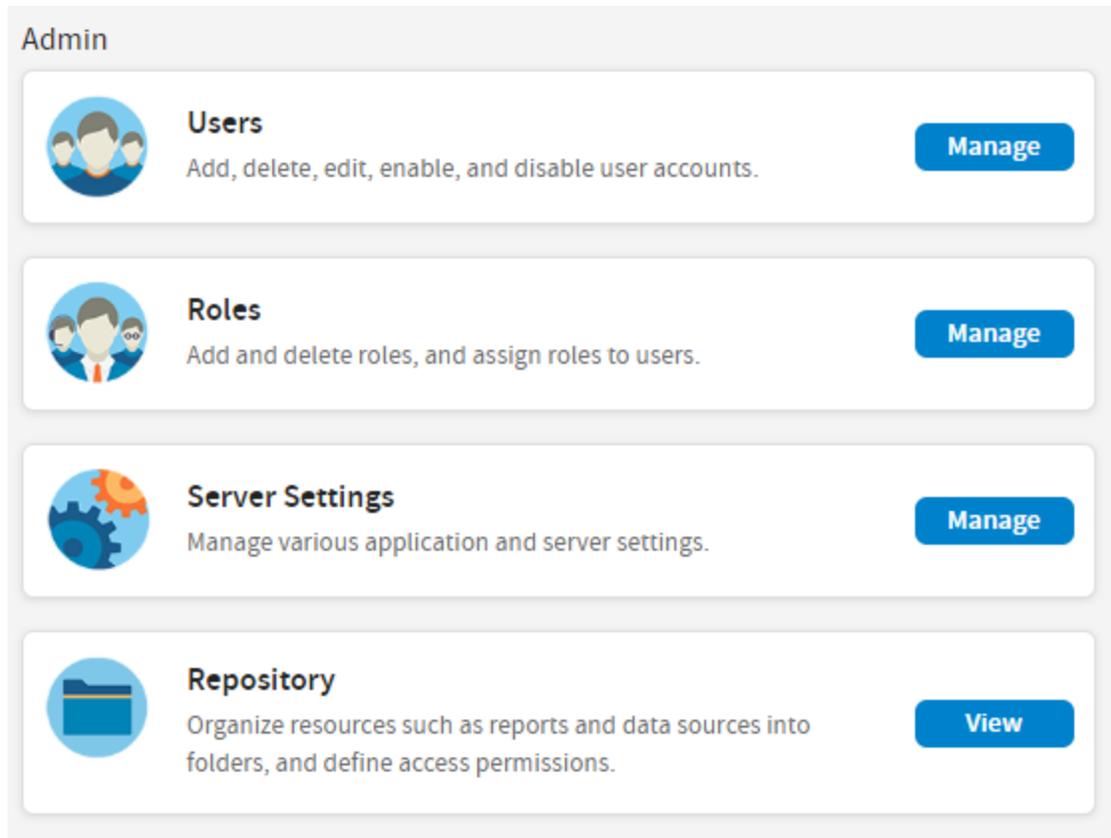
Figure 7: Manage Server Page for System Admins (superuser)

The pages for administering organizations, users, and roles are documented in User and Role Management. The pages for administering Server Settings are documented in Configuration Settings in the User Interface.

The About JasperReports Server link in the footer of all pages displays the product version along with the software build and the details of your license. Please have this information available if you need to contact Jaspersoft for support.
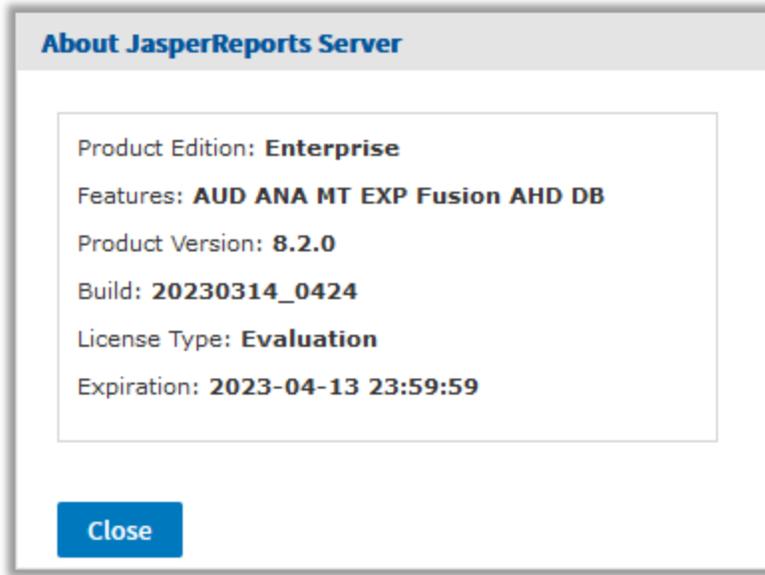
Figure 8: About JasperReports Server Dialog

User and Role Management

In a single-organization deployment, the administrator needs to create only the users and roles. In deployments with multiple organizations, administrators need to create organizations, populate them with users, and create the roles they use afterward to set access permissions. Optionally, administrators can also define attributes to control access to data.

In a deployment with multiple organizations, there can be administrators at every level of the hierarchy, as described in Delegated Administration. Part of any large deployment is to designate the administrators responsible for every task. For example, system administrators might set up the top-level organizations and default roles, but each organization's admin would then create and manage the users of their particular organization.

The interface in JasperReports Server for managing organizations, users, and roles accommodate all levels of administrators. This makes it easy for them to search among hundreds of users and roles, whether in a single organization or spread across many. The interface also enforces the scope of administrative privileges. For example, it ensures that an organization administrator cannot see roles and users from parent organizations.

Attributes are name-value pairs that can be defined on users, organizations, and at the server level. These values can provide flexibility, for example allowing each organization to access a different database when using the same data source. At the user level, attributes can provide data-level security when combined with the features of Domains and OLAP.

This chapter contains the following sections:

- Managing Organizations
- Managing Users
- Managing Roles
- Managing Attributes

Managing Organizations

System admins and organization admins use the same pages for managing organizations, the only difference is that system admins can manage top-level organizations, and organization admins are limited to suborganizations.

> Administrators of deployments with a default single organization can generally skip this section. However, the procedure in Editing an Organization can be used to change the name of the default organization.

The system admin (superuser) can view all the organizations in the server, as shown in the following figure. In the Organizations panel on the left, the system administrator's view begins at the root of the organization hierarchy and includes all defined organizations and suborganizations, so he can manage any organization or suborganization in the server. In this example, there are two top-level organizations, and one of them has several suborganizations.

Figure 9: System Admin View of Manage Organizations Page

The following figure shows the same repository as seen by the admin of Organization (jasperadmin). It shows that this administrator's view is limited to his own organization and its suborganizations, and he can access and manage only those.

Manage Organizations
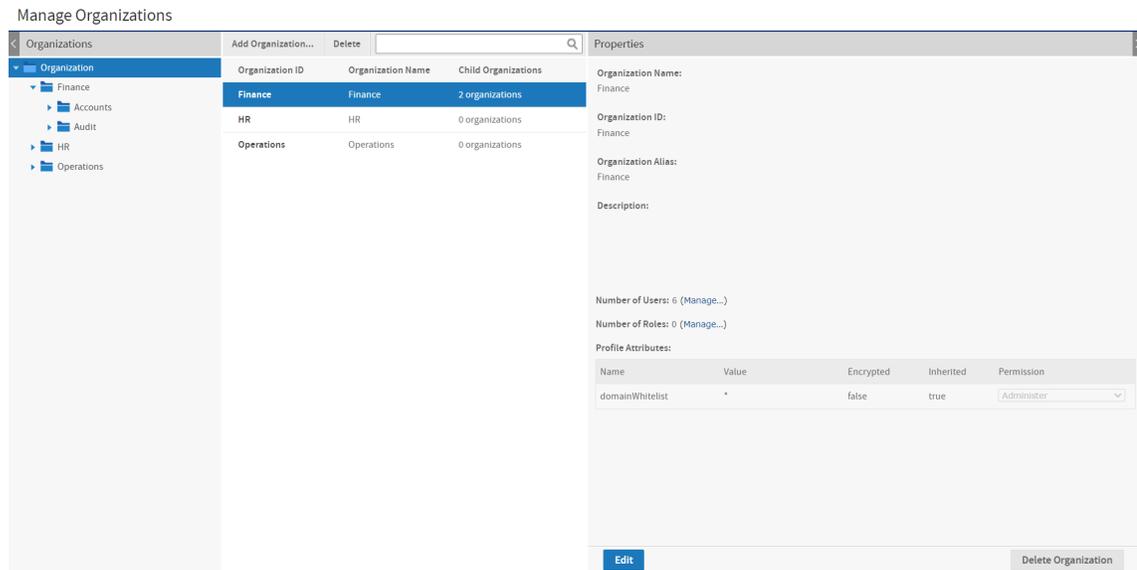
Figure 10: Organization Admin View of Manage Organizations Page

Viewing Organization Properties

Procedure

1. Log in as a user with administrative privileges for the organization you want to view.

2. Select Manage > Organizations.

3. The organization management page appears, as shown in the previous figures.

4. To select an organization, click its parent in the left-hand Organizations panel, then select the organization in the center panel. If there are many organizations, you can search for a specific organization. Note that the search only applies to suborganizations of the parent organization currently selected in the left-hand panel.

5. Once you select an organization, the Properties panel on the right shows information about it:

• Name – Display name of the organization that appears on the organization's top folder.

• ID – Unique and permanent organization ID used for logging in.

• Alias – Unique but editable short organization name that can also be used when

logging in.

- Description – Optional description that appears only in this Properties panel.

- Number of Users – Count of all users, including those in any suborganizations. Click Manage to see the list of users on the user management page.

- Number of Roles – Count all roles, including those in any suborganizations. The number of roles does not include the system roles (such as ROLE_USER) that appear at every organization level but are defined at the root level. Click Manage to see the list of roles on the role management page.

- For information about attributes on the organization, see Managing Organization Attributes.

> Both system admins and organization admins can also export and import entire organizations from the Manage Organizations page. This functionality can be used to duplicate organizations or change their hierarchy, or to create a backup. For more information, see Exporting from Organizations and Importing to Organizations.

Creating an Organization

Procedure

1. Log in as a user with administrative privileges for the parent of the new organization.

2. Click Manage > Organizations.

3. In the left-Organizations panel, expand the hierarchy of organizations to select the parent organization, for example Finance, then click Add Organization in the middle panel.

4. The Add Organization dialog appears.

Add Organization

Organization Name (required):

Investments

Organization ID (required):

Investments

The ID cannot be changed once the organization is created.

Organization Alias (required):

invest

Description:

Investments division of the Finance Department

Add Organization to Finance   Cancel
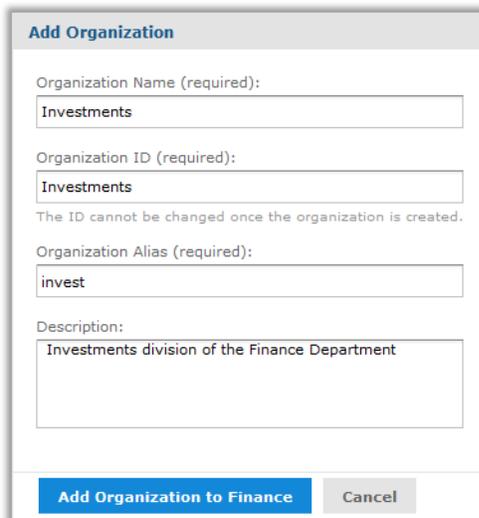
Figure 11: Adding an Organization

5. Enter the organization name; the server automatically fills in the ID and alias based on the name. You can change the ID and alias if needed before saving the organization. Once saved, the organization ID can no longer be modified. The Description is optional. The previous figure shows this dialog with sample values.

6. To save the new organization, click Add Organization to <organization>.

    The new organization appears in the Organizations panels. When you select it in the center panel, its properties appear in the Properties panel on the right.

The properties panel shows the number of users and roles in the organization and provides links to manage them. By default, new organizations have the following:

- Two users with default passwords: the organization admin (jasperadmin/jasperadmin) and a sample user (joeuser/joeuser).

    > For security reasons, always change the default passwords immediately after creating a new organization. For instructions, see Managing Users.

- The new organization has no roles of its own. The default users have the system-wide roles ROLE_USER and ROLE_ADMINISTRATOR.

- In the repository, a new folder is created in the parent's Organization folder. This new organization folder contains a copy of the parent's Organization/Folder Template folder. To manage the Organization folders, select View > Repository.

Default Folders for Organizations

Every organization contains a special folder named Organizations where the server creates suborganizations and administrators can manage its resources. The Organizations folder always contains a folder named Folder Template. When you create a new organization, the entire content of the Folder Template is copied to create your new organization's folders. You can add folders and resources in the Folder Template, and these are copied into each organization created thereafter.

The default folders in the Folder Template are:

- Ad Hoc Components\Topics – The location where the Ad Hoc Editor looks for Topics to create new reports.

- Temp – A folder visible only administrators, used by the server to store temporary files.

- Templates – A folder that holds templates used when generating reports from Ad Hoc views.

- Themes – A special folder managed by the system to contain CSS files that define the appearance of the user interface.

> The Public folder visible in every organization is a special linked folder at the root level. The repository makes it accessible to every organization, but it is not within the organization folder.

There is a Folder Template at every level of the organization hierarchy, including the root. The system admin can add content to the top-level Folder Template for use in creating top-level organizations. Organization admins can add content to their Folder Templates for use in creating suborganizations.

Finally, the Folder Template itself is copied into a new organization, so each new suborganization has the same default folders and resources as its parent.

Editing an Organization

Procedure
1. Log in as a user with administrative privileges for the organization.

2. Click Manage > Organizations.

3. In the left-Organizations panels, select the organization's parent. In the center Organizations panel, select the organization.

4. In the Properties panel, click Edit so you can change the organization's properties.
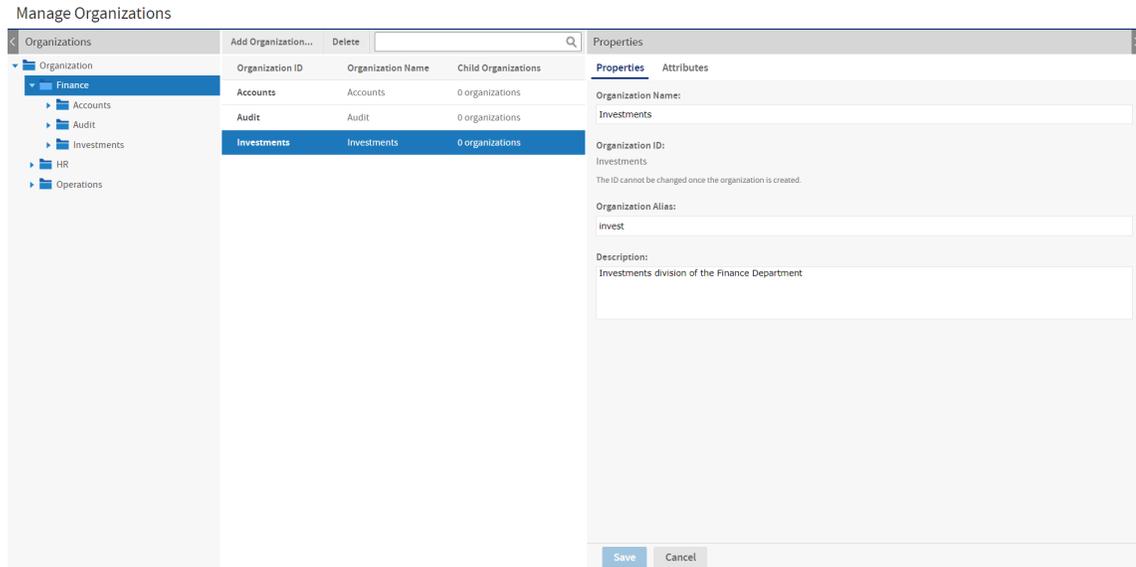
Figure 12: Editing Properties of an Organization

5. Edit the organization properties as needed. Changing the organization name changes the name of the organization's folder, as well, but no other data. You can change the alias and description. The organization ID is defined when the organization was created and cannot be modified.

6. For information about attributes on the organization, see Managing Organization Attributes.

7. Click Save to keep your changes, or Cancel to quit without saving.

Deleting an Organization

Procedure

1. Log in as a user with administrative privileges for the organization.

2. Click Manage > Organizations.

3. In the left-Organizations panels, select the organization's parent. In the center

---

Organizations panel, select the organization and click Delete.

4. Administrators cannot delete the organization to which they belong. Confirming the delete completely removes all users, roles, and folders of the organization and all of its suborganizations from JasperReports Server.

Managing Users

As with organizations, system admins (superuser) can manage all users in all organizations and create users outside of organizations, as described in Delegated Administration. Organization admins can manage the users in their own organization and in any of its suborganizations.

In a deployment without organizations, you should still be familiar with the structure because of the system admins (superuser or other delegated system admins) that exist outside of the single default organization. All other users belong to the single default organization.

The default installation of JasperReports Server includes the following users:

Default Users after Installation

Username	Default Password (Case sensitive)	Description
superuser	superuser	Default system admin who does not belong to any organization (defined at the root level).
jasperadmin	jasperadmin	Default organization admin in every organization.
joeuser	joeuser	Default end user in every organization.
demo	demo	Included in the single default organization only if you install the sample data.
CaliforniaUser	CaliforniaUser	Included in the single default organization only if you install the sample data.
anonymousUser	anonymoususer	An internal system user. It is not a regular user

Username	Default Password (Case sensitive)	Description
		and is not intended for normal usage.
		Note: Although you can disable an anonymous user's access, this user cannot be deleted.

Due to security concerns, we recommend that you remove any users that are not being used in your instance, whether created by default or otherwise.

Advise your users to change their passwords regularly. To configure periodic expiration of passwords, refer to the JasperReports Server Security Guide.

Viewing User Properties

Procedure

1. Log in as an administrator (jasperadmin in the user's organization or any parent organization, or superuser).

2. Select Manage > Users or on the Admin Home page, click Manage in the Users workflow block. The Manage Users page displays the users in each organization and the properties of the selected user.
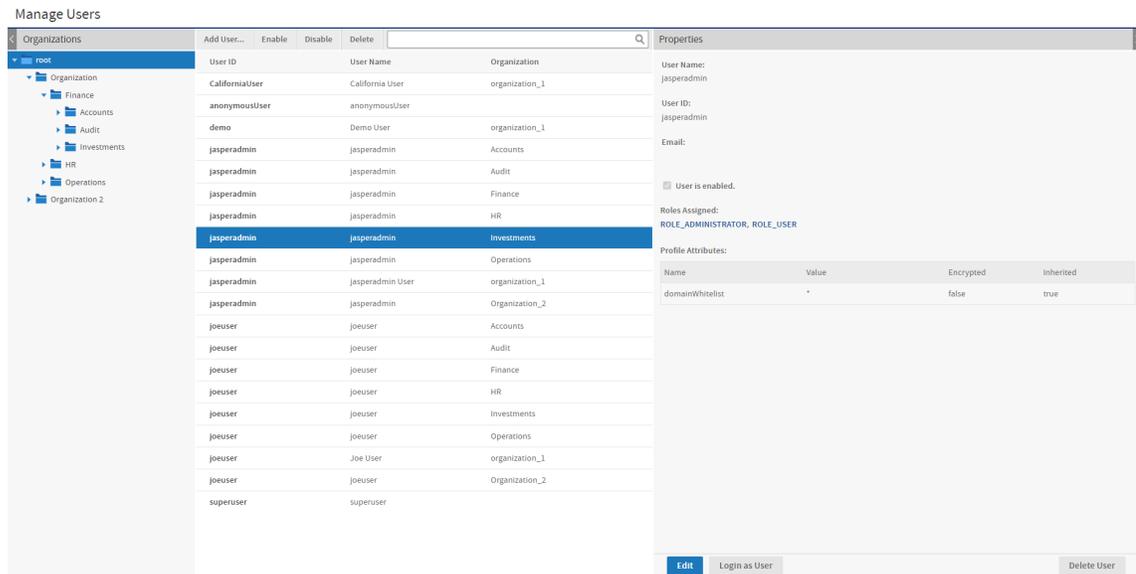
Figure 13: Manage Users Page

The columns in the Users panel list the user ID, the username, and the organization of each user. The list of users includes everyone in the chosen organization and its suborganizations. The same user ID may appear more than once, indicating that users with the same ID were created in different organizations.

In this example, the system admin can see all users in all organizations by selecting the root of the Organization hierarchy. There are always multiple jasperadmin users in a hierarchy of organizations, because it is the default administrator ID in each organization.

3. To locate a user:

   • Browse for users - Expand the organization hierarchy in the left-panel, and select an organization or suborganization. Scroll through the list of users if it is too long to fit on your screen.

   • Search for a user - Select the organization (or any parent organization) and enter a search string in the Search field of the Users panel. The search results show all user ID or names in the selected organization and suborganizations that match the search string.

4. Select a user account to view its Properties in the right-hand panel.

   User status can be Enabled or Disabled. Disabled users are displayed in gray text in the Users list. For convenience, the role names link to the role management page for

each role. For information about attributes on the user, see Managing User Attributes.

> As the admin of a given organization, you can see the roles defined in your organization and its suborganizations but not the parent organization (except for certain system-wide roles). A user may have roles defined and assigned from a parent organization that are not visible to the administrator of the user's organization. For more information, see Managing Roles.

Creating a User

Procedure

1. Log in as an administrator (jasperadmin in the user's intended organization or any parent organization, or superuser).

2. Select Manage > Users or, on the Admin Home page, click Manage in the Users workflow block.

3. In the Organizations panels, select the organization for the new user and click Add User. The Add User dialog appears.

Figure 14: Adding a User

4. Enter the following information:

    - User name - The new user's full name. The name is optional but recommended. It appears in the menu bar of the UI when the user is logged in.

    - User ID - Generated automatically from the username. You can accept the suggested value or type your own. The user ID is used to log into JasperReports Server, and for administrators to manage users and resources. The User ID must be unique within the organization, but may exist in multiple organizations.

    - Email - This is optional but must be in a valid email format.

    - Password and confirmation - Enter and confirm a password for the user.

    - User is enabled - To enable the user to log in, select this check box. Users who are not enabled can't log in. If you implement role-based permissions, you might want to delay enabling the user until you assign more roles. For more information on roles, see Managing Roles.

5. Click Add User to <organization>.

    The new user is available in the Users panel. To assign roles to the user, click Edit in the user's Properties panel.

Editing a User

One way to assign roles to a user is to edit the user's properties. Alternatively, when you edit a role, you can assign it to any number of users.

To edit a user's properties:

Procedure

1. Log in as an administrator (jasperadmin in the user's organization or any parent organization, or superuser).

2. Click Manage > Users or, on the Admin Home page, click Manage in the Users workflow block.

3. In the Organizations panel, select the user's organization or a parent organization.

4. In the Users panel, select the user.

5. In the user's Properties panel, click Edit.
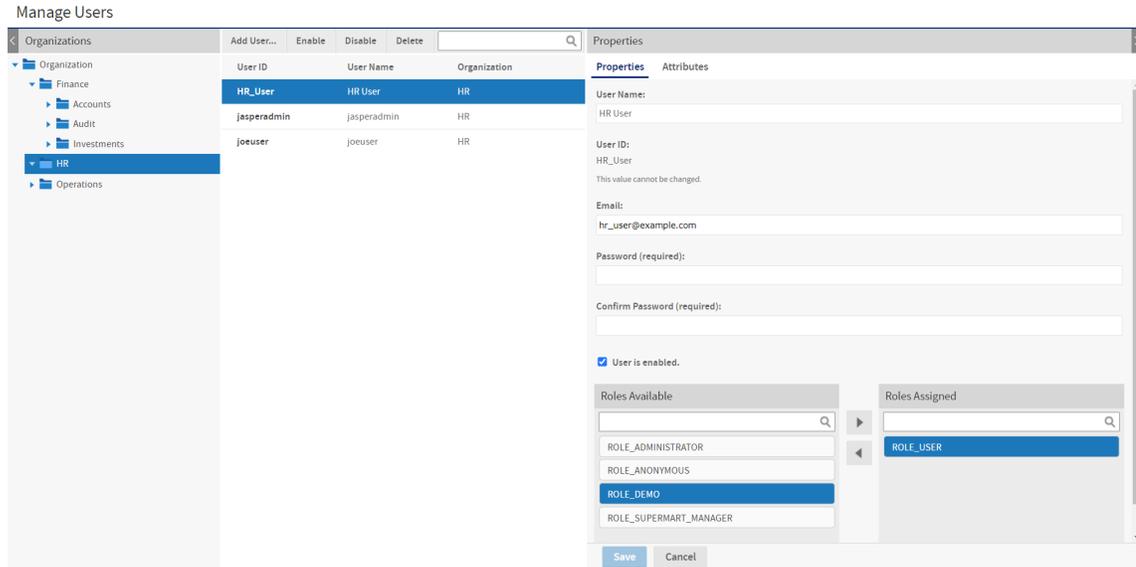
Figure 15: Editing the Properties of a User

6. Edit the user's properties as needed. You can't edit the user ID. It always has the value defined when the user was created originally.

7. To assign or remove roles from the user, select the roles, and use the arrow buttons between the Roles Available and Roles Assigned lists.

The Roles Available list includes any role in the organizations of the current administrator, as well as the special system-wide roles. For more information on creating and adding roles, see Managing Roles.

8. For information about attributes on the user, see Managing User Attributes.

9. Click Save to keep your changes.

10. In the Properties panel, click Login as User to test the user's permissions, as explained in Testing User Permissions.

> Logging in as another user is also necessary when you are maintaining resources that use absolute references in the repository. For more information, see Referencing Resources in the Repository.

Enabling or Disabling Multiple Users

You may sometimes need to disable user accounts. For example, when making configuration changes, you may want to lock out all users until the changes are finished.

Administrators can select any number of users in their organization, and the system admins (superuser) can select all users in the server, except themselves.

Procedure

1. Log in as an administrator (jasperadmin in the user's organization or any parent organization, or superuser).

2. Click Manage > Users or, on the Admin Home page, click Manage in the Users workflow block.

3. In the Organizations panel, select the users' organization; to enable or disable users in different organizations, select the common parent organization.

4. In the Users list, use Control-click and Shift-click to make multiple selections. If the User list is too long, enter a search term to find users and enable or disable them individually.

5. Click Enable or Disable in the menu bar.

Deleting One or More Users

Procedure

1. Log in as an administrator (jasperadmin in the user's organization or any parent organization, or superuser).

2. Click Manage > Users or, on the Admin Home page, click Manage in the Users workflow block.

3. In the Organizations panel, select the user's organization; to delete users in different organizations, select the common parent organization.

4. In the Users list, use Control-click and Shift-click to make multiple selections. If the list of users is too long, enter a search term to find and select the user.

5. In the tool bar of the Users panel, click Delete and confirm the action.

Preventing Deletion of a User

You can define a user from being deleted. This could be useful, for example, if you want to prevent having a superuser or jasperuser role from getting deleted.

To define a user from deletion:

Procedure

1. Edit *<tomcat-home>/webapp/jasperserver-pro/WEB-INF/applicationContext-security.xml*

2. Search for *util:list bean "restrictedUsersListToBeDeleted"*

    <util:list id="restrictedUsersListToBeDeleted" list-class="java.util.ArrayList" value-type="java.lang.String">

    <value>anonymousUser</value>

    </util:list>

By default, this location has "*anonymousUser*", but you can add additional users, for example "*superuser*"; or for tenant user, "*jasperadmin|organization_1*".

Enabling a Locked User

A user's account may become disabled due to repeated failed login attempts.

An Administrator can re-enable locked-out users from their organizations by selecting the **Enable** checkbox from the **Edit user** panel.

If a superuser account gets locked out, then such an account can be re-enabled by another superuser, or by a DB Administrator (DBA) by performing the following command to update the JIUser table, which correct the following issue:

```
update jiuser set enabled='t' where username='superuser';
```

After updating the table, restart the Tomcat server to clear the User Details Cache.

User Authentication Best Practices

The following security considerations should be taken before re-enabling a user's account:

*   All failed login attempts for External Users (except CAS, as it is not supported) before they successfully login in JasperReports Server will be tracked in JIExternalUserLoginEvents. That table has auto-clean cron-job and will reset their failed attempts by an internally scheduled process (for more details, refer to the

JasperReports Server Security Guide).

- Verifying the identity of the locked out user.

- Checking logs for human error or automated attack.

> For more information on how to check logs, refer to the *JasperReports Server Security Guide*.

- You can enable logging of user login events and failed attempts by setting:

```
com.jaspersoft.jasperserver.api.security.externalAuth.wrappers.spring.ldap.JSLdapAuthenticationPr
ovider
com.jaspersoft.jasperserver.api.security.internalAuth.InternalDaoAuthenticationProvider
```

```
to DEBUG in Manage > Server Settings > Log Settings.
```

Creating a System Administrator

For security reasons, it is recommended to have a second system administrator with a custom username in case a jasperadmin gets locked out.

Managing Roles

Roles define sets of users who are granted similar permissions. Administrators create roles, assign them to users, and set permissions in the repository (see Repository Permissions). By default, JasperReports Server includes the following roles; some are needed for system operation, some are included as part of the sample data:

Default Roles in JasperReports Server Installations

Role	Description
ROLE_SUPERUSER	This role determines system admin privileges, as explained in Delegated Administration. It's a system-level role, but in a single-organization deployment, the system admin can assign it to an organization admin.

Role	Description
	Never delete this role, it's required for proper administration of the server.
ROLE_ADMINISTRATOR	This role determines organization admin privileges. This role is automatically assigned to the default jasperadmin user in every new organization. It's a special system-level role visible in every organization, and organization admins can assign it to other users, as explained in Delegated Administration. Never delete this role, it's required for proper administration of the server.
ROLE_USER	Required to log in. This role is automatically assigned to every user in the server. It's a special system-level role visible in every organization. Never delete this role, it's required to create users and allow them to log in.
ROLE_ANONYMOUS	When anonymous access is enabled, this role is automatically assigned to any agent accessing the server without logging in. It's a special system-level role visible in every organization. This role is also assigned to the default anonymous user. By default, anonymous access is disabled and this role isn't used. It's a system role that even the system admin can't delete.
ROLE_PORTLET	This role is only found in JasperReports Server instances that have received authentication requests from a portal such as Liferay or JBoss. If you do not use a portal server, you can delete this role.
ROLE_DEMO	This role grants access to the SuperMart demo Home page, reports, and if you implement Jaspersoft OLAP, OLAP views. This role is assigned to the demo user in the default organization. These objects are available only if you installed the sample data when you installed your server. It is a special system-level role that is visible in every organization When you no longer need the sample data, this role can be deleted.

Role	Description
ROLE_SUPERMART_ MANAGER	This role is used to assign permissions relative to the sample data. It is a special system-level role that is visible in every organization. It demonstrates data security features available in Jaspersoft OLAP. See the Jaspersoft OLAP Ultimate Guide for more information. When you no longer need the sample data, this role can be deleted.

When you need to define permissions for sets of users, administrators can create new roles and assign them to users. Users can belong to any number of roles and each can be used for access to different resources.

Except for the five special system-level roles visible in every organization, roles are defined within organizations. The same role ID can be defined in multiple organizations, as long as it is unique within each organization. Admins can manage all roles in their organizations and any suborganization, but they can never see roles in a parent or sibling organization. JasperReports Server enforces this scheme to ensure that organizations are secure and only valid roles are assigned to users.

It is possible for an administrator to assign a role to a user in a suborganization, where the role is defined in a parent organization of the user. The admin of the user's organization cannot see the role when managing the user, but the admin of the role's organization can, and permissions associated with the role are properly enforced.

Viewing Role Properties

Procedure

1. Log in as an administrator (jasperadmin in the role's organization or any parent organization, or superuser).

2. Select Manage > Roles or, on the Admin Home page, click Manage under Roles. The Manage Roles page displays the roles defined in the server and in each organization and properties for each role.
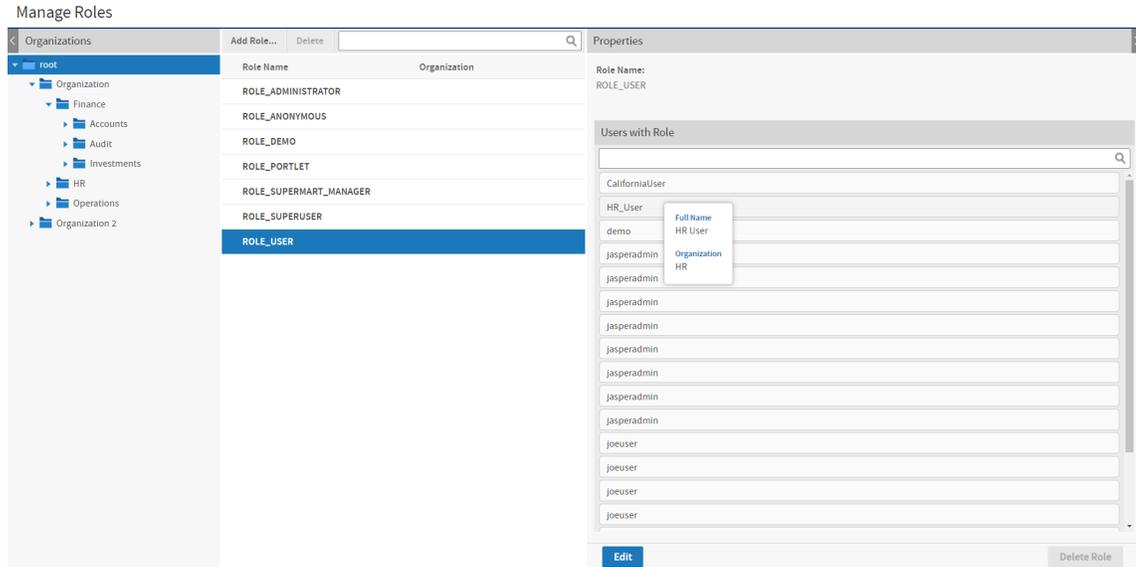
Figure 16: Manage Roles Page

> The Roles list includes all roles in the chosen organization and its suborganizations along with the five default system-level roles. The same role name may appear more than once if roles with the same name were created in different organizations. The second column (blank in this figure) gives the organization name of a particular role.
>
> In this example, the system admin can see all roles in all organizations by selecting the root of the Organization hierarchy.

3. To select a role, click its organization in the Organizations panel (Commercial edition users only). The Roles panel displays all the roles.

4. To filter the list of roles, enter a search string in the search field of the Roles panel. The search results show all of the roles in the selected organization and suborganizations whose names contain the search string. If necessary, scroll through the new list or refine your search.

5. Select the role in the Roles panel. The role's properties appear in the Properties panel.

> The Properties panel shows the role name, the organization where it's defined, and the users assigned to the role. You can enter a search term to find users in the list. Some user IDs may appear several times because the same ID can exist in different organizations. Hover over a

```
user ID to see a user's full name and organization, as shown in the
figure.
```

📝 When you view the properties of a special system-level role, you only see the users with that role in your organization or any suborganization. An organization admin can never see users outside of his organization or its suborganizations.

Creating a Role

Procedure

1. Log in as an administrator (jasperadmin in the role's intended organization or any parent organization, or superuser).

2. Select Manage > Roles or, on the Admin Home page, click Manage under Roles.

3. In the Organizations panel, select the organization to which the role will belong.

4. Click Add Role. The Add Role dialog appears.

Figure 17: Adding a Role

5. Enter the name of the role. The role name is also the role ID and does not accept spaces or special characters.

6. Click Add Role to <organization> to create the role.

   The new role is included in the Roles panel. If you want to assign users to the role, click Edit in the Properties panel of the new role.

Assigning Users to a Role

You can assign multiple users to one role. To assign multiple roles to one user, edit the user's properties as described in Editing a User.

Procedure

1. Log in as an administrator (jasperadmin in the role's organization or any parent organization, or superuser).

2. Select Manage > Roles or, on the Admin Home page, click Manage under Roles.

3. In the Organizations panel, select the role's organization.

4. Select the role in the Roles panel.

> Unless you're logged in as the system admin, you can't edit or delete the five special system-level roles.

5. In the Properties panel, click Edit. The role's properties become editable.
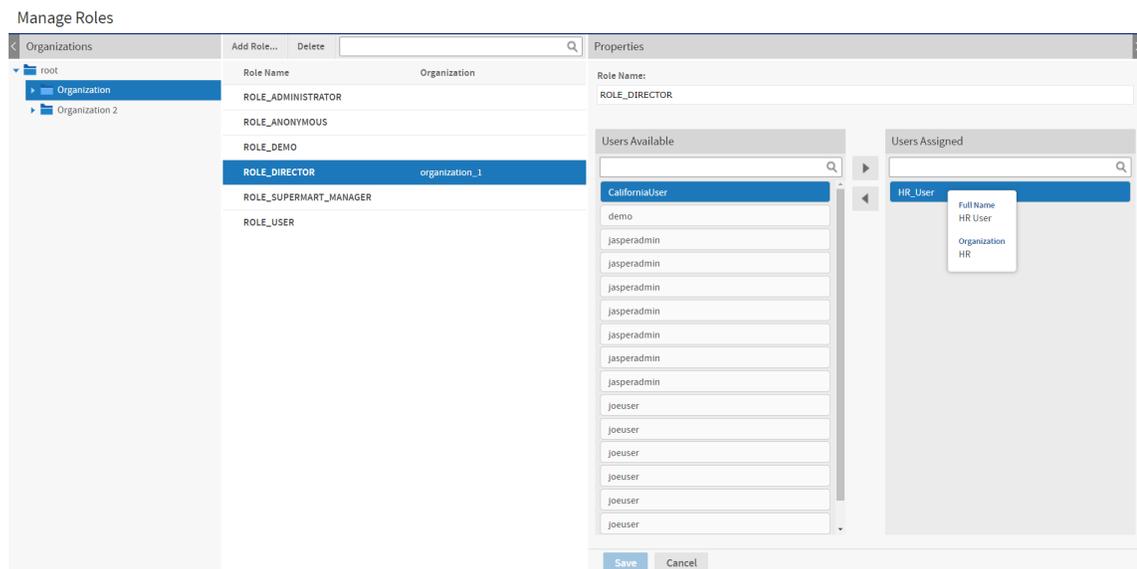
Figure 18: Editing the Members of a Role

6. Enter a different name to change the role name throughout the server.

> Permissions in the repository that use the role name are automatically updated. However, role names in security files for Domains and OLAP are not updated with the new role name and may cause a security risk. If you use security files for Domains or OLAP, do not change role names without verifying the files as well. For more information, see the JasperReports Server User Guide.

7. To assign or remove role users, select the users, and click the arrow buttons between the Users Available and Users Assigned lists. You can enter a search term to find users in the lists. Some user IDs may appear several times because the same ID can exist in different organizations. Hover over a user ID to see a user's full name and organization, as shown in the figure.

8. Click Save to keep your changes, or Cancel to quit without saving.

Deleting One or More Roles

Procedure

1. Log in as an administrator (jasperadmin in the role's organization or any parent organization, or superuser).

2. Select Manage > Roles or, on the Admin Home page, click Manage under Roles.

3. In the Organizations panels, select the role's organization. (Commercial users only. Community users skip to step 4.) The Roles panel is displayed.

4. Select the role in the Roles panel. Use Control-click and Shift-click to make multiple selections.

> Unless you're logged in as the system admin, you can't edit or delete the five special system-level roles.

5. In the tool bar of the Roles panel, click Delete and confirm the action.

Managing Attributes

Attributes are name-value pairs that are associated with users, organizations, and the server itself. Attribute values can be used to parameterize access to data sources, Domain data, OLAP data, and report output. Attributes were previously known as profile attributes.

For example, when defining a data source, you can specify that the database name be defined by an organization-level attribute called dbname. Each organization then has an attribute called dbname with a different value. You have a single data source in your repository that is accessible in each organization, but when a user runs a report with that data source, it takes data from the database specified by the value of dbname in that user's organization.

Attributes have permissions, so that system admins or organization admins can restrict the visibility and use of specific attributes at lower levels.

> The permission feature of attributes is designed to work with multiple organizations in commercial editions of JasperReports Server. The feature is enabled in all editions of the server, but has limited use in servers with only a single default organization.

Referencing Attributes

There are several features of JasperReports Server that can read attributes and use attribute values. The following features can limit or restrict access to data or take different behavior based on the user, organization, or server-level attributes. Each feature reads or references attributes in its own syntax that is described with that feature:

- Data sources - The host name, port number, database name, username, and user password can be defined through attributes. See Attributes in Data Source Definitions.

- Reports - Attribute values can be referenced in the calculations as described in Attribute-based Parameters for Queries and Reports.

- Domain security files - Access rules for rows and columns can be defined with attributes so that users or organizations may only see data that they are allowed to access. See the JasperReports Server Data Management Using Domains and a comprehensive example in the chapter "Securing Data in a Domain" in the JasperReports Server Data Management Using Domains.

- Domain pre-filters and calculated fields - Attribute values can be part of filters and calculated fields when creating Domains using the Domain Designer. See the chapter "Working with the Domain Designer" in the JasperReports Server Data Management Using Domains.

- OLAP schemas - Similar to Domains, access rules can be defined with attributes so that users may only see data that they are allowed to access. See the Jaspersoft OLAP User Guide.

Attributes are always referenced in relation to the currently logged in user who is performing an operation or running a report:

- If the value of a user-level attribute is requested, it refers to the attributes of the logged in user.

- If the value of an organization-level attribute is requested, it refers to the attributes

of the logged in user's organization.

- If the value of a server-level attribute is requested, the value depends on which server the user is logged into. For example, test and production servers may have a copy of the same repository, but different server-level attributes. For a given server, server-level attributes are shared and thus have the same value for all users in all organizations.

Attribute Hierarchy

An attribute is a named value that is defined on a user, organization, or root of the server. Any number of attributes with any name can be defined at any or all of these levels. The definition of each attribute at each level is independent of other attributes at other levels. Attributes at different levels may have the same name, unless explicitly forbidden as described in Attribute Permissions.

In the places that you reference attributes (see above), there are two ways to determine the value of an attribute:

- Categorical value - References the value of a named attribute at a specific level, either user level, organization level, or server level. If the attribute does not exist at the requested level, no value is returned.

- Hierarchical value - References the value of a named attribute across all levels, starting at the user level. The server searches for an attribute with the given name in the following order, stopping and returning the first value that it finds:

  ○ At the user level, in the user attributes of the logged in user.

  ○ At the organization level, in the organization the attributes of the logged in user's organization and all parent organizations.

  ○ At the server level.

  ○ If the attributes do not exist at any level, no value is returned.

With hierarchical values, attributes with the same name but different values may be defined at several levels and on many users or organizations. Each attribute is a distinct attribute definition, but the hierarchical value referencing takes into account the level and the presence of the definition for every given user and organization.

To help administrators define attributes on users and organizations, the UI displays all attributes that are visible in the attribute hierarchy. If an attribute is defined in a parent organization or at the server level, its hierarchical value is shown at the lower organization

level and user level, and the attribute is called inherited. This allows the administrator to see all hierarchical values that are in effect in lower levels, either to override them in the hierarchy or leave them as inherited.

Categorical and hierarchical approaches usually involve different strategies for defining and naming attributes. When using categorical values, you must ensure that each and every user or organization has the attribute defined, and you must handle the case when it is accidentally undefined, such as a new user. When using hierarchical values, you define attributes with the same name at different levels, such that each user may have a custom value, but then the organization level provides a default value for all users, and the server level provides a default value for all organizations. Each strategy is useful for different applications, depending on your needs to access and protect data. However, both strategies can co-exist and be used by at the same time by different resources, as long as the names of attributes used in each strategy or each resource are kept distinct.

Attribute Encryption

Because of the attribute hierarchy and the way that attributes are referenced, attribute names and values are one of the few mechanisms that allow lower organizations to see items defined in higher organizations. By default, an attribute and its value can be known to administrators throughout the server.

JasperReports Server provides two mechanisms to protect sensitive attributes:

- Attribute encryption - Prevents the attribute value from ever being seen. Does not prevent the attribute name from being seen or the value from being redefined at a lower level. When an attribute value is encrypted, the value stored internally is encrypted and never decrypted in the user interface. The server decrypts the attribute only when it is referenced by one of the features of the server. The decrypted value is then used for internal operations and is never visible to the user.

  For example, an encrypted attribute could store the password for a database. Once typed in and saved, the UI displays *** as the value of the attribute. When a report uses that database, the server decrypts the password and sends it as part of the database protocol, so the user and the admin never see the password.

  Encrypted attributes are similar to user passwords in the server. Administrators can change the encrypted value to a new value, but they can never view the current value, even the system admin (superuser).

- Attribute permissions - Prevents the attribute from being visible or redefined at a lower level. Attribute permissions are described in the next section.

To properly secure an attribute, you should make it encrypted so it cannot be seen and then set permissions so it cannot be overridden.

Attribute Permissions

> Attribute permissions were designed to work with multiple organizations in JasperReports Server. The feature has limited use in servers with only a single default organization.

Because the attribute hierarchy allows attribute values at lower levels to take precedence over attributes with the same name at higher levels, administrators need a mechanism to enforce the priority of higher-level attributes. The permissions on attributes allow admins to prevent the same attribute name from being seen or redefined in lower levels. Otherwise, a low-level organization admin could define an attribute with the same name as a critical server-level attribute defined by the system admin.

The attribute permissions and their effects are as follows:

- Administer - This is the absence of any restrictions. It allows lower-level administrators to define attributes with the same name and set their permissions locally for their suborganizations. This is the default permission.

    In servers with only a single default organization, use this permission unless you want to disable an attribute with the no-access permission.

- Read Only - The attribute and its value are visible to lower-level administrators, but they cannot define locally an attribute with the same name. In servers with only a single default organization, this permission has no effect.

- Execute Only - The attribute and its value are not visible to lower-level administrators, and they see an error if they try to create an attribute with the same name. However, the attribute returns its value when properly referenced. In servers with only a single default organization, this permission has no effect.

- No Access - The attribute and its value are not visible to lower-level administrators, and they see an error if they try to create an attribute with the same name. In addition, the attribute is disabled and its value cannot be referenced. When referenced, an attribute with no-access permission returns an error.

It is important to remember that attributes are always defined individually at each level. When permissions prevent a lower-level admin from "writing" an attribute, in actuality, the

permission is preventing the lower-level admin from creating a new attribute at his level with the same name (regardless of its value).

As the root user, the system admin (superuser) is not restricted by attribute permissions. Therefore, the system admin can view all attributes defined at all levels and create or modify attributes on any organization or user, regardless of the permission. Thus, the system admin could create an execute-only attribute at the server level and then define an attribute with the same name but different value on an organization or user.

The permission system is based on the permissions above, and when combined with multiple levels of organizations, it can be used to create complex attribute definitions.

Managing Server Attributes

Attributes can be defined at the server level, which is equivalent to the root organization. If a given attribute name is not defined on an organization or a user, it can have a value at the server level to act as a default value. Server-level attributes may also be used as parameters in data source definitions. For example, the same data source may be imported to several servers, such as test and production servers, but a server-level attribute with the same name on each server makes the data source connect to a different database.

To view, create, modify, or delete server-level attributes

Procedure

1. Log in as the system admin (superuser).

2. Select Manage > Server Settings and choose Server Attributes from the left-hand panel.

   Each server level attribute is listed with its value, encrypted status, and permission. Holding the pointer over an attribute name shows the description associated with the attribute. When an attribute value is encrypted, its value is shown as *** symbols.

Figure 19: Managing Attributes at the Server Level

3. To create a new server-level attribute, click Add new attribute. Enter the attribute name and value, as well as an optional description. If desired, set the permission from the dropdown list and select the Encrypt checkbox. Click OK to close the dialog and then click Save to submit the new attribute.

Figure 20: Adding a Server Attribute

4. To modify an attribute, click the edit icon 🖉. In the edit dialog, modify the desired

fields. Click OK to close the dialog and then click Save to modify the attribute.

You can also modify the attribute's permission and encryption by using the dropdown and checkbox in its table row. After confirming any changes, click Save to make them take effect.

When modifying an attribute, be aware of the following:

- Changing the name of an attribute is equivalent to deleting the original attribute and adding a new attribute with the same value. Because this may impact features that reference the attribute, you are asked to confirm the name change.

- Be aware that changing the encryption or permission of an attribute can impact the visibility of an attribute and the features that might rely on referencing its value. Again, you are asked to confirm the change.

- Removing encryption does not decrypt an encrypted attribute. To safeguard encrypted values, removing the encryption on an attribute also erases its value. You should give the attribute a new value by clicking the edit icon.

5. To delete an attribute at the server level, click the delete icon 🗑 in the attribute row and confirm the operation. Because this may impact features that reference attributes, you are asked to confirm the deletion. Click Save to make the deletion take effect.

Managing Organization Attributes

Organization attributes are attributes defined on an organization, regardless of whether it's a top-level organization or suborganization. For example, an attribute with the same name could have different values in each organization, so that the same data source definition would access a different database specific to each organization. Furthermore, suborganizations could also define the same attribute if they need access to a different database than their parent organization.

Organization attributes are managed by either the system admin or the organization admin. As with all management operations, organization admins can set the attributes in their organization and any of their sub-organizations, and the system admin can set the attributes in any organization. If a higher admin wants to restrict access to certain attributes by a lower-level admin, the higher admin must set permissions, as described in Attribute Permissions.

To view, create, modify, or delete organization-level attributes

Procedure

1. Log in as an administrator (jasperadmin in the target organization or any parent organization, or superuser).

2. Select Manage > Organizations.

3. Select the parent organization in the left-hand column, then select the target organization in the center column.

   The right-hand column displays the properties of the target organization, including a table of all attributes visible at this organization's level. The attributes include locally defined attributes, as well as inherited attributes. An inherited attribute is one defined in a parent organization or at the server-level, and whose permissions allow it to be visible in this organization.

Figure 21: Viewing Attributes on the Manage Organizations Page

In the figure above, the system admin (superuser) is logged in and viewing organization attributes. The system admin can view all attributes at all levels, even inherited attributes, regardless of permissions.

Also, the root organization is visible and represents the server level. Selecting root in the Manage Organizations page displays the same attributes and offers the same functionality as the Server Attributes page under Manage > Server Settings. By definition, none of the attributes at the server level are inherited.

4. To create, modify, or delete the attributes on an organization, click Edit in the right-hand column, then select the Attributes tab. In the following figure, the Finance organization admin is logged in and does not see the userName and password attributes because they are hidden by permissions. Also, inherited attributes with the read-only permission are shown but cannot be modified.

Figure 22: Editing Organization Attributes

5. You can filter the attributes in the list to include only the inherited attributes or only the locally defined (not inherited) ones.

6. To create a new organization attribute, click Add new attribute. Enter the attribute name and value, as well as an optional description. If desired, set the permission from the dropdown list and select the Encrypt checkbox. Click OK to close the dialog and then click Save to submit the new attribute.
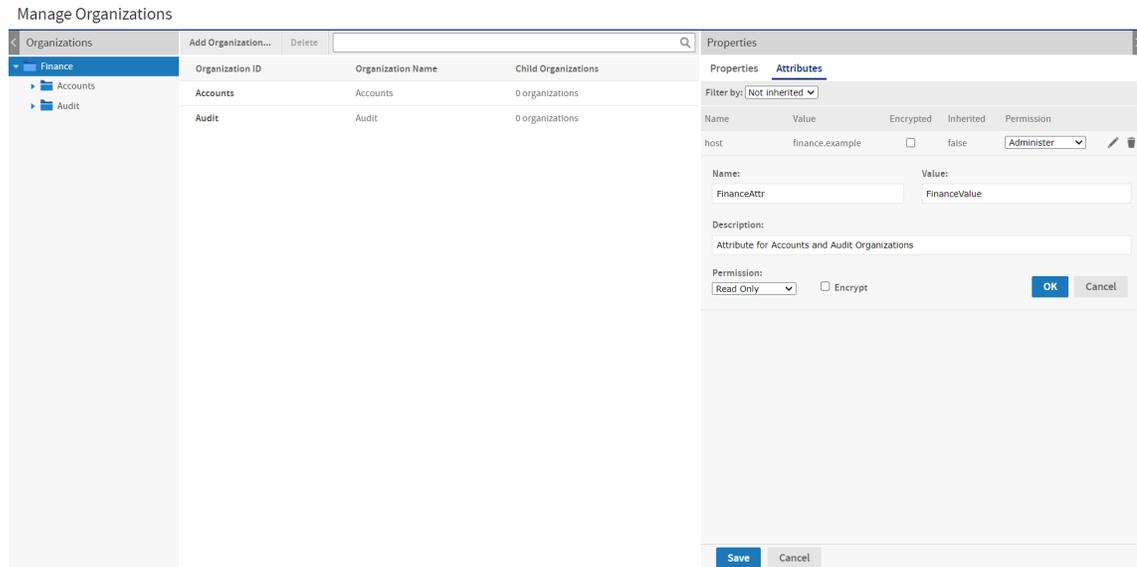
Figure 23: Adding an Organization Attribute

7. To modify an attribute, click the edit icon ✏. In the edit dialog, modify the desired fields. Click OK to close the dialog and then click Save to modify the attribute.
   You can also modify the attribute's permission and encryption by using the dropdown and checkbox in its table row. After confirming any changes, click Save to make them take effect.
   When modifying an organization attribute, be aware of the following:

   • Inherited attributes belong to a parent organization or server root and are shown at the organization level to display the hierarchical attribute values. Any modification to an inherited attribute actually creates a local attribute definition with the modified parameters. In the hierarchy for the selected organization, this new attribute takes precedence over the previously inherited attribute.

   • Changing the name of a locally defined attribute is equivalent to deleting the original attribute and adding a new attribute with the same value. Because this may impact features that reference the attribute, you are asked to confirm the name change.

   • Be aware that changing the encryption or permission of an attribute can impact the visibility of an attribute and the features that might rely on referencing its value. Again, this may impact features that reference the attribute, and you are asked to confirm the change.

- Removing encryption does not decrypt an encrypted attribute. To safeguard encrypted values, removing the encryption on an attribute also erases its value. Click the edit icon to give the attribute a new value.

8. To delete an organization attribute, click the delete icon 🗑 in the attribute row and confirm the operation. Because this may impact features that reference attributes, you are asked to confirm the deletion. Click Save to make the deletion take effect. When deleting an organization attribute, be aware of the following:

   - On deletion, some attributes remain in the list as inherited attributes. This indicates that the local definition of the attribute was deleted, but another attribute with the same name is visible higher in the attribute hierarchy.

   - You cannot delete an inherited attribute, because it belongs to a higher organization or server root. To remove an inherited attribute, you must find where it is defined then delete it at that level. The system admin can also change the permission of the attribute where it is defined to Execute Only or No Access so it does not appear to organization admins in the lower organization anymore.

Managing User Attributes

Attributes on users can provide additional information about a user and can restrict access to data through Domain security files and OLAP schemas. Every user in every organization may have attributes defined, and will also have attribute values inherited through the attribute hierarchy.

Attributes on users do not have permissions because there is no lower level of attributes. Instead, the permissions on organization attributes determine which attributes values apply to a user through the attribute hierarchy.

Users never see the attributes defined on their user profile; only administrators can view and set user attributes.

To view, create, modify, or delete user-level attributes

Procedure

1. Log in as an administrator (jasperadmin in the user's organization or any parent

organization, or superuser).

2. Click Manage > Users or, on the Admin Home page, click Manage in the Users workflow block.

3. In the Organizations panel, select the user's organization. Or select a parent organization and search for the user by name.

4. Select the user in the Users panel. The properties panel includes a table of all attributes for the user, both locally defined and inherited from the attribute hierarchy In the following figure, the system administrator can see that attr2 attribute was redefined at the organization level before being inherited by the user.

Figure 24: Viewing Attributes on the Manage Users Page

5. To create, modify, or delete the attributes on a user, click Edit in the right-hand column, then select the Attributes tab.
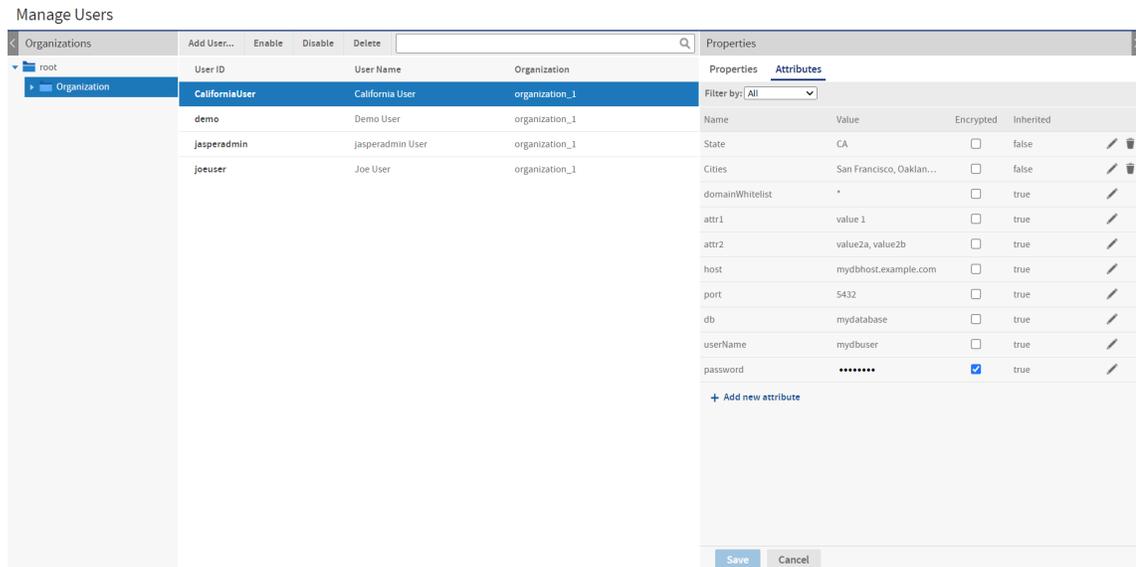
Figure 25: Editing User Attributes

6. You can filter attributes in the list to include only the inherited attributes or only the locally defined (not inherited) ones.

7. To create a new user attribute, click Add new attribute. Enter the attribute name and value, as well as an optional description. If the attribute value should not be visible to other administrators, select the Encrypt checkbox. Click OK to close the dialog and then click Save to submit the new attribute.
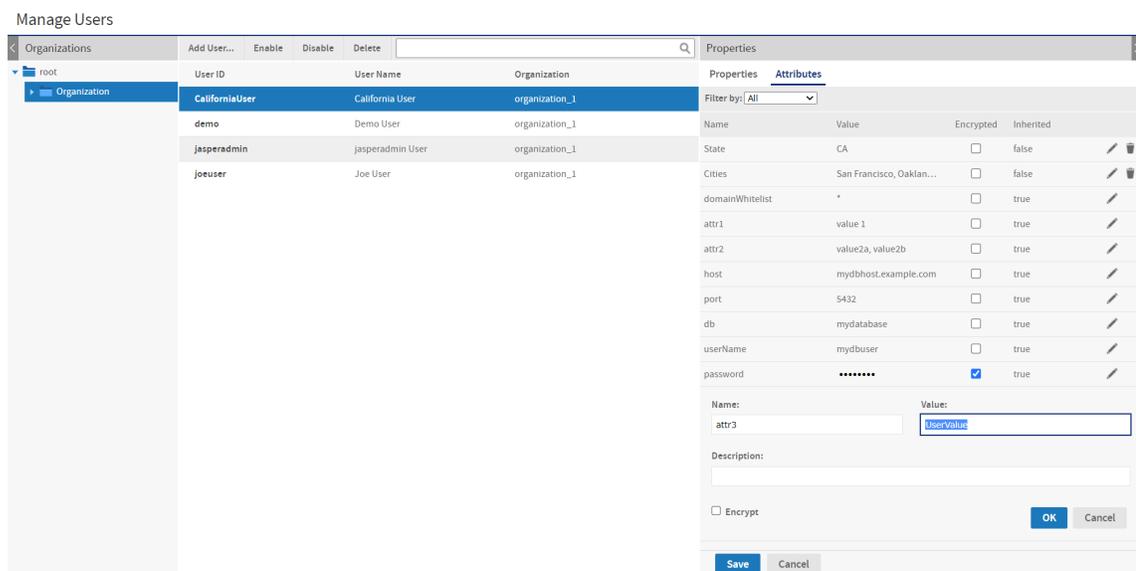
Figure 26: Adding a User Attribute

8. To modify an attribute, click the edit icon ✏. In the edit dialog, modify the desired fields. Click OK to close the dialog and then click Save to modify the attribute.
   You can also modify the attribute's encryption by using the check box in its table row. After confirming any changes, click Save to make them take effect.
   When modifying a user attribute, be aware of the following:

   - Inherited attributes belong to a parent organization or the server level and are shown at the user level to display the hierarchical attribute values. Any modification to an inherited attribute actually creates a local attribute definition with the modified parameters. In the hierarchy of attributes for this user, the new attribute takes precedence over the previously inherited attribute.

   - Changing the name of a locally defined attribute is equivalent to deleting the original attribute and adding a new attribute with the same value. Because this may impact features that reference attributes, you are asked to confirm the name change.

   - Removing encryption does not decrypt an encrypted attribute. To safeguard encrypted values, removing the encryption on an attribute also erases its value. Click the edit icon to give the attribute a new value.

9. To delete a user attribute, click the delete icon 🗑 in the attribute row and confirm the operation. Because this may impact features that reference attributes, you are asked to confirm the deletion. Click Save to make the deletion take effect.
   When deleting a user attribute, be aware of the following:

   - On deletion, some attributes remain in the list as inherited attributes. This indicates that the local definition of the attribute was deleted, but another attribute with the same name exists higher in the attribute hierarchy.

   - You cannot delete an inherited attribute because it belongs to a parent organization or the server level. To remove an inherited attribute, you must find where it is defined, then delete it at that level. The system admin can also change the permission of the attribute where it is defined to Execute Only or No Access so it does not appear on the user anymore.

Repository Administration

JasperReports Server provides a powerful and flexible environment for deploying and running JasperReports. The repository stores all the resources used to run and create reports, including data source definitions, JRXML files, datatypes, and helper files such as images. Administrators create the folders and resources so users can create, run, and save the reports they need. For administrators who want to customize the user interface, the repository also holds the CSS and image files that define themes.

The repository is structured as a hierarchy of folders based on the hierarchy of organizations. The JasperReports Server web interface enables you to browse the repository's resources, manage its folder structure, and secure its contents. This chapter covers the basic tasks of administering the repository, including:

- Creating folders and organizing repository objects.

- Managing references to data sources, images, fonts, and other resources upon which reports rely.

- Controlling access to resources in the repository through roles and object-level permissions.

Further information about the repository is available in the following sections:

- Overview of the Repository

- Other Resources in the Repository

- Themes

You can also access the repository using the web services (see the JasperReports Server REST API Reference) and APIs (see the JasperReports Server Ultimate Guide).

This chapter contains the following sections:

- Resource Types

- JasperReport Structure

- Managing Folders and Resources

- Multiple Organizations in the Repository

- Repository Permissions

Resource Types

Some resources are created interactively, others are uploaded from files. Many of these resource types are used to upload or define the components of reports. The following tables list the resource types that users and administrators can create in the repository:

Resource Types in the Repository

Resource Type	Description
Ad Hoc view	Users Create Ad Hoc views interactively in the Ad Hoc Editor by dragging and dropping columns of data onto a table, chart, or crosstab. Users can then explore their data by applying filters and performing pivot operations. An Ad Hoc view can also be saved as an interactive report and shared with other users.
Dashboard	A collection of reports, input controls, graphics, labels, and web content displayed together. Users create dashboards interactively in the Dashboard Designer and save them in the repository.
Domain	A metadata layer that selects, joins, and filters tables and fields from your data. A Domain can be the basis for an Ad Hoc report. Domains also support row-level security and localization of labels. Only administrators may create Domains. Domains are further documented in the JasperReports Server Data Management Using Domains.
Topic Domain Topic	A report or Domain that has been prepared for use in creating Ad Hoc views. Topics filter and select the fields that are available in the Ad Hoc Editor. Topics are stored in the Ad Hoc Components/Topics folder; they are further documented in the JasperReports Server User Guide.
JasperReport (JRXML report)	A JasperReport combines a JRXML file, a data source, and optional components such as input controls. Depending on the use case, both users and administrators create JasperReports in the server. For more information, see JasperReport Structure. Procedures for adding reports to the server are described in the JasperReports Server User Guide.
Report options	Reports with input controls allow you to save combinations of input

Resource Type	Description
	values so that you can run the report again directly. In the repository, report options are always listed under the original report.
Content resource	Report output of any format, either from running a report in the background or from scheduling a report. A content resource is a simple file that the repository allows users to view or download.
Data source	A connection that points to a database or other data store. Data sources define where data is stored for reporting. There are several types of data sources, based on the type of connection or location of the data: JDBC, JNDI, and several others. Only administrators may create data sources. For more information, see Data Sources.
Query	A database query string, for example in SQL. The JRXML doesn't necessarily include the query, in which case, you must define a query resource for use in the JasperReport.
Input Control	A complex type that specifies the values users can input for a report and how the input field appears when running the report, for example radio buttons or check boxes. Input controls depend on datatypes or lists of values to specify the format of the input.
Datatype	A basic resource that defines the format for input values, for example text, number, or date. A datatype may also specify a valid range for the input value.
List of Values	A basic resource that defines a list of arbitrary labels for input. Each label is associated with a value that can correspond to your data. For example, the Month Names List in the sample data associates the name of each month with a value 1 to 12.
File	A resource that stores a file in the repository, usually as part of a report, such as a JRXML file or image logo. The supported file formats are listed in File Resource Types.

Administrators and users can also manage OLAP resources in the repository, if their license supports Jaspersoft OLAP. For more information about OLAP and Mondrian resources, see the Jaspersoft OLAP User Guide.

Resource Types for OLAP

Resource Type	Description
Mondrian XML/A Source	A server-side XMLA source definition of a remote client-side XML/A connection.
OLAP Client Connection	Specifies how to retrieve data for an OLAP view. An OLAP client connection is either a direct Java connection (Mondrian connection) or an XML-based API connection (XML/A connection).
OLAP View	If you implement Jaspersoft OLAP, a view of multidimensional data based on an OLAP client connection and an MDX query. Like JasperReports, OLAP Views are collections of individual resources that define how to access and present the data.

JasperReport Structure

The repository resource that aggregates all information needed to run a report is called a JasperReport. A JasperReport is based on a JRXML file that conforms to the JasperReports Library that the server uses to render reports.

A JasperReport is a complex resource composed of other resources:

- The main JRXML file that defines the report

- A data source that supplies data for the report.

- A query if none is specified in the main JRXML.

    - The query may specify its own data source, which overrides the data source defined in the report.

- Input controls for parameters that users may enter before running the report. Input controls are composed of either a datatype definition or a list of values.

- Any additional file resources, such as images, fonts, and resource bundles referenced by the report template.

- If the report includes subreports, the JRXML files for the subreports.

The collection of all the resources referenced in a JasperReport is sometimes called a report unit. End users usually see and interact with a JasperReport as a single resource in the repository, but report creators must define all of the component resources.

Referencing Resources in the Repository

There are several ways to define and reference all the resources in a JasperReport.

In environments without JasperReports Server, reports are stored in the file system, and shared resources are usually stored on a network drive accessible to all developers and users. This solution is sometimes impractical, as you cannot always add such resources to the classpath, and the use of absolute paths has its own limitations. Also, storing the resources in the file system discourages their reuse. Developers may invest time recreating resources because they do not know they exist.

By storing resources in the repository, JasperReports Server makes it easy and reliable to share resources such as images, style templates, and subreports among reports. The repository mimics a folder and file structure, so references to external files can be handled as references to external resources in the repository.

The repo:syntax used in some previous versions is no longer required. Regular file paths are recognized and managed within the repository.

When you upload your JRXML to the repository, your file references become valid repository references, and you can store all your resources in well-known locations in the repository. This simplifies the process of uploading your reports, because you don't have to upload the resources each time. Also, you can manage these resources either through Jaspersoft Studio, the JasperReports Server user interface, or the server's REST API. For example, when you update a logo image, all reports that reference that resource displays the new logo.

Absolute References

Absolute references are the URIs of resources in the JRXML of a report. A path may refer to the file system where the JRXML was created, but when uploaded to the server, it refers to folders in the repository.

The path must start with one of the following:

> / to represent the root of the current user's organization. For example, /images/logo is the path to a resource in the /images folder.

../ to represent the folder where the JasperReport is uploaded. For example, ../myLogo is the path to a resource in the same folder as the JasperReport.

As with a file system path, the repository path is composed of the resource ID of every parent folder, ending with the ID of the resource. Jaspersoft Studio supports absolute references by allowing you to drag resources from the repository tree view into the design area.

> If you implement organizations, the absolute path is relative to the user's organization, as described in Multiple Organizations in the Repository.

When uploading the JRXML with absolute resource references as part of a JasperReport in the server, you need to ensure only that the resource with the given path exists in the repository before running the report. When the report runs, the server locates the resource in the repository and uses it to render the report.

Because file resources such as images, fonts, and JARs are the only resources for which you can create references directly in JRXML, they are the only resources for which you can create absolute references.

One disadvantage of absolute references is that JasperReports Server does not maintain the dependency between the JRXML and the absolute reference. When uploading the JRXML, there is no warning if the resource does not exist, and the server allows you to delete the resource from the repository even if it is still being referenced. If the resource is not available, running the report fails with an error.

Local Resources and External References

JasperReports Server provides more flexibility and power when you use indirect references instead of absolute references. Indirect references are placeholder names that must be manually linked to the resource when uploading the JasperReport. The syntax for an indirect reference contains only a resource placeholder name, for example:

logoImage

When you upload a JRXML with an indirect reference, the server prompts you to provide the resource. You have two options:

- Create a new resource, in this case by uploading an image that becomes part of the JasperReport. This is called a local resource. You cannot access this resource from elsewhere in the repository, it exists only within the JasperReport.

- Select a resource from the repository. This is called an external reference because it is external to the JasperReport. Any number of reports can link to the same external resource, and the resource can be managed independently of them.

While indirect references require slightly more work than absolute references in the JRXML, the server manages the dependency. Local resources exist as part of a JasperReport, and external references cannot be deleted until they are no longer referenced.

In cases where you do not want to reference existing resources, local resources allow reports to be highly customized, and self-contained. A local resource defined inside the JasperReport has all the same properties as a repository resource, but it is not accessible in the repository. Users must edit the JasperReport to access any resources it defines locally.

Indirect references are used implicitly in several other cases when you define a JasperReport:

- The main JRXML itself is either a local resource created by uploading a file or an external reference to an existing JRXML file resource in the repository.

- Every report must have a data source, and JasperReports Server gives you the option to either create a new local resource or use an external reference to an existing data source.

- Every report must also have a query that matches its data source. You may choose to create a local query resource or use an external reference to an existing query.

- Parameters in a report are implicitly handled as an indirect reference to an input control. For every parameter named in your main JRXML, you must define an input control either as a local resource or external reference.

Every level of indirect referencing is independent of the other. For example, when creating a JasperReport, you may choose to create an input control as a local resource, but that input control may have an external reference to its datatype. The server still manages the dependency between the local input control and the datatype resource in the repository.

Local resources and external references are used by several resources, for example when creating input controls, query resources, Domains, and OLAP resources.

References in Subreports

A subreport is a subordinate JRXML file within a JasperReport. As with all other resources referenced by the main JRXML, the subreport JRXML file may be specified by an absolute

reference, a local resource, or an external reference.

As a JRXML file, a subreport can reference other resources of its own. However, the subreport is run as part of the main JRXML, and any references in the subreport are interpreted relative to the JasperReport resource (represented by the main JRXML) and the context in which the JasperReport is being run.

Data Snapshots

Report resources can also store a snapshot of the report data. A snapshot is a copy of the data that the query returns when the data is refreshed. This data snapshot is an internal structure not visible or accessible from the repository. However, when data snapshots are enabled, a data snapshot is stored in the repository with each report. When users open a report, the report viewer retrieves and displays the data from the snapshot. Users then have the option of refreshing the data in the report viewer, and if they have permissions, saving the data snapshot back into the report resource.

For more information about interacting with data snapshots, see the JasperReports Server User Guide. To enable snapshots, see Enabling Data Snapshots.

Managing Folders and Resources

Administrators and users with the proper permissions can create, modify, move, and delete folders and resources in the repository. The specific roles and permissions of the user determine the actions available. For the definition of the permissions on folders and resources, see Repository Permissions.

One responsibility of an administrator is to set up an environment for users to create and save dashboards, Ad Hoc views, and reports. That usually means to create a folder structure where users have write permission. Users with write permission can also create their own sub-folders to store their reports and dashboards.

Only administrators can create data sources in the repository. So, although users with write permission can upload JRXML files and define resources, they cannot run their reports without data sources. For this reason, we recommend that administrators create data sources and other shared resources in the repository. For example, uploading one shared logo file and keeping it up to date is better than many users uploading and maintaining their own separate logo files.

Resource IDs

Each resource, including each folder, has an ID, a name, and an optional description:

- The ID is used internally to reference a resource. The ID must be unique within its folder, but may exist in multiple folders.

- The resource name is displayed in the repository.

- The description, if defined, appears in the repository and in the tooltips.

As in a file system, the IDs of nested folders define the path to a resource. For example, the path to a report might be: /reports/samples/Freight.

To view the name and resource ID of a resource, right click the folder's name or the resource in the repository or search results and select Properties... from the context menu.

Figure 27: Resource Properties Dialog for a Writable Resource

If you have write or administer permission as shown in the figure, you can also edit the name and description of the resource. For some operations such as export, you need the path, also called repository URI, which you can copy from this dialog.

> The path that is displayed is always relative to the logged in user's organization. For example, the following paths are the same resource for different users:
>
> superuser - /organizations/organization_1/images/JRLogo
>
> jasperadmin|organization_1 - /images/JRLogo

Creating Folders

Any user with write permission on a folder can create new sub-folders.

To create a folder

Procedure

1. Log in as a user who has write permission to the parent folder.

2. Select View > Repository and locate the parent folder in the Folders panel.

3. Right-click the parent folder and select Add Folder from the context menu. The Add Folder dialog appears.
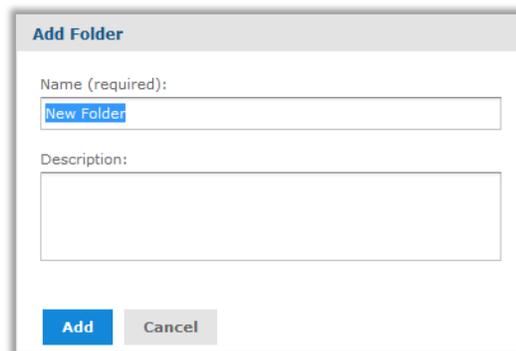
Figure 28: Add Folder Dialog

4. Enter the folder name and, optionally, a description, then click Add.
   The folder is created in the repository. The name appears in the hierarchy of folders. The description is visible only when viewing the properties of the folder, as shown in Resource Properties Dialog for a Writable Resource.
   New folders and their future contents inherit the permissions of their parent folders. Users with Administrator permissions can change permissions folders, as described in Setting Permissions.

Adding Resources

Each resource has different requirements. For example some are created from uploaded files, whereas others are created by defining values in a wizard. The procedures for adding each type are available in the documentation below:

- Interactive resources such as Ad Hoc views and dashboards are described in the JasperReports Server User Guide.

- JasperReports are covered in the JasperReports Server User Guide.

- Domains are covered in the JasperReports Server Data Management Using Domains.

- Mondrian and OLAP resources are covered in the Jaspersoft OLAP User Guide.

- Data sources are explained in the chapter Data Sources.

- Queries, data types, lists of values, input controls, and file resources are explained in the chapter Other Resources in the Repository

Most resources are created through the Add Resource menu item on the context menu for folders in the repository. In the following figure, you can see the full menu and submenu with all the resources administrators can create:
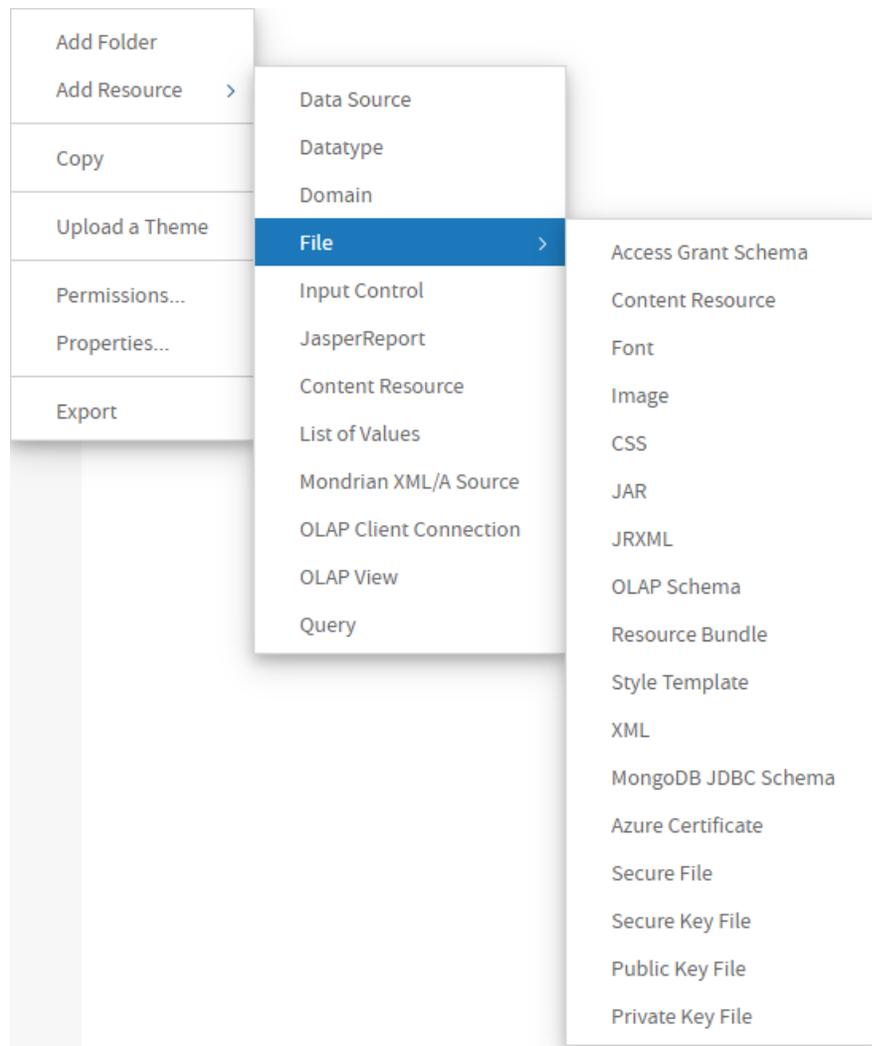
Figure 29: Add Resource Context Menu Expanded

For every resource you create, you must specify a name and resource ID for referencing the resource in the repository. Each wizard also has one or more pages for specifying the values and controls specific to the resource.

> New resources inherit the permissions of the folder in which they are created. Administrators can change the permissions on the new resource, as described in the section Setting Permissions.

Renaming Folders and Resources

Any user with write permission on a folder or resource can change its name and description.

> 📝 You cannot change the name of an organization's top-level folder in the way described here. The name of the top-level folder defaults to the name of the organization. So to change the name of the folder, you have to change the name of the organization, as described in the section Editing an Organization.

To rename a folder or resource

Procedure

1. Log in as a user who has write permission for the folder or resource.

2. In the repository, browse or search for the resource. For renaming folders, select View > Repository and locate the folder.

3. Right-click the object and select Properties... from the context menu. The Properties dialog appears.

Figure 30: Properties Dialog for a Report Resource

You can change the folder or resource's name and description, but not the ID. The ID is permanent once the resource is created.

4. Click Submit to save your changes.

Copying and Moving

The repository interface lets any user with the proper permission copy or move both resources and folders. Copying requires read permission on the source, moving requires delete permission on the source, and both require write permission on the destination folder.

You can drag-and-drop the objects, or you can copy-paste or cut-and-paste them from their context menus. Folders must be moved one at a time, but multiple resources from the same folder can be copied or moved together.

Copying and moving actions are not possible on the search interface, only on the repository interface showing the list of folders. Currently, it is not possible to create a copy of a resource in the same folder.

> The moved objects inherit their permissions from the destination folder. They do not keep the permissions they had before the move. If you want the objects to have other permissions, you can set new permissions after the move (see Repository Permissions).

To copy or moving folders and resources

Procedure

1. Log in as a user who has the required permissions for the folder or resource.

2. Click View > Repository, and expand the folders to display the object to be copied or moved.

3. Right-click the resource and select Copy or Cut (delete permission is required to cut a resource).

    You can select multiple resources with Control-click or Shift-click, but only a single folder.

4. Right-click the destination folder and select Paste in the context menu (write permission is required on the destination folder).
    Alternatively, you can drag to move the selected resource or folder to the destination folder. To copy, press and hold the Ctrl key then click and drag. When dragging resources, the destination folder is highlighted in blue if you have write permission to

it.

Editing Resources

The procedure for editing a resource depends on the resource type. All of the dialogs are available by right-clicking on the resource in the repository and selecting the proper action from the context menu. In nearly all cases, the dialog for editing is the same one that was used to create the resource.

Editing Resources

Resource Type	How to Edit
Ad Hoc view	Right-click the Ad Hoc view in the repository and select Open, then modify the view interactively. After changing the content, you can overwrite the existing view or save it as a new view. For more information, see the JasperReports Server User Guide.
	Reports created from Ad Hoc views are saved in the same format as JasperReports, but the resources referenced in the report unit are generated by the Ad Hoc Editor and should not be modified. There is one exception: administrators may create a JSP file and set it as a custom report view.
Ad Hoc Topic	Topics are JasperReports that are located in the Ad Hoc Components/Topics folder. Right-click the report and select Edit, then modify it like a JasperReport. For example, if the Topic contains a query as a local resource, you can edit it. For more information, see the JasperReports Server User Guide.
Dashboard	Right-click the dashboard in the repository and select Open in Designer, then modify the dashboard interactively. After changing the content, you can overwrite the existing dashboard or save it as a new dashboard. For more information, see the JasperReports Server User Guide
Domain	Administrators only: right-click the Domain in the repository and select Edit, then use the Domain Designer to make changes. For more information, see the JasperReports Server Data Management Using Domains

Resource Type	How to Edit
Domain Topic	Administrators only: right-click the Domain Topic in the repository and select Edit, then use the Data Chooser dialog to make changes. For more information, see the JasperReports Server User Guide
JasperReport	To change the layout of an interactive report, users may Run the report, then change the column filters or sorting. Users may save the report, either by overwriting the original or as a new copy, depending on their permissions. To change the definition of a report, right-click the report resource and select Edit. Then you can change the data source, input controls, or file resources referenced in the JasperReport. For more information, see JasperReport Structure.
Report Option	Expand the saved options under a report, then right-click one of the report options and select Edit. You can change the parameter values stored as input controls to the report.
Content Resource	A content resource is report output that is stored as a file stored in the repository. These files cannot be edited, only downloaded or deleted.
Data Source	Administrators only: right-click the data source, then select Edit from the context menu. For more information, see Data Sources.
Query	Right-click the resource, then select Edit from the context menu on these resources. You edit these resources in the same dialog that you used to create them. For more information, see Other Resources in the Repository.
Input Control	
Datatype	
List of Values	
File	

When editing a resource, you have several limitations:

- You cannot change a resource's ID. If you need to change an ID, you have to create a new resource and delete the old one.

- You cannot change the location of a resource. To change the location of the resource, see Copying and Moving.

- For file resources, you cannot see the name of the file that was uploaded, nor in most cases download and view the contents of the file. Your only option is to upload a new file to replace the old one.

Deleting Folders and Resources

Users with delete permission on folders or resources can delete those objects from the repository. To delete a folder, you also need to delete permissions on all the resources and folders in that folder. Folders must be deleted one at a time, but multiple resources can be deleted together.

You cannot undo a delete.

The repository will not allow you to delete resources currently referenced by other resources. For example, an input type used by a report or a properties file used by a Domain cannot be deleted as long as the report or Domain still references them.

To find the resources that reference the one you want to delete, you need to look at each report, view, Ad Hoc Topic, or Domain that you suspect of referencing it. When you edit the definition of a JasperReport or a Domain, you can see the resources it references. Then you can either remove the reference from the resource or delete the entire resource containing the reference.

To delete a folder or resource

Procedure

1. Log in as a user who has delete permission for the folder or resource.

2. In the repository, browse or search for the object to be deleted.

3. Right-click the object and click Delete in the context menu.
   In the repository view, you can select multiple resources and click Delete in the tool bar or in the context menu. In the list of folders, you can delete only one folder at a time, but all contents of that folder will be deleted. In the search results, you can select multiple resources and right-click to select Delete in the context menu.

Multiple Organizations in the Repository

If you implement multiple organizations, there are certain considerations when designing the repository structure for your deployment.

Organization Folders

In the repository, each organization has its own branch, contained in a folder named after the organization. JasperReports Server automatically restricts a user's view and access to their organization's folder. Organization admins can create any folder structure needed within their organization.

Top-level organizations are contained in a folder called Organizations at the root of the folder hierarchy. The top folder of a suborganization is contained in a folder called Organizations in the parent organization. Administrators can view and browse the Organizations folder, and the folders of any suborganizations. As administrators of the parent organization, they can also create folders and resources in suborganizations.

> By default, users in an organization can also view and create folders and resources in any suborganization. An administrator can control this by changing the folder permissions.

The Organizations folder in every organization is a special folder that is managed by the server. Administrators can't create folders or resources directly in the Organizations folder. The server creates the folder for each suborganization when the administrator creates a new organization through the Manage > Organizations page. Administrators can create folders and resources in the Folder Template folder in the Organizations folder; these resources are copied into new organizations. For more information, see Default Folders for Organizations.

Design Considerations

Careful design of the JasperReports Server repository leads to a clear and robust BI environment and easy, secure access for users. One of the main decisions is how you want your organizations and users to access resources: which resources are shared across organizations as opposed to those specific to an organization. Consider these use cases:

- Organizations have private resources - Data sources, reports, an other resources are stored in each organization's own folders, and perhaps only a few resources such as company logos would be shared among them.

- Organizations share resources - Resources are kept in the public folders where they can be used by all organizations and users. You may have common data sources and reports across customers, but the underlying data is partitioned by organization. Data level security restricts what users see when running public reports and OLAP views.

- Organization share resources, but have some customizations - For example, users in the organization create reports that are private and stored locally, but they access resources in the public folders.

- Organizations are hierarchical - One organization contains others. By default, the parent organization can access all the resources of its child organizations. If you don't want this, you must avoid creating suborganizations or customize the server's multi-organization architecture.

Referencing Resources in the Repository

All resources in the repository can be referenced by Universal Resource Identifiers (URIs) that specify resource names and folder paths. Because of the hierarchy of organizations, references are relative to the user accessing them. JasperReports Server transforms relative references into actual resource locations in the repository based on the user's organization and the organization's main folder. By default, folder locations are transformed in the following ways:

- For organization members, locations in /public are not transformed, but those in the organization's main folder are transformed to the actual location, for example, /organizations/organization_1.
  So if a user in org_1 runs a report that references /images/myLogo image resource, the actual path in the repository that is fetched is /organizations/org_1/images/myLogo. If the report also references /public/sharedLogo, the server fetches the resource in /public/sharedLogo.

- For a system admin (superuser) , locations throughout the repository are not transformed. They see the actual repository path names.

  If a system admin runs the same report in the example above, the reference to /images/myLogo attempts to fetch a resource named /image/myLogo, which works only if there is a folder at the root of the repository named images with a copy of myLogo. The report fails (or is missing a resource) when run by the system admin, unless he logs in as that user through the Manage > Users page.

This transformation enables URIs to reference different resources depending on the organization of the user who accesses them. For example, a report may have an organization-specific logo as an image. We can set up the report as follows:

- Logo URI specified in the JRXML: /images/orgLogo. When transformed for each user, the URI accesses a location relative to his organization's main folder.

- Every organization using this report must have a folder named /images containing an image resource with the ID orgLogo. When a user in any organization runs the report, the server fetches the organization-specific image and displays it.

There are three exceptions to references being transformed. In these cases, the references must be literal:

- In report units, references to data sources, JRXMLs, or input controls.

- In OLAP views, references to OLAP connections.

- In OLAP connections, references to data sources or schemas.

Also, because these references are not transformed, you must observe the following:

- For maintenance tasks on an organization's report units, OLAP views and OLAP connections, you must log into that organization and do the tasks there. You cannot administer the resources as superuser or another organization's admin.

- The three resources (report units, OLAP views and OLAP connections) cannot reference objects across organizations or even in their own parent organization. The reference would not be transformed; it would be taken literally and would fail. For example, if the data source for a report unit is in the /dataSources folder of This_Org, users in That_Org cannot access it because their reference cannot cross organizations.

> To test the absolute references, you should log in as an admin of the organization using the references. See Testing User Permissions.

Best Practices

Here are some best practices for resources in a repository shared by multiple organizations:

- The system admin (superuser) must log in as an organization user to maintain or run an organization's resources.

- Resources with absolute references to resources in organization folders work only for the users within the organization or the parent organization.

- If a JRXML that accesses organization resources with URIs must run across organizations, all organizations must have identical folders, object names, and expected object types for those resources.

- The public folder should be used for resources that are shared across organizations.

Repository Permissions

Permissions on folders and resources determine what users see in the repository and what actions they're allowed to perform. In the following table, the actions granted by each permission include all of the actions granted by permissions above it, except for the No Access permission. The actions granted by each permission strictly exclude all of the actions granted by permissions below it.

Permission	Actions Granted on Repository Folders and Resources
No Access	Users can never see or access the folder or resource, either directly in the repository or indirectly when running a report, Ad Hoc view, dashboard, or OLAP view. If one of these reference a resource that is set to no-access, the user will see an error when trying to run it.
Execute Only	Users can never see the folder or resource in the repository, but they can run reports, dashboards, Ad Hoc views, or OLAP views that access them. Typically, data sources are execute-only so that users may run reports but not see database connection details.
Read Only	- See the folder or resource in any server - See the properties of a folder or a resource - Copy a folder and all of its readable contents - Copy resources individually or in bulk - View (run) a report, dashboard, Ad Hoc view, or OLAP view - Run a report in the background - Schedule a report to run later

Permission	Actions Granted on Repository Folders and Resources
Read + Delete	• Cut (move) a folder and all of its contents • Delete a folder and all of its contents • Cut (move) resources individually or in bulk • Delete resources individually or in bulk
Read + Write	• Save report options for a JasperReport • Delete report options • Copy resources to a folder with this permission • Edit and modify resources
Read + Write + Delete	• Add a subfolder • Add (create) a resource in a folder (except Domains and data sources) • Paste into a folder (copy or cut) • Save a new Ad Hoc view, report, or dashboard in a folder • Save the output of a scheduled report in a folder • Rename a folder or resource and change its description string • Open an Ad Hoc view in the Ad Hoc Editor or a dashboard in the designer • Modify and overwrite an existing Ad Hoc view, report or dashboard • Add a JasperReport resource to the repository (upload a JRXML) • Edit the definition of a JasperReport resource in the repository (replace the JRXML)
Administer	• Set the permissions (by role and by user) on a folder or resource. This effectively delegates certain repository administration tasks.

Permission	Actions Granted on Repository Folders and Resources
Administer and ROLE_ ADMINISTRATOR	• Add (create) a Domain or a data source

Permissions apply when browsing or searching the repository and when using any dialog that accesses the repository, like browsing folders to save a report. Note that:

- Copying does not preserve the permissions on an object. Users may copy a read-only object, paste it into a read-write folder, then edit the object. For more details, see Copying and Moving.

- Copying and cutting (moving) actions can be completed only by a user with Read + Write + Delete access to the folder in which the object is pasted. For more details, see Copying and Moving.

- Cutting, deleting, and setting permissions on folders is allowed only if the user has the same permission on all folder contents.

- Cutting and deleting resources in bulk is allowed only if the user has at least Read + Delete permission on all selected resources.

- Deleting a resource is allowed only if no other resources rely on it. For more details, see Deleting Folders and Resources

Inheriting Permissions

According to the permission architecture, there may be a permission setting for every user and role on every folder and resource in the repository. To simplify the definition of permissions, JasperReports Server supports the inheritance of permissions from the parent folder to a folder or resource. If no permission is explicitly defined for a user or role on a given folder or resource, the user or role has the same access permission that is defined on the parent folder. When a permission is defined explicitly, that permission is enforced, regardless of those on the parent folder.

Using permissions, you can manage large hierarchies of content and keep them secure. When you set a permission explicitly, that permission for a given user or role is inherited recursively by all of the folder's contents and subfolders, unless they have an explicit definition of their own. Permissions assigned on an organization's top folder are inherited

across the entire organization. Permissions set on the root folder or Organizations folder by the system admin are inherited across multiple organizations.

For example, the system admin can make all organizations read-only by default to ordinary users, and each organization admin can make specific folders writable so that users can store their reports and output.

Cumulative Permissions

Because permissions can be assigned to both users and roles, a user belonging to one or more roles may have multiple permissions defined or inherited on any given folder or resource. In fact, every permission must be defined on the root, even if it has the default value of No Access, and therefore every role- and user-based permission on every folder and resource has a setting through inheritance. So, for every folder or resource, every user has their own user-based permission and the permission assigned to the ROLE_USER.

How does JasperReports Server determine the effective permission from the many that apply? Permissions in the server are strictly cumulative, meaning that the least restrictive among the set of all permission applies. Even if a more restrictive permission, such as No Access, is set explicitly, the less restrictive permission such as Read-Only applies, regardless of whether it is inherited or set explicitly. This is why all default permissions at the root are set to No Access initially.

Administrator Permissions

The JasperReports Server authorization architecture distinguishes between administrators and all other users. Administrators are defined as users with either ROLE_SUPERUSER (professional edition only), ROLE_ADMINISTRATOR, or both. By design, system administrators with the ROLE_SUPERUSER always have irrevocable Administer access to the entire repository, including the contents of every organization. The system administrator cannot modify the permissions for ROLE_SUPERUSER, to prevent being locked out or unable to administer some resource. Therefore, the system administrator can set permissions for all other users, on any folder or resource, and in any organization if necessary. In particular, the system administrator can modify permissions for ROLE_ ADMINISTRATOR, for example to share some resources across all organizations by making them read-only to everyone, including the organization admins.

Organization admins are organization users with the ROLE_ADMINISTRATOR, like the default jasperadmin user created in every organization. By default, organization admins

have the Administer permissions to everything in their organization, except what the system admin has changed to a lesser permission. However, organization admins cannot change the permissions granted to ROLE_ADMINISTRATOR, to prevent them from overriding the settings of the system admin and from locking themselves out of a folder or resource.

Execute-Only Permission

As in file systems, execute-only permission in JasperReports Server allows running reports, dashboards, Ad Hoc views, and OLAP views to access a resource, but keeps the resource from appearing in the repository.

Execute-only permission applies to folders as well, keeping them from appearing in the folder tree when users browse the repository, yet still allowing the resources they contain to inherit the execute-only permission. This is useful for hiding folders and resources such as data sources that only administrators and data analyst roles need to access in the repository. However, if your execute-only folder contains read-only resources, those resource are hidden when browsing folders but can be found with a repository search.

As with all other permissions, execute-only permission is either role-based or user-based, so only certain users can access a resource from a running report.

> If you have data or sensitive content in a resource, always set No Access permission for users or roles that must not be able to access it.
>
> Hiding a resource with execute-only permission does not protect against access, because malicious users who find the resource ID may be able to create a report, dashboard, or OLAP view that extracts the sensitive content.

Default User Permissions

For all non-administrator users, the default permission at the root is No Access and any permissions must be explicitly defined. In practice, the default installation of the repository contains sample data with a mix of no access, execute only, read only, and read-write permissions that allow the sample users to access folders and resources. The sample permissions demonstrate a common approach to permissions, allowing users to see the resources they can access and hiding the ones they can't, while administrators have full access.

We recommend you familiarize yourself with permissions by viewing and setting permissions in the sample data, as described in the following section.

Setting Permissions

Administrators can assign permissions to access any folder or resource throughout the repository. Users with the Administer permission on a folder can assign permissions to that folder and any contents that inherit the permission. Users granted Administer permission to a resource can set the permissions only on that specific resource.

To set permissions on a folder or resource in the repository

Procedure

1. Log in as a user with administrator privileges.

2. Browse or search the repository for the folder or resource.

3. Right-click the object and select Permissions... from the context menu:
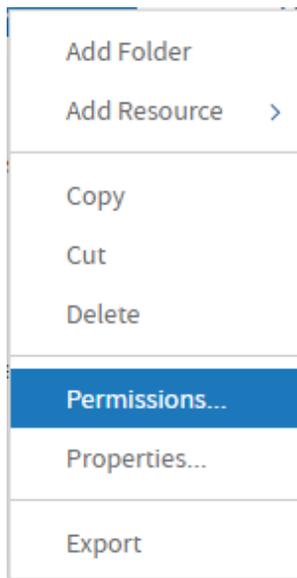
The Permissions dialog opens showing the permissions in effect for the selected object. By default, it first shows the permissions given to roles. Permissions that are inherited from the object's parent are indicated by asterisks (*).
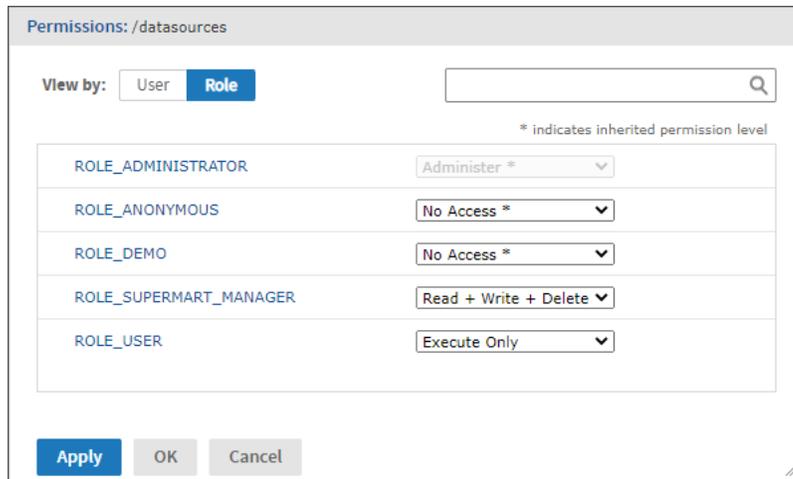
Figure 31: Permissions Dialog Showing Permissions by Role

In systems with multiple organizations, the users and roles displayed include only those within the scope of the user. For example, in the default single organization, the organization admin (jasperadmin) can't see the permission for the system admin (superuser) or for ROLE_SUPERUSER.

In the previous figure, you can see the default role-based permissions on the sample Data Source folder as seen by the organization admin (jasperadmin). Members of certain roles can see and modify the resources stored in this folder; these roles likely correspond to users such as data analysts. Regular users have execute only permission so they do not see this folder, but the reports they run can access its contents. Administrators are prevented from changing the permission for their administrator role or user name, to prevent them from removing their ability to set permissions.

4. In the dialog, click User to view the permissions assigned to specific users. Click Role when viewing user permissions to toggle back.

5. For each user or role, you can select a new permission from the drop-down.
In the next figure, you can see the default user permissions on this folder. In the default installation, all permissions are defined by role; so, all user permissions are No Access inherited from the root. The figure shows a read-only permission being granted to the sample end user. This enables joeuser to see but not modify the Data Sources folder and its contents. For all other end users, the folder is still execute-only due to the settings in Permissions Dialog Showing Permissions by Role.
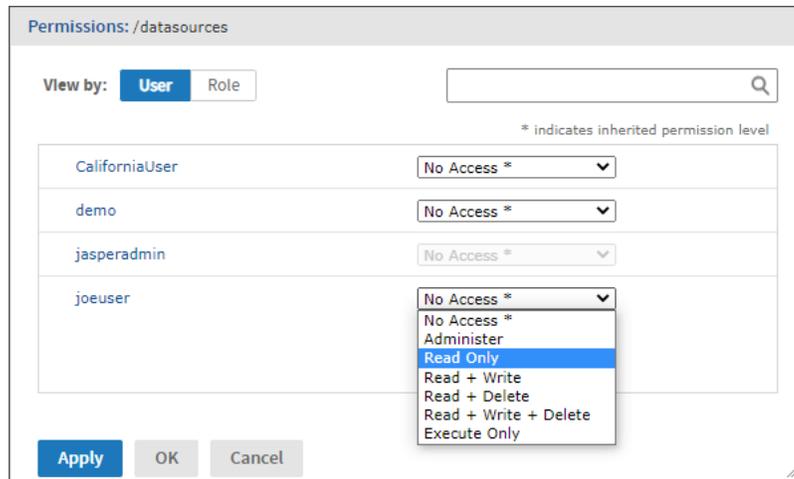
Figure 32: Permissions Dialog Showing Permissions by User

6. Click Apply to apply your changes. If you toggle between user and role permissions, first apply any changes you made.

7. Click OK to save your changes and close the permissions dialog when you're finished. You can open several permissions dialogs for different resources or folders at the same time while navigating the repository. This helps when trying to set permissions uniformly across several folders or organizations.

> There are two special cases when setting permissions:
>
> - If a resource inherits a permission, for example Read-Only, you cannot set the permission to the same value, at least not directly. You need to temporarily change the permission level on the parent folder, then set the explicit permission, then set the parent folder's permission back to the original value.
>   When a resource and its parent folder have been set to the same permission in this way, the permission dialog still shows the asterisk as if the permission were inherited. But if the parent is later given a different permission, for example Read-Write, the resource retains its explicit Read-Only permission instead of inheriting Read-Write.
>
> - To reset the permission level so that it once more inherits from its parent folder, select a different permissions level and click Apply, then select the permission with the asterisk and click Apply again.

Testing User Permissions

Once you have configured users, roles and permissions, we recommend that you test the permissions granted to a few representative users. We also recommend testing whenever

you add new users, roles, and resources or make any major modifications to your access control configuration.

To test user permissions

Procedure

1. Log in as an administrator.

2. Select Manage > Users.

3. Select the user's organization, then browse or search for the user whose permissions you are testing.

4. In the Users panel, select the user.

5. In the Properties panel, click Login as User. The selected user's Home page appears. The login information in the upper-right corner shows that you are logged in as that user.

6. Verify that the expected folders and resources are available in the repository. Make a note of any objects that should be there but are not, and any that should be hidden but are displayed.

7. When you have verified the user's permissions, click Log Out to exit that user's account.

8. To change the user's permissions, edit the permissions in the repository and modify the user or role definitions.

9. Continue testing until the user's permissions are satisfactory.

10. Repeat these steps with several representative users to ensure that your access control is properly configured.
    An untested access control configuration can't secure your data adequately.

Data Sources

A data source is a resource in the repository that specifies how and where to obtain the data displayed by reports, Domains, Ad Hoc views, and OLAP views. Administrators must define data sources before uploading reports that rely on them. Typically, a data source specifies the URI of the database server and the details you need to access it, such as a username and password.

JasperReports Server provides data source types for relational databases, most flavors of big data, and for specialized data such as Amazon Web Services and JavaBean data. Virtual data sources allow you to combine several data sources into one.

JasperReports Server can access any relational database that supports the SQL query language through the JDBC (Java DataBase Connectivity) API. In this case, you can configure two types of data sources in the repository:

- JDBC data source - Establishes a direct connection to the database server using its JDBC driver. After installation, JasperReports Server includes JDBC drivers to access the most popular databases: MySQL, PostgreSQL, Oracle, SQL Server, DB2, Salesforce. The server also includes a JDBC driver that can access Hadoop-Hive, MongoDB, Cassandra, TIBCO Data Virtualization, Google BigQuery, and more. If you want to install drivers for other databases, or if you want to use alternate drivers, the system administrator can upload and manage JDBC drivers through the UI. With JDBC data sources, JasperReports Server configures and manages the connections to the database. By default, the maximum number of simultaneous connections for each data source is 20.

- JNDI data source - Relies on the JNDI (Java Naming and Directory Interface) service of your application server to access a database connection. First configure your application server to install its JDBC drivers and configure its database connections. With JNDI data sources, the configuration of the application server determines the number of shared connections. Note that the application server connects to the database using JDBC, meaning that JNDI data sources are available for all databases that support JDBC.

JasperReports Server also supports some specialized data sources:

- Amazon Web Services (AWS) data source - Accesses data stored in your AWS data store using JasperReports Server, either on-premises or in the cloud.

- Microsoft Azure SQL data source - Allows you to access data stored in your Azure SQL Server database.

- Virtual data source - Allows you to combine multiple data sources into a single data source and join them within a Domain. You can also wrap a data source for big data to be used in a Domain.

- MongoDB - This custom data source for MongoDB supports Jaspersoft's own MongoDB Query Language. A MongoDB JDBC driver is also available.

- XLS and XLSX data sources - Allows you to generate reports based on data in the XLS and XLSX formats.

- File data source - Allows you to generate reports based on data in XML and JSON formats.

- Bean data source - Allows you to access data encapsulated in JavaBeans.

- Internal diagnostic data source - A custom data source for the server's own diagnostic data. The diagnostic information is available only to system admins (superuser by default). For more information, see Using the Diagnostic Data in Reports.

In the case of analysis data, JasperReports Server supports OLAP data sources (such as Mondrian and XML/A connections). For information about analysis data sources, refer to the Jaspersoft OLAP Ultimate Guide.

> You can extend JasperReports Server to support any custom data source. Custom data sources consist of Java implementation classes, a message catalog, and a Spring bean definition. For more information about custom data sources, see the JasperReports Server Ultimate Guide.

This chapter contains the following sections:

- Attributes in Data Source Definitions

- JDBC Data Sources

- Managing JDBC Drivers

- JNDI Data Sources

- AWS Data Sources

- Azure SQL Data Sources

- Snowflake Data Sources

- Hadoop-Hive Data Sources

- [MongoDB Data Sources](#)

- [Virtual Data Sources](#)

- [TIBCO Data Virtualization](#)

- [XLS and XLSX Data Sources](#)

- [File Data Sources](#)

- [Bean Data Sources](#)

Attributes in Data Source Definitions

You have two options for defining the parameters of a data source, such as the host and port number:

- Define the parameters statically, so that they are the same for every user.

- Have the server derive these parameters at run time based on the attributes you provide.

Attributes can be used to derive all data source parameters that are not selected from dropdown lists. For instructions on how to create attributes, see Managing Attributes.

When referring to an attribute in a data source definition, you can specify the attribute categorically or hierarchically:

- Categorical reference - If you specify a category for the attribute value, the server attempts to find that particular value of the attribute. If the attribute is not defined where specified, reports using this data source fails with an error. You can specify these attribute categories:

  - User - In the attributes defined on the logged in user.

  - Tenant - In the attributes defined on the organization of the logged in user.

  - Server - In the attributes defined at the server-level.

- Hierarchical reference - If you do not specify a category for the attribute, the server searches attributes hierarchically and uses the value of the first attribute it finds with the given name. This search starts with the logged in user, then proceeds to the user's organization and parent organizations, and finally to the server level. If the specified attribute is not found in any of these, reports using this data source fails

with an error.

The following figure is an example of the attributes used to define data source parameters.

Figure 33: Using Attributes for Data Source Parameters

In the example above, the following data source parameters are specified by attributes:

- {attribute('host','User')} - Host is derived categorically from the logged in user's attributes, because the "User" category is specified.

- {attribute('port','Tenant')} - Port is derived from the logged in user's organization attributes.

- {attribute('db','Server')} - the database name is derived from the server-level

attributes.

- The URL is generated automatically from the host, port, and database fields.

- {attribute('userName')} - The username is derived hierarchically, because no category is specified. JasperReports Server looks for a host first in the logged in user's attributes, then in the organization attributes, and finally the server-level attributes.

- {attribute('password')} - The password field can also reference an attribute, here a hierarchical attribute.

For information about attributes on users, organizations, and the server, see Managing Attributes.

JDBC Data Sources

JDBC data sources are direct connections to your database managed by JasperReports Server. To create one, you must provide the URL and credentials to access your database, along with any database-specific configuration parameters.

JasperReports Server includes JDBC drivers for the most used databases. If your database is not included, or if you want to use different JDBC drivers, the system administrator must upload the appropriate JDBC driver before creating a data source. For more information on JDBC drivers, see Managing JDBC Drivers.

To create a JDBC data source

Procedure

1. Log on as an administrator.

2. Select View > Repository, right-click a folder's name, and select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select JDBC. The page refreshes to show the fields required for a JDBC data source.

Figure 34: Setting the JDBC Data Source Type

4. Select the JDBC driver for your database. If your driver is listed as NOT INSTALLED, a system administrator must first upload the driver as described in Managing JDBC Drivers.

> Jaspersoft provides a set of JDBC drivers in the installed server. These drivers support a slightly different SQL syntax. If you see errors when running reports or creating domains that use scalar functions, see Working With Data Sources.

5. Enter the hostname, port, and database name for your database. The default hostname is localhost, and the default port is the typical port for the specified database vendor. The three fields are combined automatically to create the JDBC URL where the server will access the database. When specifying values for your JDBC data source:

    *   The JDBC drivers for some databases have their own unique fields and optional URL parameters. These are described in Unique JDBC Data Source Fields and JDBC Database URLs.

    *   You have the option to use attributes in the values of data source parameters. See Attributes in Data Source Definitions.

6. Fill in the database user name and password. These are the credentials the server will use to access the database.

Figure 35: Entering the User Name and Password

> The database user needs the privileges to run SELECT queries on the tables used in your reports. The server blocks any DROP, INSERT, UPDATE, and DELETE SQL commands through its SQL injection protection. In some cases, additional permissions may be required to execute stored procedures, depending on your configuration and needs. For more information, see Database Permissions.

7. If the date-time values stored in your database do not indicate a time zone, set the Time Zone field. When in doubt, leave the default Time Zone value (Use database setting).
    When date-time values are stored in a format other than local time zone offset relative to Greenwich Mean time (GMT), you must specify a time zone so that the server can properly convert date-time values read from the target database. Set the Time Zone field to the correct time zone for the data in the database. The list of time zones is configurable, as described in Specifying Additional Time Zones.

8. Click Test Connection to validate the data source. If the validation fails, ensure that the values you entered are correct and that the database is running. To diagnose JDBC connection issues, you can turn on logging as described in the troubleshooting section Logging JDBC Operations.

9. When the test is successful, click Save. The Save dialog appears.

Figure 36: Saving the JDBC Data Source

10. Enter a name for the data source and an optional description. The Resource ID is generated from the name you enter. If you haven't already specified a location, expand the folder tree and select the location for your data source.

11. Click Save in the dialog. The data source appears in the repository.

Creating a Trino Data Source

To create a Trino data source

Procedure
1. Add the JAR as explained in the section *Adding a Trino JDBC Driver* in JDBC Data Sources.

2. Enter URL as: jdbc:trino://<host>:<port>

3. You can also refer to the Trino documentation to get information on JDBC URL formats supported by Trino.

4. Enter **username**: *admin* and leave **password** as blank.

5. Click **Test connection**.

6. Once passed, click **Save** to save the data source in the repository.

Managing JDBC Drivers

To access a database from JasperReports Server using JDBC, you need an appropriate driver that is accessible in the server's classpath. The following drivers are installed by default:

- MYSQL (`org.mariadb.jdbc.Driver`)

- PostgreSQL (`org.progresql.Driver`)

- Snowflake (`net.snowflake.client.jdbc.SnowflakeDriver`)

- TIBCO® Data Virtualization (`cs.jdbc.driver.CompositeDriver`)

> The Amazon S3 output location is not validated during Athena connector creation.

Drivers for other databases can be downloaded from links on the Jaspersoft community website:

http://community.jaspersoft.com/wiki/downloading-and-installing-database-drivers

The system administrator (superuser) can add JDBC drivers for other databases in the following ways:

- During installation. For more information, see the JasperReports Server Installation Guide.

- Through the UI, if the setting is enabled in System Settings. As described in the following procedures, the system admin can add, replace, or remove JDBC drivers through the user interface, without needing to restart the server.

By default, no one can upload or update JAR files for JDBC drivers from the UI. Only the system administrator (superuser) can enable the setting and then manage the JDBC drivers, but once uploaded, the JDBC drivers are available to all administrators who create data sources. For more information, see the following sections:

- Enabling JDBC Driver Uploads

- Adding a JDBC Driver

- Updating a JDBC Driver

> JasperReports Server includes a JDBC driver for ElasticSearch that is not enabled by default. For information on enabling the driver and its limitations, see Enabling the JDBC Driver for ElasticSearch Data Sources.

> JasperReports Server also supports Trino. Trino drivers are not installed by default in JasperReports Server. Refer to Trino's JDBC driver documentation for more information. For instructions on uploading Trino drivers, see Adding a Trino JDBC Driver.

Enabling JDBC Driver Uploads

By default, no one can upload or modify JAR files for JDBC drivers from the UI. Only the system administrator (superuser) can enable the setting and then manage the JDBC drivers, but once uploaded, the JDBC drivers are available to all administrators who create data sources.

To enable this setting

Procedure

1. Select Manage > Server Settings and choose General from the left-hand panel.

2. Turn on the switch to allow uploading and updating JAR files for JDBC drivers.

For more information, see the following sections:

- Adding a JDBC Driver

- Updating a JDBC Driver

> After uploading a JDBC driver, Jaspersoft recommends disabling driver uploads again for security reasons. This avoids the risk of drivers being accidentally modified or malicious JAR files being uploaded.

Adding a JDBC Driver

Procedure

1. Log in as the system administrator (superuser).

2. Enable JDBC driver uploads, as described in Enabling JDBC Driver Uploads.

3. Select View > Repository, right-click a folder's name, and select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page.

4. In the Type field, select JDBC Data Source. The page refreshes to show the fields necessary for a JDBC data source.

5. The dropdown selector for the JDBC Driver field shows the available JDBC drivers and those that are not installed.
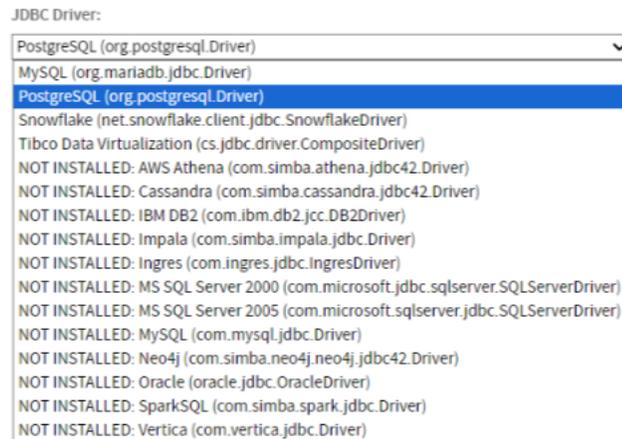
Figure 37: Viewing the List of Available JDBC Drivers

6. Select the driver that has not been installed, then click Select Driver. The Select Driver dialog appears.

7. If you have not yet obtained the driver, click the link to Jaspersoft's community website for Downloading and Installing Database Drivers. That page has links to the most commonly used JDBC drivers. After you download a driver to your file system, you can return to the Select Driver dialog.
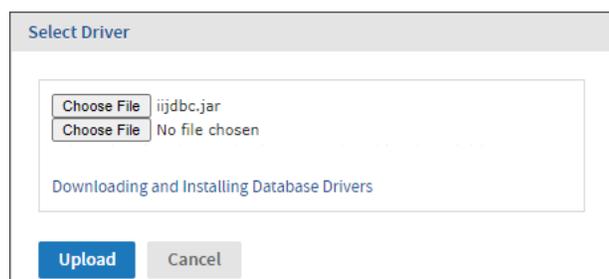
Figure 38: Adding a JDBC Driver

8. In the Select Driver dialog, click Browse to locate the appropriate driver JAR file. If your driver has more than one JAR file, click the Browse button that appears after selecting the first file.

9. Click Upload to install the driver and make it available immediately.

You can replace any driver that you upload with newer versions of the same driver. If you want to use the vendor's own driver instead of the JDBC driver, you can install it as a new driver as described in the *JasperReports Server Installation Guide*.

Adding a Trino JDBC Driver

Procedure

1. Log in as the system administrator (superuser).

2. Enable JDBC driver uploads, as described in Enabling JDBC Driver Uploads.

3. Select **View > Repository**, right-click a folder's name, and select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page.

4. In the **Type** field, select **JDBC Data Source**. The page refreshes to show the fields necessary for a JDBC data source.

5. The dropdown selector for the JDBC Driver field shows the available JDBC drivers and those that are not installed.

6. Select **Other**, then specify the JDBC Driver (required) as *io.trino.jdbc.TrinoDriver*.

7. Click **Select Driver**. The **Select Driver** dialog appears.

8. In the **Select Driver** dialog, click **Browse** to locate the appropriate driver JAR file.

9. Click **Upload** to install the driver and make it available immediately.

Updating a JDBC Driver

Procedure

1. Log in as the system administrator (superuser).

2. Enable JDBC driver uploads, as described in Enabling JDBC Driver Uploads.

3. Select View > Repository, right-click a folder's name, and select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page.

4. In the Type field, select JDBC Data Source. The page refreshes to show the fields necessary for a JDBC data source.

5. The dropdown selector for the JDBC Driver field shows the available JDBC drivers and those that are not installed.

6. To update a driver that has already been installed, select it from the list, then click Select Driver. The Select Driver dialog appears and notifies you that selecting a driver will overwrite the existing one.
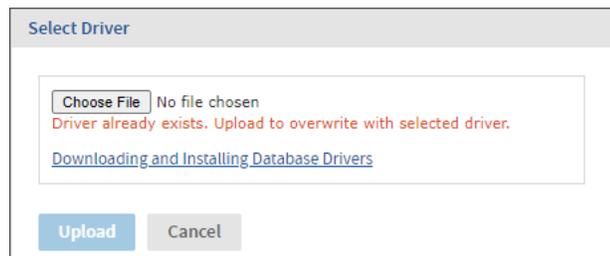
Figure 39: Updating a JDBC Driver

7. In the Select Driver dialog, click Browse to locate the JAR file for the new driver.

8. Click Upload to replace the existing driver and make it available immediately.

9. You can now use this driver to create a data source, and the driver will be installed when other administrators create data sources.

Removing an Uploaded JDBC Driver

Procedure

1. Log in as the system administrator (superuser).

2. Select Manage > Server Settings and choose Restore Defaults from the left-hand panel.

3. Locate the driver you uploaded in the list of properties. The drivers with the value

[SYSTEM] are the default drivers configured at installation time. Do not remove the [SYSTEM] drivers.

4. Click 🗑 beside the driver that you want to remove, then confirm your choice.
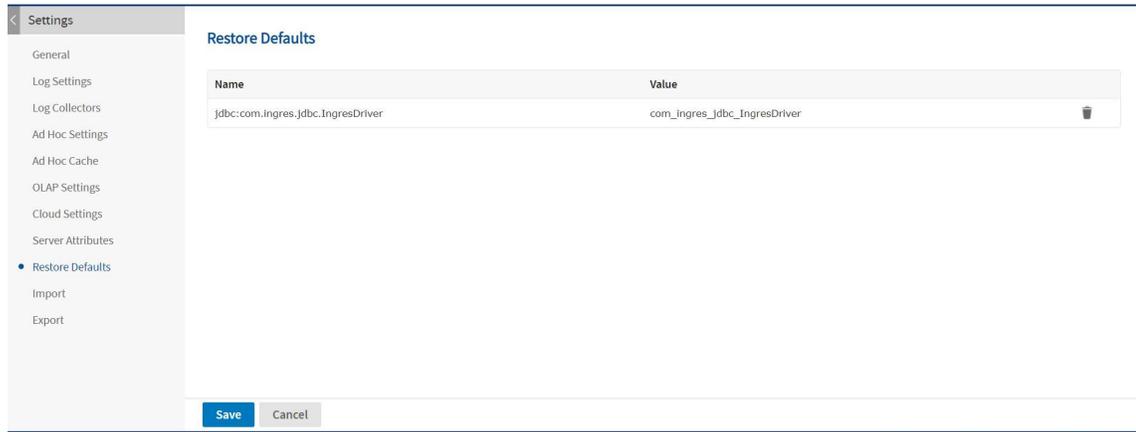
Figure 40: Removing an Uploaded JDBC Driver

5. Click Save to save your changes.

> 📝 If the JDBC driver you remove is one that updated a default driver, the default driver will reappear as an installed driver the next time you use the New Data Source wizard.

JNDI Data Sources

The JNDI data source accesses a database connection previously defined in the application server and published as a resource or service through JNDI (Java Naming and Directory Interface). Instead of specifying a driver and database as you do with JDBC data sources, you need to specify only the JNDI service name in your application server.

> 📝 Application servers use JDBC connections themselves to expose a database through JNDI. You must specify the JNDI service name of a JDBC connection. Your application server must also have the appropriate JDBC drivers and be configured to use them.
>
> Ensure the database user in your JNDI definition has the privileges to run SELECT queries on the tables used in your reports. In some cases, additional permissions may be required to execute stored procedures, depending on your configuration and needs. For more information, see Database Permissions.

For information about setting up a JNDI connection in your application server, see the following sections:

- JNDI Services on Apache Tomcat

- JNDI Services on JBoss

- JNDI Services on WebLogic

To create a JNDI data source

Procedure

1. Log on as an administrator.

2. Click View > Repository, expand the folder tree, and right-click a folder to select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select JNDI. The information on the page changes to reflect what's needed to define a JNDI data source.

4. Fill in the service name. This is the name the application server exposes through JNDI.
   You have the option to use attributes in the service name. See Attributes in Data Source Definitions.

Figure 41: JNDI Data Source Page

5. If the date-time values stored in your database do not indicate a time zone, set the Time Zone field. When in doubt, leave the default Time Zone value (Use database setting).

    When date-time values are stored in a format other than local time zone offset relative to Greenwich Mean time (GMT), you must specify a time zone so that the server can properly convert date-time values read from the target database. Set the Time Zone field to the correct time zone for the data in the database. The list of time zones is configurable, as described in Specifying Additional Time Zones.

6. Click Test Connection to validate the data source. If the validation fails, ensure that the values you entered are correct, that the database is exposed through JNDI, and that the database is running. Also see the troubleshooting section JNDI Services on Apache Tomcat.

7. When the test is successful, click Save. The Save dialog appears.

8. Enter a name for the data source and an optional description. The Resource ID is generated from the name you enter. If you haven't already specified a location, expand the folder tree and select the location for your data source.

9. Click Save in the dialog. The data source appears in the repository.

AWS Data Sources

Amazon Web Services (AWS) provide computation and data storage on demand in the cloud. We partner with Amazon to deliver business intelligence solutions based on AWS.

JasperReports Server supports two of the AWS database services as data sources for reporting:

- Amazon Relational Database Service (RDS)

- Amazon Redshift data warehouse

JasperReports Server can access either of these services when you define a data source with the correct configuration information and credentials. The AWS data source wizard uses the AWS credentials that you provide to discover RDS and Redshift data sources. Then it uses those credentials to properly configure security groups to maintain the connection between JasperReports Server and the AWS data source, even when the IP address changes. You can access AWS data sources from both stand-alone server instances that you maintain on your own computers and virtual server instances that you run on Amazon's Elastic Compute Cloud (EC2). For more information, see https://www.jaspersoft.com/cloud.

Creating an AWS Data Source

To create an AWS Data Source

Procedure

1. Log into JasperReports Server as an administrator.

2. Click View > Repository, expand the folder tree, and right-click a folder to select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select AWS. The information on the page changes to reflect what is needed to define an AWS data source.
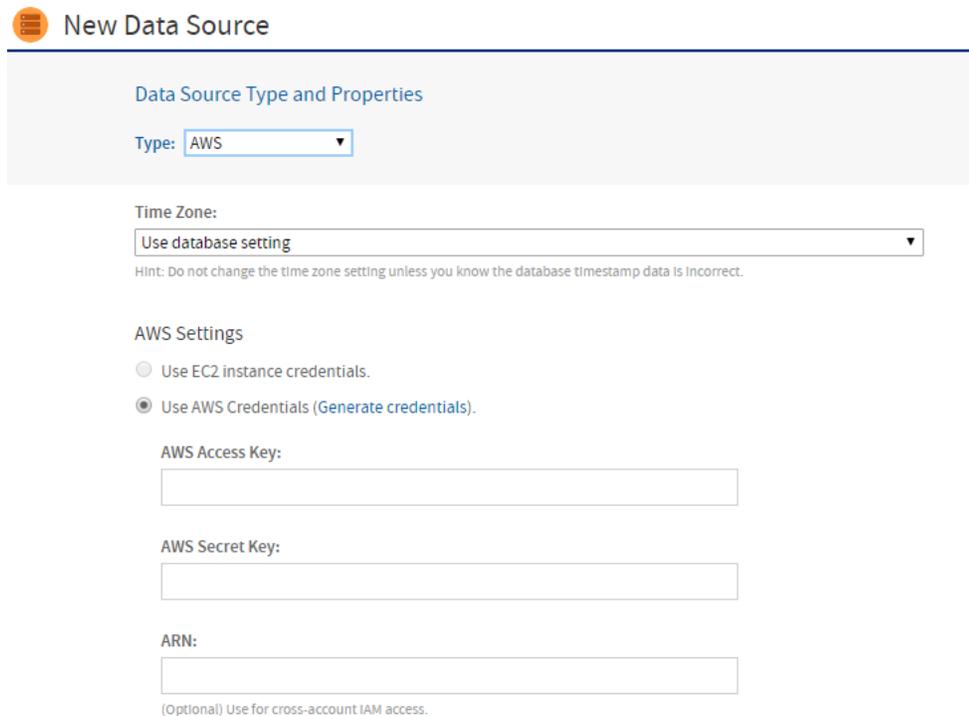
Figure 42: Selecting AWS Credentials

4. Under AWS Settings, specify your Amazon credentials in one of the following ways:

   - If your JasperReports Server is running in Amazon's EC2 service, and it has the

proper instance role assigned, the server detects this and automatically use your EC2 credentials. Using the EC2 instance credentials requires that the role was properly set up and assigned when the instance was created. If you are using the EC2 service, we strongly recommend that you use the EC2 credentials.

- If your JasperReports Server is not running on Amazon's EC2, enter the AWS credentials associated with the RDS or Redshift service. If you do not have AWS keys, click Generate credentials, then look for them on the Outputs tab for your Stack on the Amazon console:

Figure 43: AWS Access and Secret Keys

5. Under Select an AWS Data Source, specify the connection details of the AWS data source that you want to connect to:

   a. Select your AWS Region from the dropdown list.

   b. Click the Find My AWS Data Sources button.
      The AWS data source queries your environment and displays your available data sources, as shown in the figure below.

   c. Select your data source.

   d. Enter your username, password, and database name.
      The AWS data source queries your environment and adds the appropriate driver and URL.

Figure 44: Select an AWS Data Source Section

6. When you have entered all the information, click Test Connection.
    If your connection is successful, a message appears to the right of the button.
    Sometimes the process takes a few minutes. In that case you will see an alert. Try the
    test again after one or two minutes. The test performs the following actions:

    • Validates the username and password.

    • Creates a database security group.

    • Adds the IP address of your JasperReports Server instance to the security group
      to authorize ingress to the data service (RDS or Redshift).

    If you want to control details of the security group name or specify the IP
    address manually because you have a complex VPC Topology, see Configuring
    Cloud Services. You can also change the default JDBC driver through the
    configuration.

7. Click Save. The Save dialog appears.

8. Enter a name for the data source and an optional description. The Resource ID is
    generated from the name that you enter. If you have not already specified a location,
    expand the folder tree and select the location for your data source.

9. Click Save in the dialog. The data source appears in the repository.

Filtering the Regions For AWS Data Source

JasperReports Server automatically displays the regions associated with your
AWS credentials when creating an AWS data source. You can remove regions from the AWS
Region dropdown list by editing the .../WEB-INF/applicationContext.xml file.

For a list of the supported regions, see the AWS documentation:
https://docs.aws.amazon.com/general/latest/gr/rande.html

To filter AWS regions for your data source

Procedure

1. Open the .../WEB-INF/applicationContext.xml file for editing.

2. Locate the element <util:list id="filterAwsRegionList" list-class="java.util.ArrayList">.

3. Add the AWS regions that you want to remove from the dropdown list within the
   element. For example:

```
<util:list id="filterAwsRegionList" list-class="java.util.ArrayList">
    <value>us-gov-west-1.amazonaws.com</value>
    <value>us-gov-east-1.amazonaws.com</value>
</util:list>
```

4. Save the file.

5. Restart JasperReports Server.

Azure SQL Data Sources

Microsoft Azure provides data storage in the cloud as a service. Microsoft offers different
types of storage options for Azure users, but only the Azure SQL Server database service is
currently supported. JasperReports Server uses the Microsoft SQL Server JDBC driver
(`com.microsoft.sqlserver.jdbc.SQLServerDriver`), a management certificate, and your
credentials to create a secure connection between JasperReports Server and the Azure SQL
data source.

Uploading an Azure Certificate File to the Repository

Before you can create an Azure SQL data source, you will need a management certificate to authenticate JasperReports Server with Azure. The management certificate must be a x.509 certificate and can be either self-signed or signed by a trusted certificate authority. The certificate can use a public or private key. You will need to upload the management certificate (.cer) file to the Azure Management Portal and the key exchange (.pfx) file, which contains the server certificate and the key, to JasperReports Server. You can also store a server certificate (.cer) file in the repository.

To upload a certificate file to the repository

Procedure

1. Log into JasperReports Server as an administrator.

2. Click View > Repository and expand the folder tree.

3. Browse to the folder where you want to save the certificate.

4. Right-click the folder and select Add Resource > File > Azure Certificate from the context menu.

5. Click Choose File to locate and upload the certificate key exchange (.pfx) or server certificate (.cer) file.

6. Enter a name and resource ID for the file.

7. Click Submit to save the file to the repository.

Creating an Azure SQL Server Data Source

The data source wizard uses the Azure credentials that you provide to discover your Azure SQL Server databases. It then uses those credentials to properly configure access rules to maintain the connection between JasperReports Server and the data source. For information on configuring access rules for Azure, see Configuring Cloud Services.

To create an Azure SQL data source

Procedure

1. Log into JasperReports Server as an administrator.

2. Click View > Repository, expand the folder tree, and right-click a folder to select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select Azure SQL. The information on the page changes to reflect what's needed to define an Azure SQL data source.

Figure 45: Entering Azure User Information

4. Under Azure Settings, enter your Azure subscription ID, user certificate (.pfx) file, and user certificate password. Click Browse to select the certificate file from your repository. See Uploading an Azure Certificate File to the Repository for instructions on uploading the certificate file.

5. Under Select an Azure Database, specify the connection details of the Azure database that you want to connect to:

   a. Click the Find My Azure Databases button.
      JasperReports Server queries Azure and displays your available SQL Server databases.

b. Select your database.

c. Enter your server name, database name, user name, and database name. The Azure data source queries your environment and adds the appropriate URL.

d. If you have Microsoft's JDBC driver for SQL Server installed on your JasperReports Server instance, you can choose to use it instead of the existing JDBC driver by checking Use Microsoft Driver.

Figure 46: Selecting an Azure SQL Data Source

6. When you've entered all the information, click Test Connection.
   If your connection is successful, a message appears to the right of the button.
   Sometimes the process takes a few minutes. In that case you'll see an alert. Try the
   test again after one or two minutes. The test performs the following actions:

   • Validates the user name and password.

- Creates firewall access rules to authorize ingress to the data service.

- Adds the IP address of your JasperReports Server instance to the access rule.

  If you want to control details of the access rule name or specify the IP address manually, see Configuring Cloud Services.

7. Click Save. The Save dialog appears.

8. Enter a name for the data source and an optional description. The Resource ID is generated from the name you enter. If you haven't already specified a location, expand the folder tree and select the location for your data source.

9. Click Save in the dialog. The data source appears in the repository.

Snowflake Data Sources

During Snowflake data source creation, when creating a warehouse name, use the following characteristics:

- Start with a letter (A-Z, a-z) or an underscore ("_").

- The name should contain only letters, underscores, decimal digits (0-9), and dollar signs ("$").

> Special characters are not supported in warehouse names. For more information, see the Snowflake documentation.

Hadoop-Hive Data Sources

Unlike traditional databases, Hadoop supports huge amounts of data, often called big data. JasperReports Server processes requests to a Hadoop cluster using a JDBC data source with the Hive JDBC driver.

The JDBC driver for Hive works with most Hive 1, Hive 2, and Impala servers. However, the original Hive 1 server has high latency with access times on the order of 30 seconds and up to 2 minutes. Hive 2 is much faster, but still not as fast as relational databases. As a result, Hadoop-Hive data sources have certain limitations and guidelines for use in JasperReports Server:

- Hadoop-Hive data sources are not suitable for creating reports interactively in the Ad Hoc Editor.

- Reports based on Hadoop-Hive are not suitable for dashboards.

- Filters and query-based input controls that rely on Hadoop-Hive data sources will be slow to populate the list of choices.

- You must configure your query limits and timeout to handle latency (see Ad Hoc Data Policies for Big Data).

- You must configure your JVM memory to handle the expected amount of data (see the JasperReports Server Installation Guide).

In general, reports based on JDBC-Hive data sources are best suited to be run in the background from the repository. For very large reports, consider scheduling them to run at night so the output is available when you need it during the day.

To create a Hive JDBC data source, follow the same procedure as in JDBC Data Sources.

MongoDB Data Sources

MongoDB is a big data architecture based on the NoSQL model that is not relational or SQL-based. We provide connectors that allow reports to use a MongoDB data source or a MongoDB JDBC data source. Reports based on MongoDB data sources can be used as Topics that allow users to create Ad Hoc views based on the fields returned by the MongoDB query. JasperReports Server also supports Kerberos, SSL, and x509 authentication for MongoDB data sources.

As with all big data stores, MongoDB data sources have the following limitations and usage guidelines within JasperReports Server:

- MongoDB data sources are not supported for OLAP connections.

- MongoDB data sources cannot be used directly in Domains. To use MongoDB in a Domain, see MongoDB Data Sources.

- MongoDB data sources can be used in Ad Hoc Topics, but they do not support query optimization.

- You must configure your query limits to handle big data (see Ad Hoc Data Policies for Big Data).

- You must configure your JVM memory to handle the expected amount of data (see the JasperReports Server Installation Guide).

Creating a MongoDB Data Source with the Native MongoDB Driver

MongoDB data sources do not support virtual data sources when using the native MongoDB driver. For virtual data support, Create a MongoDB JDBC data source. See MongoDB Data Sources

To create a MongoDB data source with the native driver

Follow these steps to create a MongoDB data source with the native MongoDB driver.

1. Log in as an administrator.

2. Click View > Repository, expand the folder tree, and right-click a folder to select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you have installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select MongoDB. The information on the page changes to reflect what is needed to define a MongoDB data source.
   You have the option to use attributes in the values of data source parameters. See Attributes in Data Source Definitions.

Figure 47: MongoDB Data Source Page

4. Fill in the required fields, along with any optional information.
   The MongoDB URI has the form:

   mongodb://<hostname>:27017/<database>

   To enable SSL, include the argument ssl=true in the URI. For example:

   mongodb://<hostname>:27017/<database>?ssl=true

   To enable x509 authentication for the data source include it as well in the URI. For example:

   mongodb://<hostname>:27017/<database>?ssl=true&authMechanism=MONGODB-X509
   To enable Kerberos authentication for the data source include the data source and Kerberos username in the URI. For example:

   mongodb://<hostname>:27017/<database>?authMechanism=GSSAPI

   > Before you can enable x509 authentication, you need to set up the Java Secure Socket Extension (JSSE) in your application server. Before you can enable Kerberos authentication, you need to perform the steps in Using Kerberos Authentication with MongoDB Data Sources.

5. Click Test Connection to validate the data source.

6. When the test is successful, click Save. The Save dialog appears.

7. Enter a name for the data source and an optional description. The Resource ID is generated from the name that you enter. If you have not already specified a location, expand the folder tree and select the location for your data source.

8. Click Save in the dialog. The data source appears in the repository.

The Jaspersoft MongoDB Query Language

MongoDB is designed to be accessed through API calls in an application or a command shell. As a consequence, it does not have a defined query language. To write queries for MongoDB data sources, we have developed a query language based on the JSON-like objects on which MongoDB operates. JSON is the JavaScript Object Notation, a textual representation of data structures that is both human- and machine-readable.

The Jaspersoft MongoDB Query Language is a declarative language for specifying what data to retrieve from MongoDB. The connector converts this query into the appropriate API

calls and uses the MongoDB Java connector to query the MongoDB instance. The following examples give an overview of the Jaspersoft MongoDB Query Language, with the equivalent SQL terms in parentheses in the descriptions:

- Retrieve all documents (rows) in the given collection (table):

```
{ 'collectionName' : 'accounts' }
```

- From all documents in the given collection, select the named fields (columns) and sort the results:

```
{
  'collectionName' : 'accounts',
  'findFields' : {'name':1,'phone_office':1,'billing_address_city':1,
                  'billing_address_street':1,'billing_address_country':1},
  'sort' : {'billing_address_country':-1,'billing_address_city':1}
}
```

- Retrieve only the documents (rows) in the given collection (table) that match the query (where clause). In this case, the date is greater-than-or-equal to the input parameter, and the name matches a string (starts with N):

```
{
  'collectionName' : 'accounts',
  'findQuery' : {
    'status_date' : { '$gte' : $P{StartDate} },
    'name' : { '$regex' : '^N', '$options' : '' }
  }
}
```

The Jaspersoft MongoDB Query Language also supports advanced features of MongoDB such as map-reduce functions and aggregation that are beyond the scope of this document. For more information, see the language reference on the Community website.

Using Kerberos Authentication with MongoDB Data Sources

The native MongoDB driver can connect to MongoDB Enterprise using Kerberos Authentication. Kerberos is a network authentication protocol that allows clients and servers on a non-secure network to use "tickets" to identify themselves.

To use Kerberos authentication, you need to set up the following

Procedure

1. Install the Kerberos client tools on the JasperReports Server host. The client tools are included in the JDK/JRE for Windows. Client tools packages are available for Linux and OS X. You are prompted to enter information on your Kerberos system during the installation.

2. Download the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy files from Oracle and install them on the JasperReports Server. The files are available for download at the Oracle Java SE Downloads page.

3. Set these Java system properties for JasperReports Server:

```
java.security.krb5.realm=<Kerberos_realm>
java.security.krb5.kdc=<KDC_server_hostname>
javax.security.auth.useSubjectCredsOnly=false
java.security.auth.login.config=<path>/jaas.conf
```

4. Edit the jaas.conf file to add the following:

```
com.sun.security.jgss.krb5.initiate {
    com.sun.security.auth.module.Krb5LoginModule
    required
    useTicketCache=true
    doNotPrompt=true;
};
```

5. Run the kinit mongodb/[mongo-db-hostname]@<Kerberos_realm> command on the JasperReports Server host to create or renew a ticket for the Kerberos user. This is the account that you enter as the username for the data source.

Virtual Data Sources

Virtual data sources allow you to combine multiple JDBC and JNDI data sources and make them available to be joined through a Domain. You can combine any number of data sources, including schemas from different databases, databases from different vendors, and different schemas within the same database into a single object.

A virtual data source can wrap a single data source for big data, or combine any number of big data, JDBC, and JNDI data sources.

Once you have created a virtual data source, you create a Domain that joins tables across the data sources to define the relationships among the data. Ad Hoc views and reports based on the Domain can access the combined data transparently. For more information about Domains, see the JasperReports Server Data Management Using Domains.
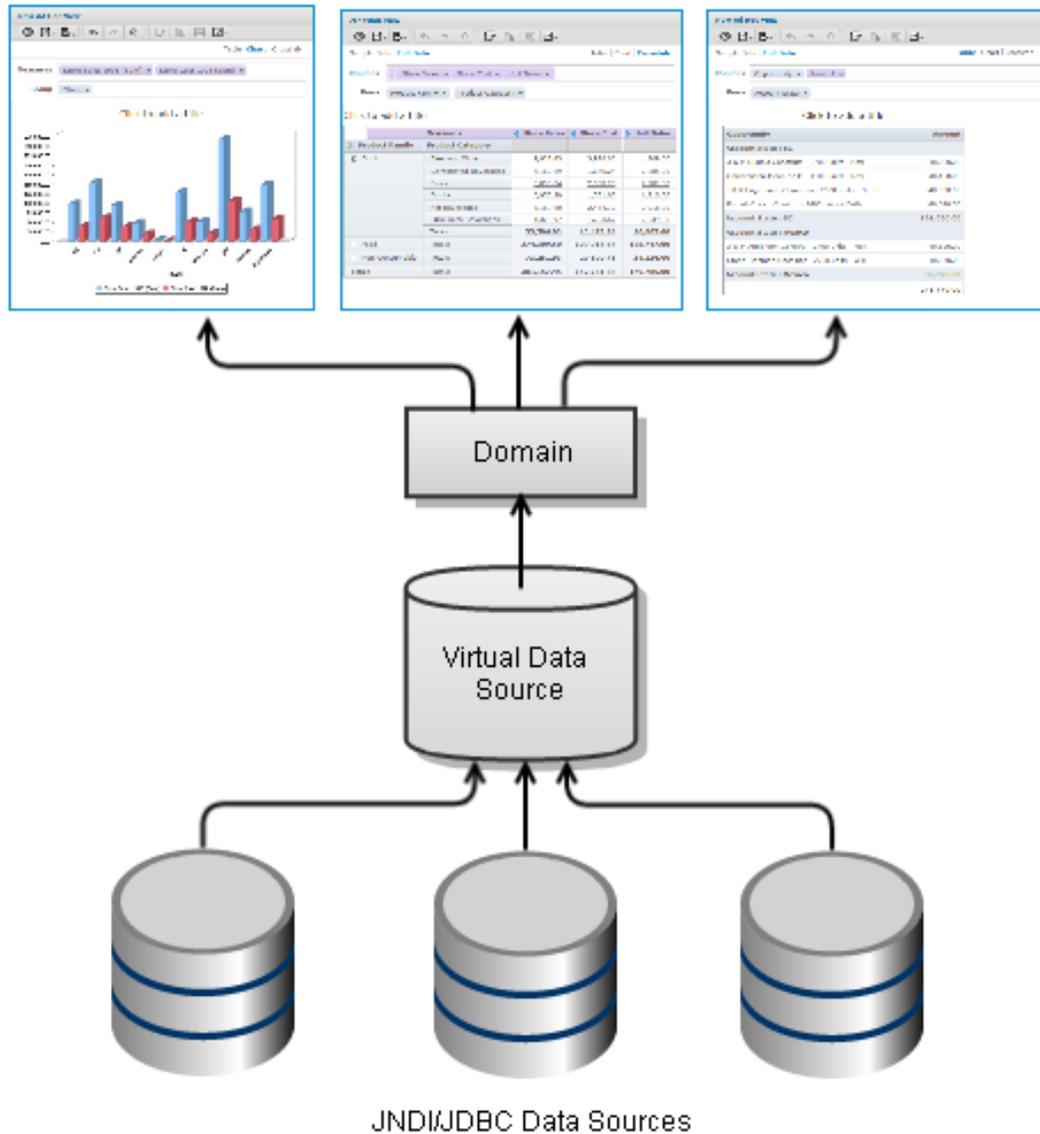
Figure 48: Virtual Data Source Scenario

When you combine data sources into a virtual data source, you select an alias for each data source you include. This alias is added as a prefix to the tables in the original data source to ensure that table names are unique across the virtual data source.

To create a virtual data source

Procedure

1. Log in as an administrator.

2. Click View > Repository, expand the folder tree, and right-click a folder to select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you have installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select Virtual Data Source.

4. Locate the data sources that you want to use in the Available Data Sources pane. Double-click to select each chosen data source. The data source is shown in the Selected Data Sources pane.

5. Change the aliases by editing them directly in the Alias column (optional). The alias identifies the selected data source within the virtual data source. It is also added as a prefix to the name of each table in that data source. Spaces are not allowed in aliases.

The following figure shows the values for creating a virtual data source by combining two of the databases included in the sample data: the Foodmart database and the SugarCRM database.
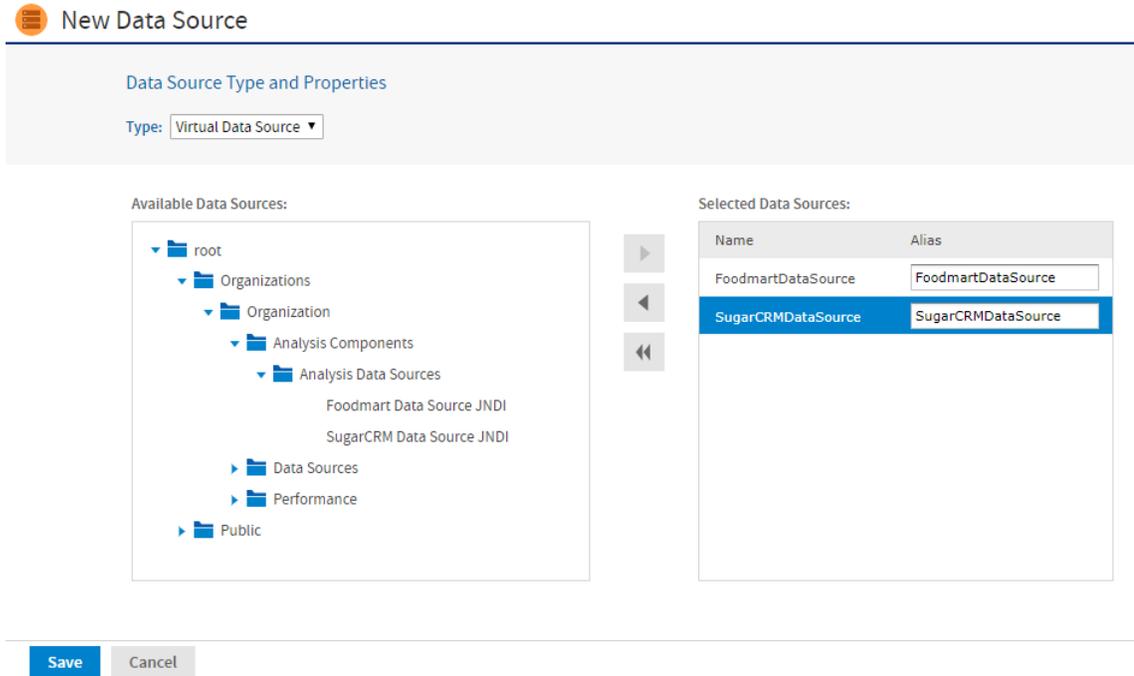
Figure 49: Creating a Virtual Data Source

> Virtual data sources cannot use the Time Zone field that may be set on individual data sources. If used in a virtual data source, a target data source with a time zone will not return the expected date-time values. Therefore, we recommend that you do not use data sources with time zone settings in a virtual data source.

6. Click Save. The Save dialog appears.

7. Enter a name for the data source and an optional description. The Resource ID is generated from the name that you enter. If you have not already specified a location, expand the folder tree and select the location for your data source.

8. Click Save in the dialog. The data source appears in the repository.

You can edit a virtual data source to add or remove the data sources it uses. If the virtual data source is used by a Domain, you can add data sources, but you cannot remove them. Removing a data source from a virtual data source modifies only the virtual data source. The data source that you removed remains in the repository.

To edit a virtual data source

Procedure

1. Log in as an administrator.

2. Click View > Repository and expand the folder tree to locate the folder containing the data source.

3. Right-click the data source and select Edit from the context menu.

4. To add a data source, locate the data source in the Available Data Sources pane and double-click. To remove a data source, select it in the Selected Data Sources pane and click the left arrow.

5. Click Save. If you try to delete a data source from a virtual data source that is used by a Domain, you will receive a warning and the data source remain unchanged. Otherwise, the data source is updated in the repository.

Performance Considerations

Virtual data sources are based on the Teiid engine to handle multiple data sources and combine the results from them. How you design the tables in each of your data sources and how you combine them in a Domain join can significantly affect performance.

Consider these issues when designing a solution that combines several databases in a virtual data source:

- When using a virtual data source in a Domain, you must select at least one of the tables from the Tables tab to appear in the Domain before using it to create a derived table. Virtual data sources are optimized to use the smallest set of metadata, but they do detect Domains that use only derived tables.

- The virtual data source runs in JasperReports Server and must allocate memory and use processing to handle result sets.

- The Teiid engine uses its own cache for data accessed through a virtual data source. You can clear the Teiid cache at the same time as the Ad Hoc cache, as described in Ad Hoc Cache Management.

- The virtual data source can push down certain operations (joins, filters, and aggregations) to the database, others it must perform in memory. The more it can push down, the smaller the datasets it handles in memory.

- You should design your schema to maximize the processing pushed to the database and minimize data handled in the virtual data source in a join between data sources.

For example, if you create a duplicate copy of a date dimension table in both of your data sources, they can be joined much more efficiently. If this table existed in only one database, the Teiid engine would need to retrieve all the rows from the other database and perform the join itself. With this table in both data bases, the Teiid engine can push down all of the time-dimension joins to the individual databases and perform a final join on much smaller datasets.

It might take a long time to get a list of virtual data source tables in a domain designer for the first time. It happens because the Teiid engine performs indexing on each table inside the schema, and it can be a long process for the databases having many schemas.

To improve the performance, do the following:

Procedure

1. Open the file .../WEB-INF/applicationContext-virtual-data-source.xml for editing.

2. Locate the bean named abstractTeiidVirtualQueryService and the following property.

```
<property name="customSelectedSchemas">
```

3. Add new entries to the map.

```
<entry key="Microsoft SQL Server">
    <map>
        <entry key="schemasToBeIncluded">
            <set value-type="java.lang.String">
                <value>[schema1]</value>
            </set>
        </entry>
    </map>
</entry>
```

This would look like the following.

```
<property name="customSelectedSchemas">
    <map>
        <entry key="oracle">
            <map>
                <entry key="usernameAsSchema">
                    <set value-type="java.lang.String">
                        <value></value> <!-- username will be used as schema here -->

                    </set>
                </entry>
                <!-- if only specific schema(s) needs to included for this DB then
                    uncomment the below entry and add the schemas to be included-->
                <!--<entry key="schemasToBeIncluded">
                    <set value-type="java.lang.String">
                        <value>[schema1]</value>
                        <value>[schema2]</value>
                    </set>
                </entry>-->
            </map>
        </entry>
        <entry key="Microsoft SQL Server">
            <map>
                <entry key="schemasToBeIncluded">
                    <set value-type="java.lang.String">
                        <value>[schema1]</value>
                    </set>
                </entry>
            </map>
        </entry>
    </map>
</property>
```

> The entry key should be JDBC type, and the set of values should contain only the schemas that are used for indexing the tables.

Another fix that you can do to improve the performance is increasing the time for cached schemas metadata to stay longer. Use the following procedure to increase the time for cached schemas metadata.

1. In the same bean, find the following line:

```
<property name="poolTimeoutInMinute" value="20"/>
```

2. Change the line as follows to increase the time for cached schemas metadata.

```
<property name="poolTimeoutInMinute" value="720"/>
```

For more information about optimizing your data for use in Teiid, see
http://www.jboss.org/teiid/.

Creating Cassandra Data Connectors

Virtual data sources can connect to Cassandra big data sources to make them available to a Domain. The virtual data source extracts the connection information from the Cassandra data source and uses an internal Teiid connector to access the data. The Teiid connectors map the various structures used in Cassandra's big data model to a relational model with tables and fields. This connector is distinct from what are called the native data sources for big data. For this reason, when a data source for big data is wrapped in a virtual data source, the resulting data source has the following limitations:

- The Cassandra connector in virtual data sources does not support query parameters ($P and $X). Therefore, if you use a big data connector wrapped in a virtual data source as the data source for a stand-alone query, report or Topic, you cannot include parameters to create input controls. When used in Domains and then Ad Hoc views, you can define filters to replace this functionality.

- The Cassandra connector for virtual data sources does not support any aggregation functions.

However, there are significant advantages to accessing big data through virtual data sources:

- When wrapped in a virtual data source, you can access Cassandra through a Domain, Domain Topic, Ad Hoc view, and Ad Hoc report.

- A virtual data source can contain any mix of JDBC, JNDI, and Cassandra data connectors. When you define a Domain using this data source, you can access the tables from each store and define joins between compatible fields.

- Virtual data sources that use a Cassandra data connector support query optimization, unlike the native data sources for big data. In fact, the big data connectors for virtual data sources support query optimization in Ad Hoc views and reports based on stand-alone Topics, and in Ad Ho views and reports based on Domains. The only exceptions are calculated fields, which cannot be optimized when used in Ad Hoc views or reports that are based on Topics or Domains. For more information about query optimization, see Ad Hoc Data Policies for Big Data.

> 📝 If you plan to use a Cassandra data source with Domains or Ad Hoc views, use the JDBC Cassandra data source driver. You do not need to create a virtual data source as was required in some older JasperReports Server versions.

To create a virtual data source that accesses a Cassandra data source

Procedure

1. Create a Cassandra data source, or verify that it was created as described in JDBC Data Sources.

2. Create a virtual data source as described in Virtual Data Sources.

3. In the virtual data source creation dialog, select the Cassandra data source that you created in the first step, and save the virtual data source. You can select one or more Cassandra data sources, or any mix of Cassandra, JDBC, and JNDI data sources.

4. Create a Domain, specify the virtual data source you just created, and then select the Cassandra data tables when you create the Domain schema. The data from the data source is mapped to tables and fields in the Domain that you can use to create joins, filters, and all other features of a Domain.

Logging for Virtual Data Sources

If you have issues with your big data connections through virtual data sources, you can enable logging in the following classes:

```
com.jaspersoft.jasperserver.api.common.virtualdatasourcequery.VirtualDat
aSourceQueryService
```

```
com.jaspersoft.jasperserver.api.engine.common.virtualdatasourcequery.tei
id.TeiidEmbeddedServer
```

For information about enabling logging, see Configuring System Logs.

TIBCO Data Virtualization

A TIBCO Data Virtualization (TDV) virtual data source integrates multiple data sources into a single virtual data layer. The TDV enteprise solution provides tools that support all phases of data virtualization development, run-time, and management. With these tools, you can build a virtualized view for your data sources that acts like a single database and provide JasperReports Server access to the data through a JDBC driver. This allows JasperReports Server to query and retrieve data from the virtual database like it would for a normal database.

Documentation for TDV is available at https://docs.tibco.com/products/tibco-data-virtualization.

To create a TIBCO Data Virtualization data source

Procedure

1. Log in as an administrator.

2. Select View > Repository, right-click a folder's name, and select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select JDBC and then select Tibco Data Virtualization from the JDBC driver list. The page refreshes to show the fields required for the data source.

Figure 50: TIBCO Data Virtualization Driver

4. Enter the host machine and port number for the data source. The default hostname is localhost, and the default port is 9401.

5. Enter the name of the domain (composite or LDAP) to which the database user belongs.

6. Enter the name of the published TDV virtual data source. Please note that JasperReports Server only works with published data sources. Do not enter the name of the folder that stores your TDV data resources.
The four fields are combined automatically to create the URL where the server

accesses the database.

7. Fill in the database username and password. These are the credentials that the server uses to access the database.

8. Click Test Connection to validate the data source. If the validation fails, ensure that the values you entered are correct and that the database is running.

9. When the test is successful, click Save. The Save dialog appears.

10. Enter a name for the data source and an optional description. The Resource ID is generated from the name that you enter. If you have not already specified a location, expand the folder tree and select the location for your data source.

11. Click Save in the dialog. The data source appears in the repository.

XLS and XLSX Data Sources

XLS and XLSX data sources read a file in the Microsoft Excel XLS or XLSX format and allow you to query its contents as a relational table. These data sources support Domain-related features like:

- Returning JDBC metadata

- Creating derived tables

- Supporting calculated fields

- Domain pre-filtering

- Joining other data sources in a virtual data source

- Full support in the Domain Designer

To create an XLS or XLSX data source

Procedure

1. Log in as an administrator.

2. Click View > Repository, expand the folder tree, and right-click a folder to select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data

Source page appears.

3. In the Type field, select either the XLS or XLSX Data Source. In this example, we create an XLSX data source.

Figure 51: File Data Source Page for an XLS/XLXS Data Source

4. Enter the date and time format patterns used for the data source.

5. Enter the file path for the XLSX file. You can specify a file in the repository with the

repo: syntax, followed by the repository path of the file. Specify ftp:, http: or https: to use those Internet protocols. To specify a file on your server's file system, specify its path directly, starting from the root, for example /tmp/MyDataFile.xlsx. The user running the server process must have permission to access the file.

> Only the **File Path field** is required, and the remaining fields are optional.

6. Enter the regional locale for the file. The locale property displays the data in a specific format based on the region. (optional)

7. Enter the number pattern for the file. (optional)

8. Enter the name of the Excel file sheet to use for the data source. (optional)

9. Enter the local time zone for the file. (optional)

10. Specify whether the first row in the XLSX file is a header as true or false. (optional)

11. Click Save. The Save dialog appears.

12. Enter a name for the data source and an optional description. The Resource ID is generated from the name that you enter. If you have not already specified a location, expand the folder tree and select the location for your data source.

13. Click Save in the dialog. The data source appears in the repository.

File Data Sources

File-based data sources read a text file in JSON, XML, or XMLA format and allow you to query its contents as a relational table. A text file in JSON format may be stored in the server's repository, in the server's file system. While a text file stored in JSON, XML, or XMLA format can be accessed from a URL. This means you can create a data source based on data retrieved from websites and even REST APIs.

Domains support file-based data sources, such as text files in JSON and XML format as well as Excel files. These file-based data sources support features domain-related features like returning JDBC metadata, creating derived tables, supporting calculated fields, domain pre-filtering, joining other data sources in a virtual data source, and full support in the Domain Designer.

Older file-based data sources, which presented domain data in a single flat table and lack the features of the newer data sources, appear in the Type dropdown menu with (Before 6.4) next to their names.

By default file data sources do not appear in the New Data Source dialog and must be enabled first.

The following file-based data sources are available:

- Remote XML data source (Simple single-table support and full domain support)
- JSON data source (Simple single-table support and full domain support)
- JSON Series data source (Simple single-table support)
- JSONQL (JSON Query Language) data source (Simple single-table support and full domain support)
- Text (CSV) data source (Full domain support)
- XMLA Query data source (Simple single-table support)

Note that if there are two versions of a file-based data source type in the .../WEB-INF/applicationContext-remote-services.xml file, the version with full domain support has a 2 at the end of its name.

To enable file data sources in the UI

Procedure

1. Open the file .../WEB-INF/applicationContext-remote-services.xml for editing.
2. Locate the element <util:set id="customDataSourcesToHide">.
3. Comment out the file data sources that you want to display.
4. Save the file and restart JasperReports Server.

In the following example, the JSON with simple domain and full domain support is commented out so it appears in the Type dropdown menu:

```
<util:set id="customDataSourcesToHide">
    <!-- value>remoteXmlDataSource2</value>
    <!-- value>remoteXmlDataSource</value>
    <value>mongoDBQueryDataSource2</value>
    <value>mongoDBQueryDataSource</value>
    <!-- value>jsonDataSource2</value>
    <!-- value>jsonDataSource</value>
    <value>jdbcQueryDataSource2</value>
    <value>jdbcQueryDataSource</value>
    <!-- value>xlsDataSource</value>
    <!-- value>xlsxDataSource</value>
    <value>textDataSource</value>
    <value>JsonSeriesDataSource</value>
    <value>xmlaQueryDataSource</value>
</util:set>
```

After file data sources are enabled, you can create a file data source.

> The headers in data source text files must not be blank and cannot contain spaces or special characters other than underscores.
>
> To upload a JSON file to the JasperReports Server repository, you must save it with the .xml extension on your computer. Then create an XML file resource and select your JSON.xml file. Give your resource a name such as MyDataFile_JSON so you can find it. The repository calls it an XML file, but the data source can interpret the JSON contents.

To create a file data source

Procedure

1. Log in as an administrator.

2. Click View > Repository, expand the folder tree, and right-click a folder to select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select a file data source that you enabled, for example JSON Data Source or Remote XML Data Source. In this example, we create a JSON data source. The fields on the page change to reflect what is needed to define a file data source. You have the option to use attributes in the values of data source parameters. See Attributes in Data Source Definitions.

Figure 52: File Data Source Page for a JSON Data Source

4. Enter the URI of the JSON file. You can specify a file in the repository with the repo: syntax, followed by the repository path of the file. Specify ftp:, http: or https: to use those Internet protocols. To specify a file on your server's file system, specify its path directly, starting from the root, for example /tmp/MyDataFile.json. The user running the server process must have permission to access the file.

5. You can specify an optional query to limit or select the data accessible through the data source. The query has the following syntax:

   <tableName> or <tableName>/<fieldName>

   where:

   <tableName> corresponds to an XML element or JSON structure name

   <fieldName> corresponds to an XML sub-element or JSON field name

6. Click Save. The Save dialog appears.

7. Enter a name for the data source and an optional description. The Resource ID is generated from the name that you enter. If you have not already specified a location, expand the folder tree and select the location for your data source.

8. Click Save in the dialog. The data source appears in the repository.

Once you have defined a file data source:

- You can use SQL queries in reports to access the data as a relational table.

- You can create a Domain on a file data source, allowing you to alter the visibility and names of the fields extracted from the file. On the Display tab of the Domain Designer, you can also specify which fields are measured.

- You can create Ad Hoc views using the Domain based on your file data source, allowing you to explore and interact with data from the file.

- You can create virtual data sources that combine several file data sources, even different file formats such as XML and JSON, as long as their data structure is compatible so the tables can be joined. You can then create a Domain based on the virtual data source to join the tables and access the joined data in Ad Hoc views and reports.

Enabling JasperReports Server XMLA Server

The XMLA server component is disabled by default. JasperReports Server can still receive and process data from a Mondrian XMLA data source as a client, but that XMLA connection can no longer be used as a server to define a second data source. If you have reports based on such a datasource, they must be modified to use the Mondrian XMLA data source directly.

It is strongly encouraged to keep the JasperReports Server XMLA Server disabled for security considerations. However, the XMLA Server can be re-enabled through the following steps:

Procedure

1. Edit **web.xml**.

2. Find *JasperXmlaServlet3*.

3. Uncomment the servlet definition and mapping to:

```
<servlet>
<servlet-name>JasperXmlaServlet</servlet-name>
<servlet-class>com.jaspersoft.jasperserver.war.xmla.XmlaServletImpl</servlet-class>
<load-on-startup>50</load-on-startup>
</servlet>
<servlet-mapping>
<servlet-name>JasperXmlaServlet</servlet-name>
```

```
<url-pattern>/xmla</url-pattern>
</servlet-mapping>
```

4. Restart the server.

Bean Data Sources

The bean data source type is a key extension because it allows you to make use of any custom or exotic data that you might need to report on. Bean data sources serve as a bridge between a Spring-defined bean and a JasperReport. The Spring bean is responsible for providing the data or parameters that fill the report.

To use a bean data source, you must first configure the underlying Spring bean and make it available in the server's web application context. For example, you would add a bean definition to one of the .../WEB-INF/applicationContext*.xml files.

The bean must resolve to a ReportDataSourceService instance, either directly or by way of a factory no-argument method. You can use any Spring instantiation method (for example, a constructor or factory) and bean scope (for example, singleton or prototype) for the data source service bean.

The ReportDataSourceService instance is responsible for supplying data source parameters to the JasperReport. Custom ReportDataSourceService implementations can follow two approaches:

-   If the implementation can provide the data to be used to fill a report, it needs to wrap the data into a suitable JRDataSource implementation and pass the data using the REPORT_DATA_SOURCE report parameter.

-   If the data comes from the report query by way of a JasperReports query executor, the data source service must set values for the connection parameters defined by the query executor. The connection parameters are usually obtained from the properties of the data source service instance.

For example, you could implement a Hibernate data source service that would be injected in a session factory. The factory would create a Hibernate session that would be passed as a value for the HIBERNATE_SESSION parameter. The JasperReports Hibernate query executor then uses the parameter to run the HQL report query.

The ReportDataSourceService interface contains two methods: setReportParameterValues and closeConnection. The former provides data and connection parameter values; the

latter is required to close and release any resources or connections created during the call to setReportParameterValues.

As of JasperReports Server 6.0, there have been some changes to bean data sources that might affect your custom code. If your custom data sources generate errors after upgrading, see the troubleshooting section Upgrading Bean Data Sources.

Once the data source service bean is available through Spring, you can add the bean data source to the repository.

To create a bean data source

Procedure

1. Log on as an administrator.

2. Click View > Repository, expand the folder tree, and right-click a folder to select Add Resource > Data Source from the context menu. Alternatively, you can select Create > Data Source from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears, as shown in the figure below.

3. In the Type field, select Bean. The information on the page changes to reflect what's needed to define a bean data source.

4. Enter the bean name. If the data source service is to be instantiated through a factory method of the Spring bean, you should also enter the name of the method.

Figure 53: Bean Data Source Page

5. Click Test Connection to validate the data source. If the validation fails, ensure that the values you entered are correct and that the bean is in the classpath.

6. When the test is successful, click Save. The Save dialog appears.

7. Enter a name for the data source and an optional description. The Resource ID is generated from the name you enter. If you haven't already specified a location, expand the folder tree and select the location for your data source.

8. Click Save in the dialog. The data source appears in the repository.

Other Resources in the Repository

The previous chapter introduced data sources necessary to create any report. But the repository also stores other resources needed by reports. This chapter goes into detail about how to create queries, input controls, and file resources. These are the resources that users reference when uploading a JRXML file to create a JasperReport.

As an administrator, you need to define the resources required for your users' reports. If you have users who are proficient at creating their own reports in Jaspersoft Studio they can upload them as JasperReports to the server. In this case, you'll work with them to prepare the resources required by their reports. For less proficient users, you'll have to create and upload their JasperReports to the server.

For instructions on creating and uploading JasperReports to the server, see the JasperReports Server User Guide.

This chapter contains the following sections:

- Queries
- Datatypes
- Lists of Values
- Input Controls
- Query-based Input Controls
- Cascading Input Controls
- File Resources

Queries

A JasperReport uses a query to select the data to be returned from the data source. You can define the query in the JRXML itself, but you can also create and save queries in the repository. A query saved in the repository can be re-used by multiple JasperReports.

Reusing a query enables you to adapt reports to different audiences. The query returns the same data from the same data source every time, but each report presents the data in a

different way. Also, separating the query from the JRXML makes it easier to maintain large numbers of reports when the data source changes and the query needs to be updated.

You can also use queries to populate list input controls, as described in Query-based Input Controls and Cascading Input Controls.

To create a reusable query

Procedure

1. Login as an administrator.

2. Click View > Repository and locate the folder for the query.

3. Right click the folder's name and select Add Resource > Query from the context menu. The Add Query page appears.

Figure 54: Add Query - Name the Query Page

4. Enter a name for the query. Resource ID is filled in automatically, and the description is optional. Click Next. The Link a Data Source page appears.

Add Query: My Query

Link a Data Source to the Query

Optionally, link a data source to the query. You can choose a data source from the repository or create a new one.

Name the Query

- **Link a Data Source**

Define the Query

◉ Do not link a data source

◯ Click here to create a new data source

◯ Select data source from repository

[   ]

Browse...

Previous   Next   Cancel

Figure 55: Add Query - Link a Data Source Page

5. Select the data source and click Next. Your options are:

    - Do not link a data source. If no data source is associated with the query, the server uses the data source associated with the report that references the query.

    - Create a new data source. You can define a local data source within this query resource that's not accessible to any other resource. Click the link to create any data source as described in Data Sources. This new data source overrides any data source specified in reports that use the query.

    - Select data source from repository. This creates a reference to a data source in the repository. Click Browse to select an existing data source. The data source you select overrides any data source specified in reports that use the query.

    Click Next. The Define the Query page appears.

Add Query: My Query

Define the Query

Select a language and enter the query

Name the Query

Link a Data Source

● **Define the Query**

Query Language: SQL

Query String:

SELECT * FROM orders

Previous   Save   Cancel

Figure 56: Add Query - Define the Query Page

6. Select a Query Language. For this example choose SQL.
   The query language Domain (sl) is selected when opening Domain-based queries created in versions of the server before 3.7. It is used only for backward compatibility and should not be selected for new Domain-based queries.

7. Enter the text of your query in the Query String field, for example:

```
SELECT * FROM orders
```

8. Click Save.

By default, JasperReports Server supports SQL, HQL (Hibernate), HiveQL (Hadoop-Hive), MongoDB, CQL (Cassandra), JasperQL, and Domain queries, while JasperReports Library supports several more (such as EJBQL, xPath and MDX). However, JasperReports Server can support additional query languages if a query executor implementation is properly configured for each additional language when the server is deployed.

You can use a specialized bean data source to support multiple query languages. For information about bean data sources, refer to Bean Data Sources. Another option to add new types of data sources to the server to extend the reach of the JasperReports Server platform. Custom data sources are described in the JasperReports Server Ultimate Guide.

Datatypes

A datatype defines the format of a single-value input control, for example text or numerical. Datatypes determine what users can enter in the input fields that correspond to parameters in a report:

- Text

- Number

- Date

- Date/time

To create a datatype

Procedure

1. Log on as an administrator.

2. Click View > Repositoryand locate the folder for the datatype.

3. Right click the folder's name's name and select Add Resource > Datatype from the context menu. The Add Datatype page appears.

Figure 57: Add Datatype Page

4. Enter a name and optional description for the datatype. The resource ID is filled in

automatically.

5. Select the datatype and provide related information. Your options are:

    *   Text – You can specify a regular expression in the Pattern field. The expression is used to validate the text the user submits. For instance, you could enter an expression that tests for email addresses.

    *   Number – With numerical datatypes, you can control the range of acceptable values by specifying minimum and maximum values and whether the specified values are themselves acceptable (Minimum is Strict and Maximum is Strict check boxes). If you select a Strict check box, the specified value is not acceptable.
        For instance, for a percent field, you might specify a minimum of 0 and a maximum of 100. If you do not want to accept 0 percent, you would check Minimum is Strict. If you want to accept 100 percent, you would clear Maximum is Strict.

    *   Date and DateTime – Click the calendar icon to the right of the Minimum and Maximum date or date time fields to choose their values.

6. Click Save. The datatype resource appears in the repository.

Lists of Values

A lists of values resource is a static list of values for single-select or multi-select input controls. Depending on the type of input control, the user selects one or more of these labels using radio buttons, check boxes, or drop-down lists.

To create a list of values

Procedure

1. Log on as an administrator.

2. Click View > Repository and locate the folder for the new list of values.

3. Right click the folder's name and select Add Resource > List of Values from the context menu. The Add List of Values page appears.

Add List of Values

Add List of Values

Identify the list, then create the name-value pairs.

Name (required):

List of Plan Years

Resource ID (required):

List_of_Plan_Years

Description:

Name	Value	
Year One	2000	Remove
Year Two	2001	Remove
Year Three	2002	Remove
Year Four	2003	Add

Submit   Cancel

Figure 58: Add List of Values Page

4. Enter a name for the list of values. Resource ID is filled in automatically, and the description is optional

5. Enter the name and value for each item in the list and click Add.
   The name and value are both treated as strings. Users see the label only in an input control that uses the list, and the report receives only the value. To remove an item, click Remove beside the value.

6. Click Submit. The list of values resource appears in the repository.

Input Controls

Any JasperReport can be parameterized so that its generated output is a function of values given at run time (query filters), or so that its layout is changed to accommodate different users (such as changing the title).

When writing JRXML, you can declare parameters and accommodate any run time value that needs to be passed into the query executor, the rendering engine, or the calculation engine. However, the parameter information in a JRXML file does not provide everything JasperReports Server needs to build a complete user interface and prompt users for values. You must also define an input control resource that defines the following:

- The range of possible values or list of discrete values allowed.

- The type of input, for example single-select or multi-select, and the widget to display the possible values, for example drop-down list or check boxes.

- Display options such as labels and whether the value is required.

- The name of the corresponding parameter in the JRXML.

When a user runs the report, the server uses this information to prompt the user for a value and to validate that input. For example, consider a report that returns sales data for all of a company's products; the user might input a product to view by selecting a product name in a list.

JasperReports Server supports several types of input controls, each of which can map to certain types of parameters in the report's JRXML. The input control also determines the kind of widget the user interacts with:

- Boolean – Presented as a check box. These input controls return a java.lang.Boolean object to the report engine in response to the user's selection. Boolean input controls return .TRUE or FALSE as values, depending on whether the box is checked.

- Single value – Presented as a free-form text box. You must specify a datatype, for example text or numerical value, and the user's entry is validated against this datatype.

- Single-select – Presented either as a drop-down list or a set of radio buttons. A single-select input control returns a single value.

- Multi-select – Presented as a scrollable list of values or a set of check boxes. A multi-select input control returns a collection of values.

One advanced feature of single-select or multi-select input controls is that the values they present can be the result of a dynamic query. The query retrieves actual values from the data source before presenting them as choices to the user. These queries can contain parameters themselves, for example based on the logged-in user or the selection of a previous input control. Query parameters are described in Query-based Input Controls and Cascading Input Controls.

Input controls rely on other resources in the repository, such as datatypes, static lists of values, or queries. You can manage these resources the same way you manage other resources. You can define them locally (available only to the input control) or reference them externally (reusing a resource in the repository). For more information, see Local Resources and External References.

> Ad Hoc views based on Domains and Domain Topics always use locally defined input controls that are created automatically based on the chosen filters. They cannot refer to input controls stored in the repository, and you should not modify them. For more information, see the JasperReports Server User Guide.
>
> Some input controls rely on queries to populate their options. These more complex controls are described in Query-based Input Controls.

As with other resources, input controls can be created locally as part of a JasperReport, in which case they cannot be seen outside that JasperReport, or they can be created separately in the repository and referenced in multiple reports.

To use an input control in a report, the control must meet two conditions:

- The parameter name in the input control must correspond to the name of the parameter in the report. No error occurs for a mismatch, but at run time NULL is passed instead of actual value of the parameter.

- The input control and its corresponding parameter must be of compatible datatypes (for example, both must be text types or date types). If there is a mismatch, the report fails and an exception is returned.

This section explains how to create an input control in the repository. To reference input controls in a JasperReport, see the JasperReports Server User Guide.

To create an input control

Procedure
1. Log on as an administrator.

2. Click View > Repository and locate the folder for the input control.

3. Right click the folder's name and select Add Resource > Input Control from the context menu. The Add Input Control page appears.

Figure 59: Add Input Control Page

4. Select the type of input control from the Type list. In this example, Single Value is selected.

5. Enter the prompt to tell users how to use the control. This example, uses the prompt Select the text for the report title.

6. In practice, the prompt text is often the same as the parameter, so the parameter name is automatically filled in. If you have used a different prompt, edit the Parameter Name field and enter the exact name of the parameter for your control. Remember, the parameter name must be the same here as in the reports that use this input control.
   For this example, the parameter name is title. Description is optional.

7. Select options for the control. Your options are:

   - Mandatory – Forces the end user to supply a value.

   - Read-only – Displays the value of the parameter without allowing the end user to modify it.

   - Visible – Makes the input control visible in the report options dialog.

8. Click Next.
   Subsequent pages depend on what type of input control you chose:

   - Boolean types do not require any further information.

   - Single-value types require a datatype the user can enter.

- Single-select and multi-select types based on static lists require a list of values.

- Single-select and multi-select types based on queries require a query.

9. In this single-value example, the Locate Datatype page appears. Choose the option to select a datatype from the repository and click Browse. In the repository dialog, select /datatypes/TextGeneralDatatype, which is similar to the datatype we created in Datatypes.

> If you choose to define a datatype, the wizard takes you through the same procedure as in section Datatypes. You can then define any datatype you need, but it's local to the input control and not reusable in other input controls.

Add Datatype

Locate Datatypes

○ Define a Datatype in the next step

● Select a Datatype from the repository

/datatypes/My_Datatype

Browse...

Previous   Next   Cancel

Figure 60: Locate a Datatype for an Input Control

10. Click Next. The input control resource is created in the repository.

11. Locate the input control in the repository manager. Notice that the text of the prompt that you entered in Enter the prompt to tell users how to use the control. This example, uses the prompt Select the text for the report title. is also used as the name for the resource.

Query-based Input Controls

Query-based input controls display a dynamic set of values for the user to choose from. They are input control resources in the repository, but instead of being based on a datatype or a static list of values, they perform a query to retrieve a list of values. For

example, a report could have a city parameter, and the query-based input control could display the list of cities in your data. Because the queries use standard syntax, you can include filters in a WHERE clause, for example, to restrict the list of cities to a certain country.

Creating a Query-based Input Control

In this first example, we create a query-based input control that returns a list of all cities the user can choose from.

Procedure

1. Log in as an administrator.

2. Browse the repository and select the folder for the query-based input control.

3. Right click the folder's name and select Add Resource > Input Control. The Add Input Control dialog appears.

Figure 61: Adding an Input Control - Naming

4. Select the type of query-based input control from the type drop-down list. This determines how the input control appears to users, either as a drop-down list, a set of radio buttons, a multi-select list, or a set of check boxes. In this example, we

choose a single-select query-based input control type.

5. Specify the prompt text, parameter name, optional description, and appearance options in the same manner as when defining a regular input control.

6. Click Next. Because we selected one of the query-based types, the Locate Query page appears:

Figure 62: Adding an Input Control - Locating the Query

If you have a suitable query defined in the repository, you can select it here as an external reference. In this example, we'll define a query locally inside the input control.

7. Click Next to define the local query resource. The query naming dialog appears:

Add Query:

Name the Query

Enter the required property values.

• **Name the Query**

    Link a Data Source

    Define the Query

Name (required):

MyInputControlQuery

Resource ID (required):

MyInputControlQuery

Description:

[Previous]   [Next]   [Cancel]

Figure 63: Adding an Input Control - Naming the Query

Although the query resource is not visible in the repository, it may still have a name, ID and optional description within the query resource. However, the values for these fields are not important.

8. Enter any name, and the ID is filled in automatically. Then click Next. The link data source page appears:

Add Query: MyInputControlQuery

Link a Data Source to the Query

Optionally, link a data source to the query. You can choose a data source from the repository or create a new one.

    Name the Query

• **Link a Data Source**

    Define the Query

◉ Do not link a data source

◯ Click here to create a new data source

◯ Select data source from repository

[Browse...]

[Previous]   [Next]   [Cancel]

Figure 64: Adding an Input Control - Linking to a Data Source

Like all queries, the query inside the input control may optionally link to a data source, either in the repository or its own internally defined one. If no

data source is linked, the query in the input control uses the same data source as the report. In this example, we use the default of not linking to a data source.

9. Click Next. The query definition page appears:

Add Query: MyInputControlQuery

Define the Query

Select a language and enter the query

Name the Query

Link a Data Source

• **Define the Query**

Query Language: SQL

Query String:

SELECT country, state, city FROM accounts WHERE country = "USA" ORDER BY state, city

Previous   Save   Cancel

Figure 65: Adding an Input Control - Defining the Query

10. Select the query language, in this example SQL, and enter a query string. The SELECT statement should contain the names of all fields used in the display, value, or filter for the input control. In this example, the query returns three fields, country, state, and city. Country limits the values to a single country. The ORDER BY clause ensures that the values from the query are sorted alphabetically when they appear in the input control.
For an example in a different query language, see Domain-based Queries.

11. Click Save to complete the query definition. The parameter values page appears:

Add Input Control

Set Parameter Values

Provide parameters for the value column and any other columns you want to appear in the input control.

Value Column

| city | (required) |

Visible Columns

city	Remove
state	Remove
	Add

Previous   **Submit**   Cancel

Figure 66: Adding an Input Control - Setting Parameter Values

On the parameter values page, you specify which fields in the query result are displayed, and which field contains values that become the parameter value. When chosen.

a. First, specify the value column. This is the field whose value is passed to the report. The datatype of the field must match the type of the corresponding parameter in the report.

b. Next, specify the visible columns. These are the fields whose values appear in the input control the user chooses from. In the simplest case, enter the same field as the value column. If you add multiple fields to the visible columns, the input control displays the fields together, in the order listed, separated by a vertical bar (|). In this example, the user can see and choose from:

Los Angeles | CA

San Francisco | CA

Denver | CO

Only the city value (without the state) is passed to the report. Showing additional fields in this way can help users find the value they want in long lists of results.

The value and display columns may also be entirely different, for example, displaying the full name of a sales representative, but using the employee ID as

the value returned by the input control. The only restriction is that all fields used in the value or display list must be selected by the query.

Built-in Parameters for Query-based Input Controls

The LoggedInUser and LoggedInUsername parameters are always available for query input controls; they're always available to reports, as well, even if an input control isn't defined for them. The standard parameters are also provided for reports if they're defined as parameters in the JRXML.

Built-in Parameters for Query-based Input Controls

Parameter Name	Type	Notes
LoggedInUser	User	The currently logged in user. This parameter isn't available in query input controls, but is used as a parameter to the report.
LoggedInUsername	String	The user name of the current user.
LoggedInUserFullName	String	The full name of the current user.
LoggedInUserEmail Address	String	The email address of the current user.
LoggedInUserEnabled	Boolean	Indicates whether the current user is enabled.
LoggedInUserExternally Defined	Boolean	Indicates whether the current user is authenticated externally.
LoggedInUserTenantId	String	The current user's organization.
LoggedInUserRoles	Collection <String>	The roles assigned to the current user. This is helpful for parameters that use $X.

Attributes defined on users, organizations, or at the server-level can also be used in reports and query-based input controls. For more information, see Managing Attributes.

Attribute-based Parameters for Queries and Reports

Parameter Name	Type	Notes
LoggedInUserAttributes	Map<String, String>	The attributes of the logged-in user. This parameter isn't usable in query input controls, but it is used as a parameter to the report. If the user has no attributes, the parameter is an empty map.
LoggedInUserAttribute Names	Collection <String>	The names of the attributes of the logged-in user. This is helpful for parameters that use $X. If the user has no attributes, the parameter is an empty map.
LoggedInUserAttribute Values	Collection <String>	The values of the attributes of the logged-in user. This is helpful for parameters that use $X. If the user has no attributes, the parameter is an empty map.
LoggedInUserAttribute_ <attribute-name>	String	For the logged-in user, the value of the attribute matching the name passed as <attribute-name> (such as att1). If there's no attribute with this name for this user, the parameter is empty. This parameter is available only if it's defined in a query or as a report parameter.
LoggedInTenantAttribute_ <attribute-name>	String	The value of the named attribute defined in the organization of the logged-in user. If there's no attribute with this name for this organization, the parameter is empty. This parameter is available only if it's defined in a query or as a report parameter.
ServerAttribute_ <attribute-name>	String	The value of the named attribute defined at the server level. If there's no attribute with this name at the server-level, the parameter is empty.

Parameter Name	Type	Notes
		This parameter is available only if it's defined in a query or as a report parameter.
Attribute_ <attribute-name>	String	The value of the named attribute is determined hierarchically. The first value found for the named attribute in the following order: • On the logged-in user. • On the organization of the logged-in user, then in any parent organizations. • At the server level. If there's no attribute with this name defined in those locations, the parameter is empty. This parameter is available only if it's defined in a query or as a report parameter.

Domain-based Queries

For reports that use a Domain as the data source, a query-based input control must contain a query against the Domain. When defining the query as shown in Adding an Input Control - Defining the Query, set the query language to Domain.

> 📝 The query language Domain ("sl") is selected when opening Domain-based queries created in JasperServer 3.5 or earlier. It's used only for backward compatibility and should not be selected for new Domain-based queries.

Domain queries have their own special syntax, the same one that's used in the Domain design. A Domain-based query references fields, called items, by their item IDs, along with any set IDs that determine the path of the item within the Domain. For example, if you want your query input control to return a list of store cities, where the field with ID ej_store_store_city is nested in the set with ID expense_join_store, you would use the following Domain query:

```
<query>
  <queryFields>
    <queryField id="expense_join_store.ej_store_store_city" />
  </queryFields>
</query>
```

The list contained inside the <queryFields> tag in a Domain query is equivalent to the fields given in the SELECT statement of an SQL query. Given the query above, you can create an input control for a Domain-based report that lets the user select a city as a parameter to the report.

Sometimes, you may want the input control to display more information than the actual value returned. As with standard query-based input controls, you can select more fields, and then display those fields in your input control. For example, to make the list of cities unambiguous, you could include the state and country in your display. In that case, the Domain-based query must also retrieve those items:

```
<query>
  <queryFields>
    <queryField id="expense_join_store.ej_store_store_city" />
    <queryField id="expense_join_store.ej_store_store_state" />
    <queryField id="expense_join_store.ej_store_store_country" />
  </queryFields>
</query>
```

Then, when specifying your visible query columns, as shown in Query for the Country Input Control, you'd add the 3 fields to the list in the order you want them to appear. When specifying fields in the list of visible query columns, use the full ID of the field, including any set IDs. For example, the following list of fields:

```
expense_join_store.ej_store_store_country
expense_join_store.ej_store_store_state
expense_join_store.ej_store_store_city
```

creates a list of values such as the following for users to choose from (the separator | is added automatically):

USA | CA | Los Angeles

USA | CA | San Francisco

USA | OR | Portland

USA | WA | Redmond

The Domain-based query also has the option to filter the query results, as shown in the following example:

```
<query>
  <queryFields>
    <queryField id="expense_join_store.ej_store_store_city" />
    <queryField id="expense_join_store.ej_store_store_country" />
    <queryField id="expense_join_store.ej_store_store_state" />
  </queryFields>
  <queryFilterString>expense_join_store.ej_store_store_country == 'USA'
and
        expense_join_store.ej_store_store_state == 'CA'
  </queryFilterString>
</query>
```

The <queryFilterString> tag contains a DomEL (Domain Expression Language) expression that references the full ID of the fields, including any set IDs. For more information about DomEL, see the JasperReports Server Data Management Using Domains. The <queryFilterString> tag in a Domain query is equivalent to the WHERE clause of an SQL query. The list of fields in the <queryFields> tag must include all fields being referenced in the filter string.

The filter string of the Domain-based query may also reference hierarchical attributes. In the following example, the query will return the products with the brand name matching the value of the brand attribute:

```
<query>
  <queryFields>
    <queryField id="sales_fact_ALL.sales__product.sales__product__brand_
name"/>
  </queryFields>
  <queryFilterString>sales_fact_ALL.sales__product.sales__product__
brand_name
                  == attribute('brand')</queryFilterString>
</query>
```

Finally, you can specify sort columns to arrange your query results using the sortList element, as shown in this example:

```
<queryFields>
  <queryField id="ShipRegion" />
```

```
      <queryField id="RequiredDate" />
      <queryField id="OrderId" />
      <queryField id="ShipCity"/>
   </queryFields>
   <sortList>
   <sortColumn columnName="ShipCity" />
   <sortList>
</query>
```

JasperQL Query Language

The JasperQL is a JSON-based Query Language for defining a query to access data from a domain on JasperReports Server. However, unlike Domain ("sl"), JasperQL is selected for new domain-based queries. When defining the query for new domain-based queries, set the query language to JasperQL.

Figure 67: Setting Query Language to JasperQL

The JasperQL query language is only supported in JasperReports Server 7.8 or higher versions.

JasperQL Query Language supports advanced features such as aggregation, filtering, aliasing, and DomEL function to create calculated fields. In the following example, the JasperQL input query will return a table with two fields **store_sales_98** and **product_ name**. You can use distinctFields to eliminate the duplicate information. Also, you can filter the information you want to display, rearrange the information in a group using groupBy, sort the records using orderBy.

```
{
  "select" : {
    "fields" : [ {
      "field" : "sales_fact_ALL.sales_fact_ALL__store_sales_98"
    }, {
      "field" : "sales_fact_ALL.sales__product.sales__product__product_
name"
    } ],

    "distinctFields" : [ ],
    "aggregations" : [ ]
  },
    "where" : {
      "parameters" : [ ]
    },
    "groupBy" : [ ],
    "orderBy" : [ ]
}
```

In JasperQL, if you want your query input control to return a limited number of records, you can set a record limit using Limit.

Cascading Input Controls

A cascading input control is one whose values depend on the selection made in a previous input control. You create a cascading input control using parameters in the query string of a related input control. In other words, the parameter you defined for an input control can be used in another query-based input control.

In a query-based example of cities and states such as:

Los Angeles | CA
San Francisco | CA
Denver | CO

the query may still generate a list of hundreds of cities to scroll through. Even though each city is easy to identify with the state, scrolling through a long list is time consuming. With cascading input controls, this example can have two input controls, one for the state and one for the city: The city input control is empty until the user selects a state.

- When input controls are displayed, the query for the state input control returns an alphabetical list of unique state names.

- When the user selects a state, the query for the city input control is triggered and returns the list of cities in that state. When the user selects one and submits it, the city name is passed as a parameter to the report.

For an especially large number of cities, you can use more cascading input controls, such as region of state, to reduce the list.

The parameter values determined by each cascading input control may or may not be used in the report. For example, if the report only shows data about a city, the country input control exists only to speed up the choice of city. However, if the report also shows information such as city average compared to country average for a given measure, the country parameter is also used in the report.

Parameters in Input Control Queries

Parameter substitution in query input controls follows the same approach as for JasperReports queries. Queries of all types of data sources can use parameter substitution, and $P, $P! and $X (for SQL queries) parameters are supported. The $X notation has two principal forms explained in the following list:

- $P{parameter_name}
  The value of the parameter is substituted into the query. In cases where the parameter contains a string, the substitution inserts the proper escape characters to create valid SQL. Use this for single-select input controls and simple comparison operators such as greater-than or less-than. For example:
  select name from EMPLOYEES where deals_closed > $P{DEALS}
  Do not use $P{parameter_name} with equality because the parameter value can be null, and field = NULL is not valid SQL. Instead use $X{EQUALS, ...} as explained below.

- $P!{parameter_name}
  The value of the parameter is treated as raw text. The server replaces the parameter with the value of the input control without performing extra checking or value

escaping. This is used in complex cases where the input control provides a piece of the query or sometimes the entire query.

- $X{EQUALS, column, parameter_name} or $X{NOTEQUAL, column, parameter_name}
  This syntax performs equality verification and handles the case when the parameter value is null. Use this everywhere instead of the old column = $P{parameter_name} syntax. The $X{EQUALS...} syntax performs the following substitution before submitting the query:

  column = parameter_value -- when parameter_value is non-null

  column IS NULL -- when parameter_value is NULL

- $X{IN, column, parameter_name} or $X{NOT IN, column, parameter_name}
  Use this parameter for cascading with multiple-select input controls. The $X{IN...} operator is true when the field value matches any one of the multiple values of the input control. In the country/cities example, we can allow the user to pick any number of countries, and show all the cities in the selected countries. The query-based input control would have the following query:
  select city from ACCOUNTS where $X{IN, country, COUNTRIES}
  If the user selects the values Canada, Mexico, and USA in the COUNTRIES multi-select input control, the $X{} syntax translates into the following query for the CITIES input control:
  select city from ACCOUNTS where country IN ('USA','Canada','Mexico')

> 📝 When defining these parameters in a report, don't use a defaultValueExpression element. Due to a limitation in JasperReports Server, these parameters are null when a defaultValueExpression is provided.

The $X syntax also supports the following operators. They are all designed to handle null input by generating 0 = 0 when the parameter value is null:

Parameter Syntax	Meaning
$X{GREATER, column, parameter}	column > parameter
$X{[GREATER, column, parameter}	column >= parameter
$X{LESS, column, parameter}	column < parameter
$X{LESS], column, parameter}	column <= parameter

Parameter Syntax	Meaning
$X{BETWEEN, column, start_param, end_param}	start_param < column < end_param
$X{BETWEEN], column, start_param, end_param}	start_param < column <= end_param
$X{[BETWEEN, column, start_param, end_param}	start_param <= column < end_param
$X{[BETWEEN], column, start_param, end_param}	start_param <= column <= end_param

For more information on using $P, $P! and $X to build dynamic queries, see the JasperReports Library Ultimate Guide and the Jaspersoft Studio User Guide.

Any number of parameters can be used in a query, just as any number of input controls can be defined in a JasperReport. In addition to the standard input control parameters, a cascading input control query can use the built-in parameters described in Built-in Parameters for Query-based Input Controls.

Creating a Cascading Input Control

In this example, we'll create a cascading input control to allow users to report on orders for a specific city by selecting a country and then a city within that country. Our example uses Jaspersoft Studio to edit a JasperReport on the server. The report uses sample data from the SugarCRM database shipped with JasperReports Server.
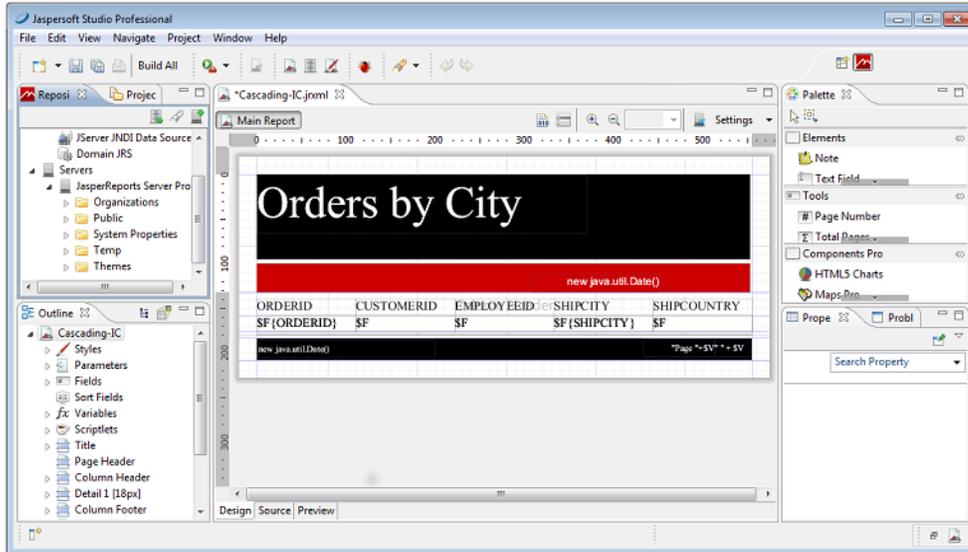
Figure 68: Simple Report Filtered by City

> You can connect Jaspersoft Studio to JasperReports Server. Once connected, the reports on the server will appear in the repository tree in Jaspersoft Studio. For more information see the *Jaspersoft Studio User Guide*.

We'll start with a report containing the following dataset fields: ORDERED, CUSTOMERID, EMPLOYEEID, SHIPCITY, and SHIPCOUNTRY.

Our sample report is based on the following query that requires the user to enter a city: select * from orders where SHIPCITY=$P{City}

Creating the Country input control

Procedure

1. On the repository tab, right click your report's folder and select **New** from the context menu. The Add JasperReports Server Resource wizard appears.

2. Select **Input Control** and click **Next**.

3. Enter Country for the **Name** of the input control and click **Next**.

4. Select the query type **Single Select Query**.

5. Click the ellipsis button next to the **Local Resource** field. The Query window opens.

6. Enter the SQL query:
   select distinct SHIPCOUNTRY from orders order by SHIPCOUNTRY
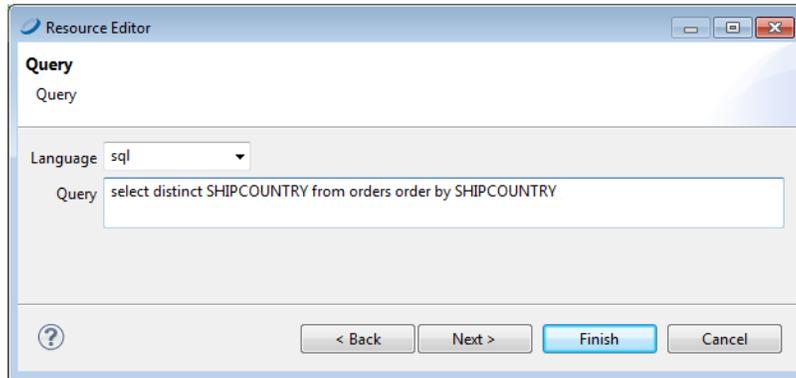
Figure 69: Query for the Country Input Control

7. Click **Next**. The Data Source window opens.

8. Click the **Data Source from Repository** option, select your data source, then click **Finish**.

9. On the **Value and Visible Columns** tab, enter SHIPCOUNTRY in the **Value Column** field and click **Add**. This is the column whose value is returned as the value of the Country input control.

10. Under **Visible Query Columns**, enter SHIPCOUNTRY. This is the column whose values populate the list of cities the user can choose from. In our case it is the same as the value column.

11. Click **Finish**.
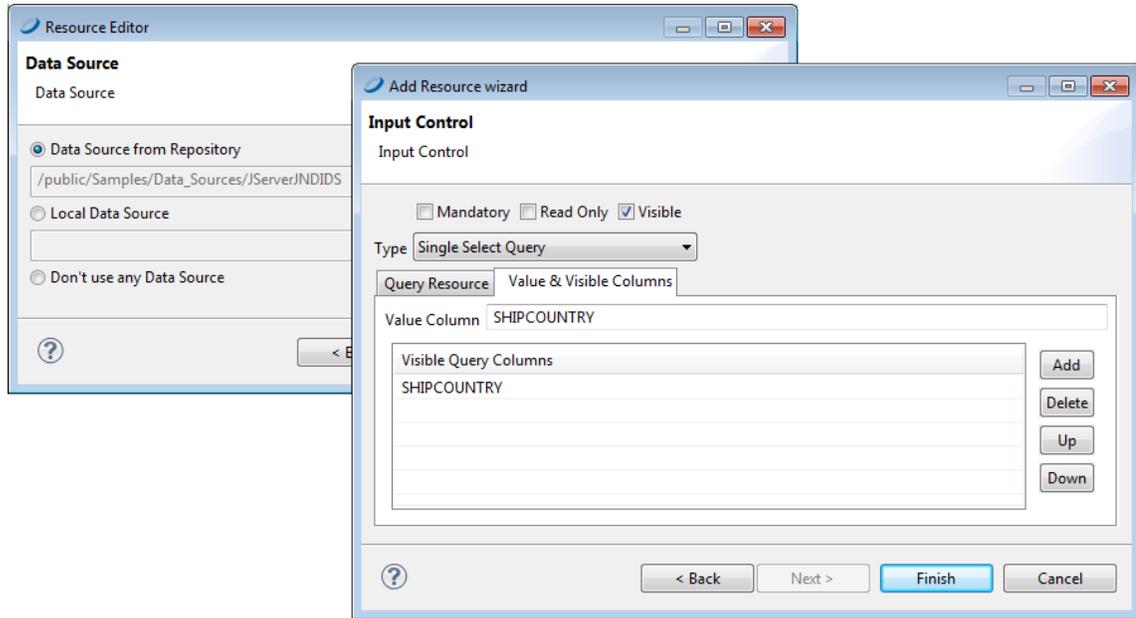
Figure 70: Country Control Value and Visible Columns

Creating the City input control

Procedure

1. Create a new City input control the same way you created the Country input control, using **Single Select Query** type.

2. Define an SQL query that uses the value of Country to derive valid values for City.

   ```
   Select distinct SHIPCITY from orders where SHIPCOUNTRY = $P{Country}
   order by SHIPCITY
   ```
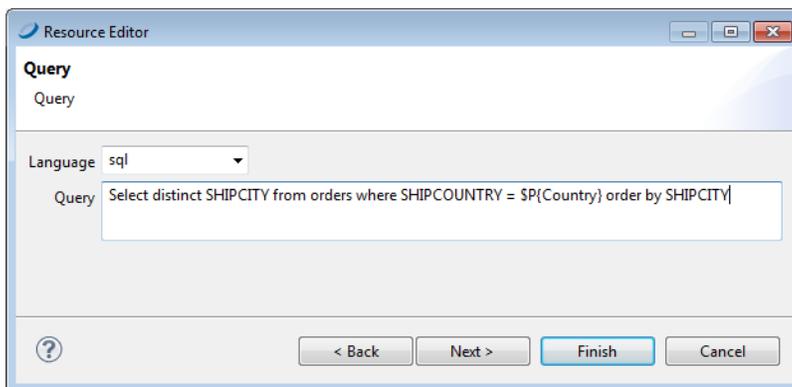

Figure 71: Query Using Country to Select Cities

3. Click **Next**. The Data Source window opens.

4. Click the **Data Source from the Repository** option, select your data source, then click **Finish**.

5. On the **Value and Visible Columns** tab, enter SHIPCITY in the **Value Column** field and click **Add**. This is the column whose value is returned as the value of the City input control.

6. Under **Visible Query Columns**, enter SHIPCITY. This is the column value used to create the list of cities the user can choose from.

7. Click **Finish**.
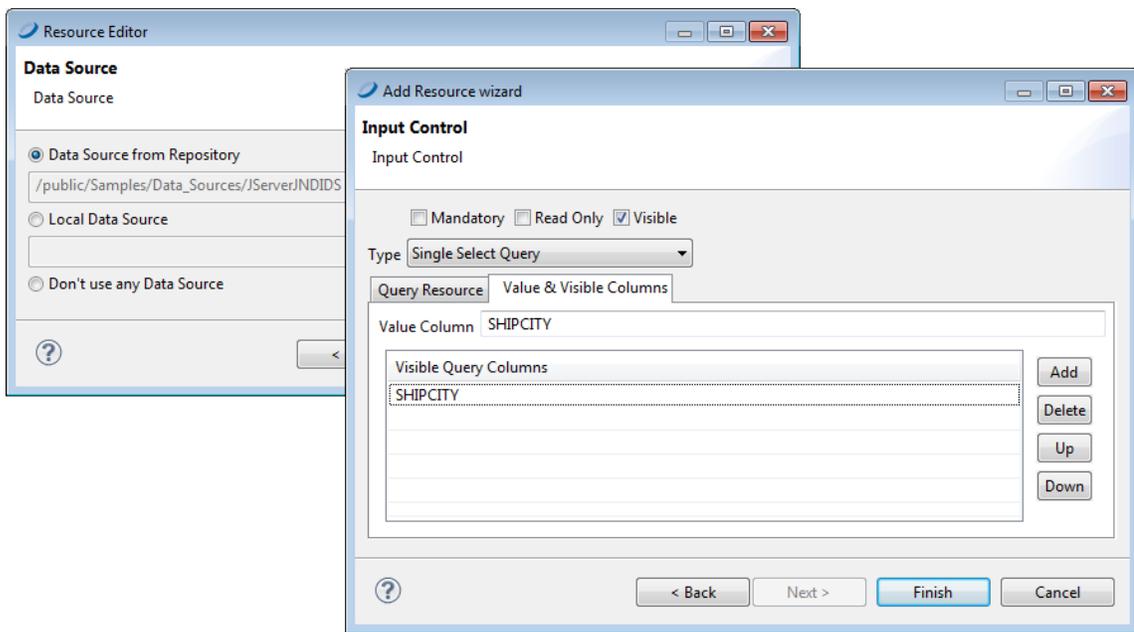
Figure 72: City Control Value and Visible Columns

8. Publish your report to JasperReports Server.
   Now when you run your report in JasperReports Server, You'll see an input control for country and another for city. After you select the country, the values available for city are those within that country. Notice that the Country value is not passed to the report. It is used only to help the user select a city.
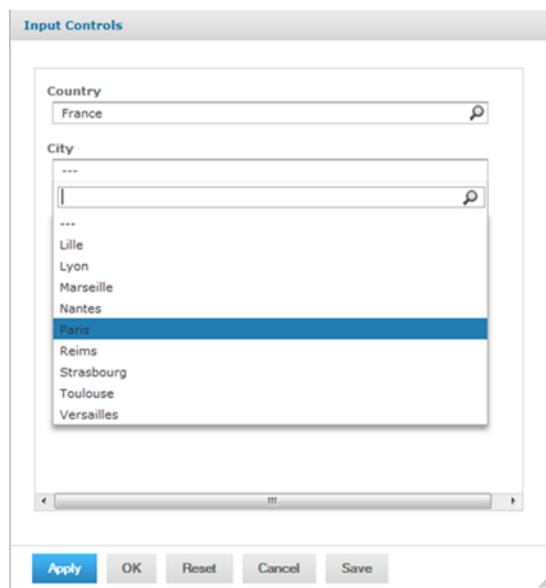
Figure 73: A Report with a Cascading Input Control

File Resources

You create file resources by uploading files so they can be referenced by Jasper Reports. JasperReports Server supports the following file types:

File Resource Types

File Type	Description
CSS	Cascading Style Sheet file that helps define the user interface as part of a theme.
Font	True Type font (.ttf) file to extend the set of fonts available in a report and allow embedding of fonts in the PDF output, if needed (see Fonts).
Image	Any image format supported by the JVM (Java Virtual Machine), such as JPEG, GIF, and PNG. Image files can be referenced in JasperReports, and also in CSS files.
JAR	Libraries that provide functionality for your reports (see JAR Files).

File Type	Description
JRXML	The definition of a report in JasperReports' XML-based report definition language. A JRXML file can be uploaded separately for use in multiple JasperReports.
OLAP Schema	Defines the data in an OLAP cube, including how to aggregate the dimensions.
Resource Bundle	A Java .properties file containing key-value pairs for localization of reports (see Resource Bundles).
Style Template	A JRTX file containing a style template that can be shared among JasperReports.
XML	XML file used in Domains and analysis (XML/A) to define data-level security. Can also be used for XML and JSON data files (see File Data Sources).
MongoDB JDBC Schema	JSON object that defines the structure and contents of your data. For more information, see MongoDB Schemas.
Azure Certificate	An x.509 server certificate (.cer) or key exchange (.pfx) file used to authenticate JasperReports Server with Microsoft Azure (see Uploading an Azure Certificate File to the Repository).
Secure File	An SSH private key file for SFTP file transfers that require an SSH key (see Uploading an SSH Private Key File to the Repository).

The way in which fonts, JAR files and resource bundles are associated with reports is further explained in the following sections.

Fonts

The server relies on the JasperReports Library as its content rendering engine, which enables it to produce high-quality, pixel-perfect documents. The server can use any of the fonts available to its JVM as logical or physical fonts. This solution is perfect for HTML reports that are stored in the server.

However, when exporting the report to PDF, you may need to take additional steps if the report includes fonts that the PDF viewer doesn't recognize, or if the report requires fonts that your users do not have on their computers. In this case, you must embed the font in the PDF file itself. To embed a font, you must edit the report's main JRXML file; the TTF (True Type Font) file that the report references must be available to the server at run time. One way to ensure that the server has the correct font is to upload it to the repository by creating a file resource. Then, the report can refer to the font's URI in the repository.

In the same way, you can export the dashboard to PDF when the report used in the dashboard is using external fonts. However, when custom fonts are used in the dashboard like text dashlets, you need to deploy related TTF files to the server environment if its operating system does not have the font that is used in the dashlets.

For details about working with fonts and PDF export, refer to the JasperReports documentation.

JAR Files

JasperReports can leverage third-party APIs. When run, reports can make direct API calls to third-party code using JRXML expressions. This provides enormous flexibility for incorporating business logic or other utility code into report generation.

In some cases, you can make the third-party code available to the report generating process by adding the necessary libraries to the server's application classpath when it's deployed. In other cases, upload the third-party or additional JAR files to the repository by creating a file resource. Then the report can refer to the code by referencing them as additional file resources.

Resource Bundles

When a single JRXML template is used to generate documents in multiple languages, it needs a resource bundle to accommodate the locale-specific content. If you upload such resource bundles by creating a file resource, your JRXML files can refer to them.

The name of the resource bundle created as a file resource in the repository must have .properties as its file extension. For example, the default resource bundle might be named MyReport.properties, and its French translation MyReport_fr.properties. For more information about resource bundles for reports, refer to the JasperReports Server User Guide.

Creating a File Resource

Administrators should organize file resources into folders in the repository to make them easier to find when creating references.

To add a file resource

Procedure

1. Log in as an administrator and select View > Repository.

2. In the Folders panel, right-click the parent folder's name and select Add Resource > File from the context menu , and select a resource type. In this example, Font. The Add File dialog appears.

3. Enter the required information for the file resource. In additions to the name and ID, file resources only require you to enter the path to a file. Click Browse to locate a file on your file system.
   The figure below shows the dialog for adding a Font file. All file resources are created by uploading a file in this way.

Figure 74: Adding a File Resource

4. When done, click Submit. The new file resource appears in the selected folder in the Repository panel.

Editing a File Resource

The following example shows how to edit a file resource.

To edit a file resource

Procedure

1. Log on as an administrator.

2. In the repository, browse or search for the resource.

3. Right-click the resource and select Edit from the context menu. In this example, we edit the font resource created in Creating a File Resource.

Figure 75: Editing a File Resource

4. Use the Edit dialog to view or modify the resource definition and its values. In the figure above, you can see how the Description field was changed. You can also change the contents of the file resource by specifying another file to upload. The Path to File field is not required unless you want to reload the file from disk.

5. Click Submit to save any changes.

Uploading an SSH Private Key File to the Repository

JasperReports Server allows you to export scheduled reports or dashboards over SFTP to an FTP server. If the FTP server uses SSH key authentication, you will need to upload a SSH private key file to the repository as a Secure File resource.

While the SSH private key is being used in a scheduled report or dashboard job, it cannot be deleted from JasperReports Server.

To upload an SSH key file to the repository

Procedure

1. Log into JasperReports Server as an administrator.

2. Click View > Repository and expand the folder tree.

3. Browse to the folder where you want to save the SSH key.

4. Right-click the folder and select Add Resource > File > Secure File from the context menu.

5. Click Choose File to locate and upload the SSH key file.

6. Enter a name and resource ID for the file.

7. Click Submit to save the file to the repository.

Themes

A theme is the set of all CSS (Cascading Style Sheets) and associated images that define the appearance of the user interface. Themes are stored in the Themes folder in the repository, with special menus on theme folders for activating, uploading, and downloading a theme. You can store any number of themes in the repository and switch among them, to change the UI's appearance.

For multi-organization deployments, administrators can set the theme individually for each organization, or rely on theme inheritance to use the same theme everywhere without needing to set it explicitly. Theme inheritance supports a mix of explicit and inherited settings, so that you can override any setting or image in a local organization, but inherit the rest of the theme from the parent or system-wide theme.

> In some cases, the set of files in the default theme changes between versions. Custom themes developed in for earlier version may need to be updated to work with theupdated theme. For more information, see the JasperReports Server Upgrade Guide.

This chapter contains the following sections:

- Introduction to Themes
- How Themes Work
- Administering Themes
- Creating Themes
- Working With CSS Files

Introduction to Themes

The default appearance of the JasperReports Server user interface (UI) can easily be modified to suit your needs.

The UI's appearance is determined by CSS (Cascading Style Sheets). A theme is a collection of CSS files and associated images that specify the appearance for all or part of the UI. A theme controls only the UI's looks, such as fonts, colors, lines, spacing, and images. It has no effect on content or functionality.

Themes are defined globally at the repository root and individually for each organization. Only administrators can set the theme. Administrators can add, upload, edit, copy, and delete the files that make up the theme, just like other resources in the repository. The repository allows special actions on theme folders for downloading and uploading themes as ZIP (compressed archive) files, and for activating the theme. Themes are active immediately without needing to restart the server.

> Themes are fully integrated with the multi-organization architecture in JasperReports Server. Some features of themes discussed in this chapter apply only to deployments that are licensed to use multiple organizations. However, single-organization deployments use the same architecture, for example there are overrides and inheritance between themes in the single default organization and the system root.

Themes are hierarchical and very flexible, allowing administrators to easily change the global appearance or set organization-specific overrides. For example, all of the following scenarios are possible with themes:

Scenario	Description
Use the default theme unchanged.	If the default theme suits your needs, there's no need to customize it or develop new themes. After a standard installation, the default theme is set at the root level and is automatically inherited by all organizations so every user sees the server with the same interface.
Quickly modify the default theme.	You can specify overrides of individual CSS rules or replace images in the default theme. After creating and uploading the new files, your theme is active immediately. Theme inheritance ensures that every organization sees your theme immediately as well.
Create an entirely new theme.	With CSS experience or Jaspersoft Professional Services, you can change the entire look and feel of the server. You can create a custom theme to match or blend with nearly any other web design. Inheritance again ensures that every organization uses the new theme, while allowing you to manage it from a single set of files.
Override themes to customize the UI for each organization.	You can give each organization or suborganization a customization of the default theme, for example a new logo, while retaining all other aspects. The benefit of this approach

Scenario	Description
	is that the default theme can still be modified and inherited by all organizations, while retaining each organization's overrides. This approach can be combined with the previous one, so that organization-specific overrides are applied to a custom theme.
Create a new theme in each organization.	For SaaS vendors, each organization can be a different client that needs a special interface. Themes allow each organization to fully define the UI and still retain override and inheritance for its own suborganizations. In such deployments, each organization admin can create or modify the appearance of his own user interface.

How Themes Work

Themes are stored in a special folder named Themes at the root of the repository and in every organization. Each Themes folder contains a default theme that can't be edited and any number of custom theme folders. Each theme is stored in its own folder and is known by the name of the folder.

The folder named default in every Themes folder is a special theme whose contents are controlled by the server. In the Themes folder at the root, the default theme contains the complete definition of every style that makes up the default theme shipped with JasperReports Server. In the Themes folder of every organization, the default is a system generated theme that contains all styles inherited by the given organization. None of the default theme folders can be modified, even by administrators.

> ⚠ You cannot modify the files of the default theme through the repository. If you try to do so by circumventing the repository, you could inadvertently change rules and make the UI unusable. If this happens, you'll need to re-install JasperReports Server to recover.

This chapter uses the following terminology to distinguish between root-level and organization-level themes.

Name	Folder	Description
Default theme	root > Themes > default	The unmodified UI as it appears at installation. The default theme is defined in the default folder in the Themes folder at the root of the repository.
System theme	root > Themes > active-theme	The active theme set at the root level. All users in all organizations see this theme unless an organization-specific theme is activated.
Inherited theme	Organization > Themes > default	The combination of all themes applied to the parent organizations and inherited by a suborganization. The organization's inherited theme is stored in the default folder within the organization's Themes folder.
Active theme	Organization > Themes > active-theme	The theme that's active at the organization or system level. Users see a combination of the active and inherited theme, depending on the files in the active theme and the inheritance rules.

Every level of organization, including the root, has a designated active theme. If no custom theme is made active, the default theme at every level is the active theme.

The following figure shows the files of the default theme in the Themes folder at the root of the repository. The name of the folder and its subfolders are bold indicating that it's the active theme.

Figure 76: Contents of the Root default Theme

Theme Files

A complete theme consists of the files listed for the default theme, as shown in the previous figure, along with all referenced images. The file samples.css is provided for reference when creating themes. The files overrides_ie7.css and overrides_ie8.css are only loaded with the style sheets when the user's browser is Internet Explorer 7 or 8, respectively.

The images associated with a theme include all the icons in the user interface and backgrounds for buttons and borders. Several icons and backgrounds can be stored in the same file called a sprite. The theme also includes the favicon.ico file that appears on browser tabs. There are over 70 image files in the default theme.

The image files for the default theme are stored in a folder named images. In a custom theme, there are two ways to change an image of the default theme:

- Create a folder named images and an image file with the same name as the one you want to replace.

- Modify the corresponding CSS rules to specify the name and location of a different image.

When you modify the CSS rules, you can use any of the following ways to reference image files or any other helper file:

- Directly in the theme folder. In this case the file is referenced without a path, for example "myfile.png" in CSS.

- In any folder path located in the theme folder. For example, your custom CSS file could refer to "MyImages/myfile.png" if you create a folder named MyImages in the theme folder and upload your images there.

- Anywhere on the Internet. Following the CSS standard, your custom CSS can refer to images or any helper file with a regular URL.

Inheritance

In order to render the user interface, JasperReports Server must load each of the CSS files listed in Contents of the Root default Theme. Because each file can be stored in multiple themes, inheritance determines which file to load. To locate the file, the server looks in the following locations, in the order listed below.

1. The active theme folder for the user's organization: <organization>/Themes/<active-theme>.

2. The inherited theme stored in the folder named <organization>/Themes/default.

When one of the CSS files references an image file or a helper file, including any path to that file, the server looks for that path and filename in the same two locations, in the same order. In this way, each file and image is resolved first in the active theme, and if not found, then in the inherited theme.

The active theme does not need to contain all the files because the inherited theme that is maintained by the server is guaranteed to contain all the files. Maintaining the inherited theme in every organization is the second task of inheritance.

The server maintains the inherited theme in each organization using the same algorithm. Whenever an administrator changes the active theme or modifies a file in the active theme, the server uses the same algorithm to find all files that define the active theme in this organization and makes a copy of them in every child organization. Thus, the active theme at the parent level becomes the inherited theme at the child organization level. For nested levels of organizations, the algorithm repeats on each level after updating the copy of the inherited theme. In this way, any changes are propagated down to every organization, and yet every level can maintain local overrides in its active theme. For more information, see .

> Propagating changes to the inherited themes is computationally intensive and can take several moments after making a change to a theme. However, determining inheritance when changes are made is an effective trade-off so that CSS files for rendering client request are resolved nearly instantly.

CSS Priority Scheme and Custom Overrides

Once inheritance determines which files to load, the standard CSS priority scheme determines which rules are visible, based on the order in which files are loaded.

This leads to two general ways of developing custom themes:

- The quickest way is to copy individual CSS rules from the default theme files, modify the rules to change the UI, and save them in the overrides_custom.css file. This is the only file in your new theme. Because overrides_custom.css is always the last CSS file to be loaded, its rules override the same rules in other files. This allows you to easily change any number of rules, and manage them all in a single file.
For example, if you want to increase the size of text on all the buttons in the default theme, you can do this with a few rules in the overrides_custom.css file. You may need to adjust the spacing for certain buttons, but the idea is you only need to change a limited number of rules.

- If you modify the user interface extensively, you can use the existing structure of CSS files in the default theme. In this case, copy the relevant files from the default theme, make your modifications, and save the files in your new theme. The new files are inherited when you activate the theme.

    An example of these extensive changes would be if you want to increase the size of the buttons themselves in the default theme. You would need to rewrite the majority of the rules in the buttons.css file and create images for the new buttons. In this case, it is much easier to copy the buttons.css file than to copy dozens of rules into the overrides_custom.css file. You could still use the overrides_custom.css file to adjust the spacing of elements around the buttons, because there would be fewer of those rules to modify.

    We recommend using the custom overrides method for most custom themes. A custom theme that changes simple appearances such as colors, fonts, and spacing has relatively few rules and is easily manageable in a single file. And many changes can be made by copying and modifying image files in the custom theme, without writing any CSS rules. Only if you change the fundamental layout

or appearance of the user interface, should you consider copying and modifying the other CSS file.

Copying and modifying CSS files is more prone to error, and is slightly less flexible due to file-based inheritance. Your copy of the file must contain all of the CSS rules as the original. If any rules are accidentally deleted or modified, even by a single character, the theme may not work properly. Also, the unmodified rules in the copy of your file now override any updates made to the same file in a parent organization.

For example, if you copy a file that defines gray buttons with plain text and you change the CSS rule to make the text bold, you have created a theme with bold, gray buttons. However, if the theme on the parent organization or system theme is modified so that buttons are blue, your file overrides the new inherited color, and you still have bold, gray buttons. If you had defined the bold text as a single rule in the overrides_custom.css file, your theme would show bold, blue buttons now.

Propagation

The propagation of themes is the internal server mechanism that copies theme files according to the inheritance rules and CSS priority scheme described above, so that they are available in each organization. In general, propagation does not occur until theme files are needed, in order to reduce the performance impact.

By definition, a theme is not displayed until a user of a given organization logs in and the UI needs to be rendered. Thus, when the first user of each organization logs in after a theme change, the server determines the theme inheritance and copies the necessary files to propagate the theme. If a user is already logged in when the administrator activates a theme, propagation occurs the next time that the UI is rendered for the user, such as when running a report or going back to the repository.

For example, when the system administrator sets the root theme, he or she can immediately look at the theme files in other organizations and see that they are unchanged. However, when a user of a given organization logs in, then the files of the inherited theme for that organization will be updated to reflect the theme change. Inherited themes in other organizations will not be updated yet.

Determining theme inheritance and copying theme files will have a performance impact, and the user may have a small delay after login or before the UI is rendered again. Administrators can use the login-as feature to force the propagation of themes in a given organization and also verify the appearance of the theme in that organization.

Administrators may also force the propagation of themes at server start-up with the following configuration setting:

Forcing Theme Propagation on Server Restart		
Configuration File		
.../WEB-INF/applicationContext-onStart-web-pro.xml		
Property	Value	Description
forceFullTheme PropagationOnStart	false (default)	When set to true, the server will propagate theme files to all organizations during the start-up process. This involves determining inheritance of all theme files and copying the necessary ones into every organization. Due to the overhead, this may affect server start-up times, however the first login into each organization will no longer have the delay from the propagation of themes.
		By default, this setting is false and no themes are propagated during start-up, thus avoiding the performance delay.
		This setting only affects server start-up. Between server restarts, changes to themes are still propagated only when needed by the first user of each organization.

Administering Themes

Themes are sets of CSS and image files stored in a folder in the repository. The root of the repository has a Themes folder where active and inherited themes are stored. In the repository browser, the Themes folder and individual theme folders have special actions for administrators to manage them. You can also use the repository search to find CSS and image files.

The folders and actions for managing themes are visible only to administrators. The Themes folder has execute-only permission for ROLE_USER so that all users can load the theme files and see the user interface, but not access the folders and files in the repository.

This section gives the basic procedures for administering existing themes. To create theme folders and files, see Creating Themes. For information about how to work with CSS in themes, see Working With CSS Files.

> The Easy Access theme is specifically intended to improve the web UI's accessibility. It increases color contrast and highlighting in the web UI. It can improve the user experience of those with visual impairment.

Setting the System Theme

This procedure assumes you have already created and uploaded your theme

Procedure

1. Log into JasperReports Server as system admin (superuser).

2. Click View > Repository and expand the root/Themes folder if necessary.

3. Right-click the chosen theme folder and select Set as Active Theme. For example, the sample data includes a theme called jasper_dark that you can set as active.
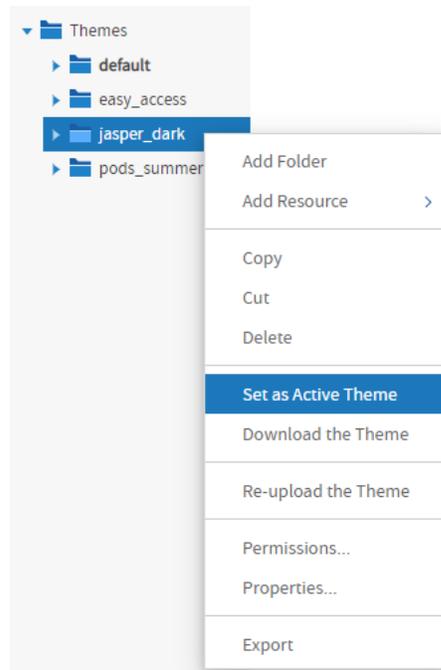
Figure 77: Setting a System Theme

As soon as the screen is refreshed, you see the effect of the new theme. Notice how the jasper_dark theme changes the colors in the user interface with just the overrides_ custom.css file and some image files.
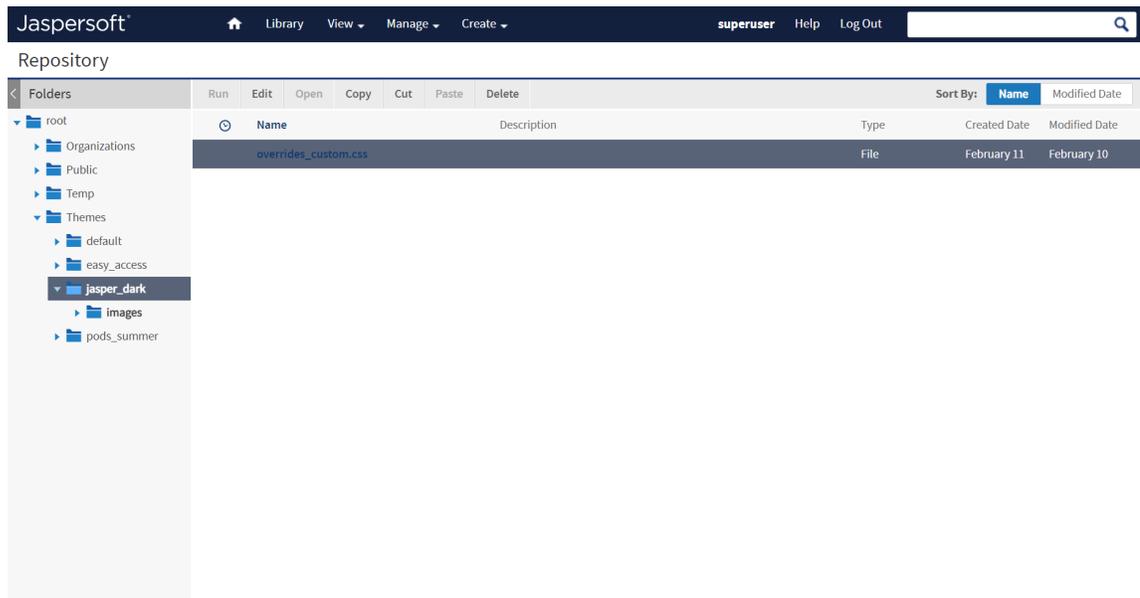
Figure 78: The Sample Theme jasper_dark

Because the system theme is set at the root level, the new theme appears to all users in all organizations, unless the organization has its own theme. When an organization has its own theme, it may still see elements of the system theme through inheritance. Also, the system theme set at the root applies to the login page, as shown in the following figure.
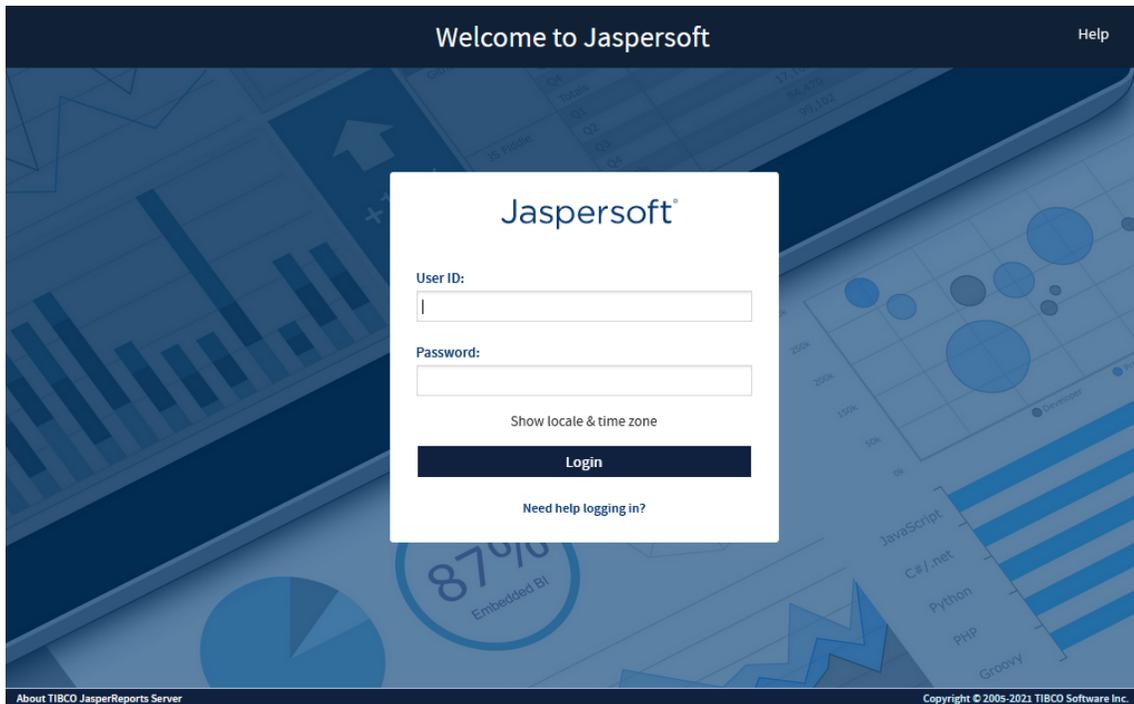
Figure 79: The Login Page as Seen With a New System Theme

The following procedures assume that the system theme is still set to the default theme.

Setting an Organization Theme

Professional edition users can give different themes to their organizations.

Procedure

1. Log into JasperReports Server as system admin (superuser) or as the organization admin (jasperadmin).

2. Click View > Repository and expand the Themes folder if necessary. The organization's Theme folder is shown in the following figure.

Figure 80: Organization Themes Seen by Organization Admin

3. Right-click the new theme folder name and select Set as Active Theme.
   As soon as the screen is refreshed, you see the effect of the new theme. The new theme applies to all organization users and is inherited by all suborganizations, if any.

   Organization admins can thus customize the user interface by creating and activating new themes within their organization.

Restricting Access to Themes

System admins may want to restrict access to themes, so that all themes are controlled from the system level.

> This procedure applies only to system admins. Organization admins cannot modify the ROLE_ ADMINISTRATOR permission, even in their suborganizations. They must request that the system admin perform the procedure for them.

Procedure

1. Log into JasperReports Server as system administrator (superuser).

2. Click View > Repository.

3. Expand the Organizations folder.

4. Locate the name of the organization where you want to restrict access to themes and expand its folder.

5. Right-click the Themes folder name and select Permissions.

6. Change the permission for the ROLE_ADMINISTRATOR from Administer to Execute Only.
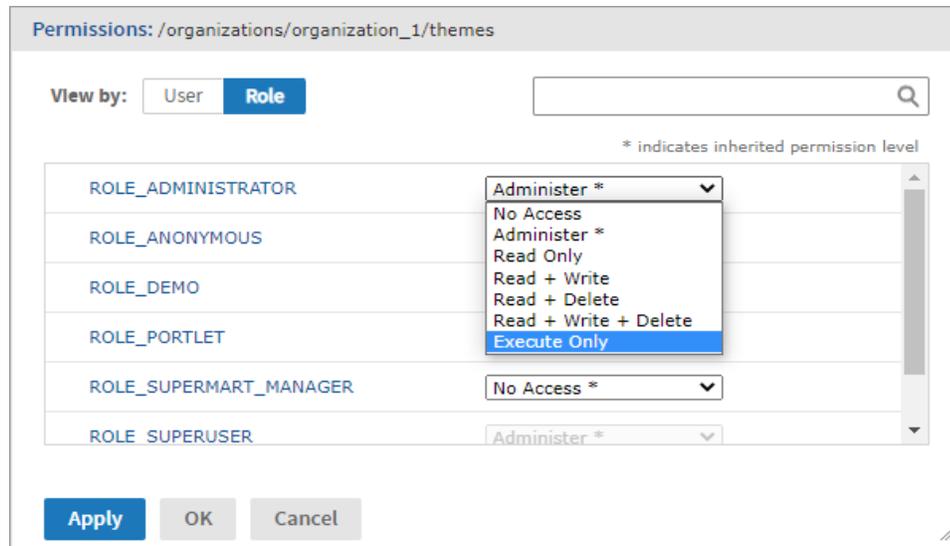
Figure 81: Restricting jasperadmin Access to Organization Themes

By setting Execute Only access, the organization admins cannot see the Themes folder in the repository, and thus cannot change themes or create a new theme.

> You shouldn't change any other permissions on themes, even if the permissions dialog allows it. You could inadvertently make the user interface inaccessible.

7. To restrict access to all organizations, repeat Locate the name of the organization where you want to restrict access to themes and expand its folder. to Change the permission for the ROLE_ADMINISTRATOR from Administer to Execute Only. for every organization in the server, including suborganizations.

8. If you want to restrict access in the same way in all future organizations, repeat Right-click the Themes folder name and select Permissions. and Change the permission for the ROLE_ADMINISTRATOR from Administer to Execute Only. in the Folder Template of every organization and suborganization in the server. Fore more

information, see Placing Themes in the Folder Template.

Creating Themes

There are three ways to create the folders and files that make up a theme:

- Creating Theme Folders and File Resources – Create them individually as resources in the repository.

- Downloading and Uploading Theme ZIP Files – Copy and modify existing themes as ZIP (archive) files.

- Placing Themes in the Folder Template – Initialize themes automatically in multi-organization deployments.

This section explains only how to store CSS files in the repository. For information about creating CSS file contents, see Working With CSS Files.

Creating Theme Folders and File Resources

A theme is simply a folder in the repository that contains CSS and image files, with optional sub-folders. Administrators can use the repository menus to create theme folders. System admins can create theme folders and files at the system level or in any organization. Organization admins can create theme folders and files in their organization or any suborganization.

To create theme folders and file resources

Procedure

1. Log in as an administrator with access to the location where you want to place the theme.

2. Click View > Repository and expand the folder tree to view the Themes folder where you want to place the theme.

3. Right-click the Themes folder and select Add Folder. Give your folder a name and optional description as you would when creating any folder. The folder name is used as the name of the theme.

> Theme folders and files can be created, copied or moved anywhere in the repository, but they can only be made active, uploaded, or downloaded when properly placed in a Themes folder.

4. Right-click your new folder and select Add Resource > File > CSS, and use the dialog to upload an individual CSS file. In order to be used as part of a theme, it must be one of the file names shown in Contents of the Root default Theme.

5. To add images to your theme, create any image folders and upload image files with Add Resource > File > Image.

6. Repeat Right-click your new folder and select Add Resource > File > CSS, and use the dialog to upload an individual CSS file. In order to be used as part of a theme, it must be one of the file names shown in Contents of the Root default Theme. and To add images to your theme, create any image folders and upload image files with Add Resource > File > Image. to create all the files and images you need. If several themes use the same files or images, you can copy-paste the file resources or entire image folders from one theme to another.

7. If you need to change the contents of a CSS or image file, you can right-click it and select Edit to specify another file to upload and replace the current file.

> If you upload CSS and image files into the active theme, the changes are visible after reloading the page in your browser.

Interacting with theme folders and files through the repository is a convenient and flexible way to create a theme. However, this method suffers from the limitation that, like other repository resources, you cannot download the files or images to edit them. For this purpose, the repository provides special download and upload actions on theme folders.

Downloading and Uploading Theme ZIP Files

The process of creating a theme often starts with the files of an existing theme that you modify with CSS and image editors on your computer. To support this workflow, every Themes folder has special commands for downloading and uploading themes.

Because a theme is composed of any number of files and folders, JasperReports Server uses the ZIP archive format to store a theme in a single file.

To download a theme ZIP file

Procedure

1. Log in as an administrator with access to the theme you want to download.

2. Click View > Repository and expand the Themes folder if necessary.

3. Right-click the theme folder you want to download and select Download the Theme. This menu selection appears only on theme folders inside the Themes folder.

4. The server prompts you to save the file named <theme-name>.zip. Save it anywhere on your computer.

5. Use an archiving or compression utility to extract the files from the ZIP file and save them on your computer.

Once you have the theme files extracted on your computer, you can view the individual CSS and image files that make up the theme. For example, to create your own theme, start by downloading the default theme from the root/Themes folder (as superuser). Save the extracted file on your computer and create your custom theme in another folder by copying and editing the CSS files and images of the default theme. See CSS Priority Scheme and Custom Overrides for an explanation of how to edit theme files.

When you have created all the files you need in your theme, upload it with the following procedure.

To upload a ZIP file as a theme

Procedure

1. Place the CSS files, optional folders, and images files that constitute your theme in a folder on your computer.

2. Use an archiving or compression utility to create a standard ZIP file of the contents of your theme folder.

> The ZIP file should include only the contents of your theme, not the theme folder itself.

3. Log in as an administrator with access to the location where you want to upload the theme.

4. Click View > Repository and expand the Themes folder if necessary.

5. Right-click the Themes folder and select Upload a Theme.
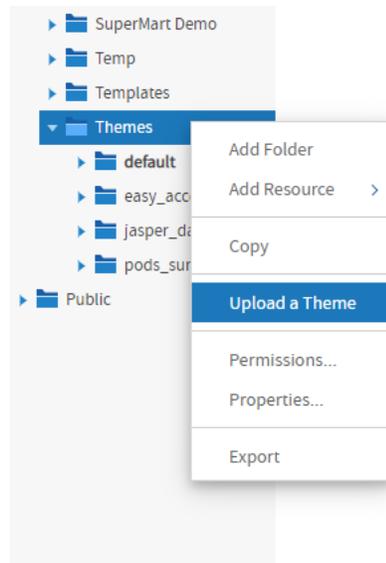
Figure 82: Uploading a Theme ZIP File in an Organization

6. In the dialog that appears, enter a name for your theme, and browse to find the ZIP file on your computer. Click Upload. The theme name becomes the name of the theme folder.

> You cannot use the ZIP upload dialog to overwrite an existing theme. You must specify a theme name that doesn't already exist in the chosenThemes folder.

The server uploads your ZIP file and extracts it contents. Then it creates a folder for the new theme and creates file resources in the folder for each of the CSS and images in your ZIP file. If you had sub-folders in your theme, they are created as well. After uploading your theme ZIP file, you can make it active to see effect of your theme on the user interface.

Creating a theme is an interactive process where you often need to make changes until you have the look and feel you want. To support this process, uploading ZIP files can be combined with the uploading of individual file resources that is described in Creating Theme Folders and File Resources. In fact, after an initial upload, it is much easier to update individual files in this way than to create the ZIP file and upload it again.

Placing Themes in the Folder Template

In deployments licensed to use multiple organizations, you can place a theme in the Folder Template that is used to create new organizations. The contents of the Folder Template

are copied to the folder of any new organization that is created at that level. Upon creation, new organizations always inherit their theme from the parent organization, but having a custom theme already present can save you time when customizing and activating it for the new organization.

The Folder Template in every organization is located in Organizations/Folder Template (organizations/org_template) and contains a Themes folder and a default theme folder. Do not modify the empty default folder, but create a new theme folder instead and place your files there. Because the Folder Template does not contain an active theme, there is no Upload Theme menu option on its Themes folder. Instead create the folder for your theme and upload files as resources, or copy an entire theme folder from the parent organization's Themes folder.

If you want to restrict access to the themes in created organizations, you can also set the permissions on the Themes folder in the Folder Template. To do this, follow the procedure in Restricting Access to Themes.

Working With CSS Files

This section is not a CSS tutorial but rather a collection of tips and tricks for working with the CSS that makes up the themes in JasperReports Server. This section focuses on how to test the themes you develop and match the CSS to its behavior in the JasperReports Server UI. There are many different editors for CSS and tools for testing it, so the recommendations in this section are just one way of developing a theme.

Theme Development Workflow

The major choice to make when developing a theme is whether to use simple theme overrides or to duplicate and modify theme files, as described in section CSS Priority Scheme and Custom Overrides. Usually, the extent of your modifications determines which method to use.

Once you have made that determination, you are ready to create your theme. The main steps in a theme development workflow are as follows:

Step	Reference
1. Download the default theme so you have a copy of the files and CSS rules that you want to modify.	Downloading and Uploading Theme ZIP Files
2. Create your new CSS rules, CSS files, and image files.	Firefox Web Developer Tools
3. Upload your new files to a test platform, and activate the theme or place them in an active theme.	Creating Theme Folders and File Resources
4. Verify your changes wherever they occur in the UI.	Test Platform
5. Repeat Create your new CSS rules, CSS files, and image files. through Verify your changes wherever they occur in the UI. for all your changes until the theme is finalized.	
6. Deploy your theme to your users.	Setting the System Theme or Setting an Organization Theme

Firefox Web Developer Tools

To help you find, view, and modify CSS rules in Create your new CSS rules, CSS files, and image files. above, you can use the web developer tools in the Mozilla Firefox browser. In particular, the Page Inspector tool displays the HTML and CSS rules of web pages as you browse. It has a dynamic interface that lets you select an element on the web page, and it displays the specific CSS rules that apply to the element. It also allows you to modify those rules in the browser memory and immediately see the effect on the web page.

The web developer tools are ideal for modifying themes in JasperReports Server. Once you locate the pages and elements that you want to modify, you can prototype your changes

directly within the tool. For example, you can see overall effect of changing a color or modifying the spacing.

If you are implementing your theme through custom overrides, you can copy the CSS rules from the tools directly into the overrides_custom.css file. Firefox displays the entire rule from its original file, so the copy overrides it exactly. If you are modifying other files from the default theme, the Page Inspector shows you the filename and line number of the rule, so that you can easily find it in your copy of the file.

And when you are testing a theme that uses overrides, the Page Inspector displays both the active CSS rule from overrides_custom.css and the original rule in the regular theme file of the inherited theme. The original rule is displayed in strike-through, so you can easily tell which rule is active and which rule it overrides, according to the CSS priority scheme.

For more information, see the Firefox Page Inspector tool overview and documentation.

Test Platform

When you upload a theme and make it active, it is immediately visible to every user in the organization, or in the case of a system theme, to every user on the server. Even editing or uploading a file into an active theme is reflected immediately in the user interface. Because developing a theme requires many iterations of uploading, activating, and testing CSS rules, you shouldn't develop themes on a production server.

In the simplest case, you can develop and test your themes before putting your JasperReports Server into production. As you test your server during the deployment, you can develop your themes without impacting real users.

For multi-organization deployments that are in production, you can test on your production server as long as you create a test organization. The test organization inherits from the system theme, creating a very realistic test environment where you can see how your theme overrides the inherited theme. Make sure your test organization reflects your real organizations, for example having OLAP views if your real organizations perform OLAP analysis. This can help you test your theme with the elements of the user interface that your organization users see the most.

For single-organization deployments that don't have a license to create organizations, you can test your themes on a second installation of the server. For example, you could download the evaluation copy of JasperReports Server and install it on the same computer where you develop the theme. This lets you see how your theme appears either as a system theme or in the default organization. As in the case of the test organization, test

your theme with all the server features, for example the Ad Hoc Editor, Dashboard Designer, input controls on reports, scheduling, and the like.

When your theme is well-tested and nearly complete, you should test it on the production server. Upload your theme to the Themes folder where you intend to deploy it, but do not activate it. Log in as a test user and add the following parameter to any URL, for example the home page URL:

```
&theme=<theme-name>
```

This activates your theme for the test user on all pages that you access until the user session times out. This allows you to navigate the entire application and see the effect of your theme in the production environment, without affecting other users.

> To set the theme back to the default append the &theme parameter to the URL with the string default (`&theme=default`). This is especially useful if a problem with the current theme has inadvertently disabled any functionality.

On each of these test platforms, look at the user interface generated by your theme with the same browsers and browser versions that your users have. If you see errors, you can still use the Firefox web developer tools to look at the CSS rules that are involved, even if the errors do not show up in Firefox. When testing your theme, look at its effect across all pages and dialogs of JasperReports Server. Your test organization and test users should access all the features of the server to view the user interface under all conditions.

Modifying the Appearance of Jaspersoft OLAP

Jaspersoft OLAP relies on a module called jPivot to display data when performing OLAP analysis. The jPivot module does not use all of the features of the new UI framework, but it supports some customizations through themes. For example, it does not use panels that can be hidden, and images for icons are not stored in a theme. However, some display characteristics of the analysis table are controlled by the theme, through the use of the analysisView ID in the theme file dataDisplays.css.

For example, you can change the lines between cells in the analysis table with the following rule in your overrides_custom.css file:

    #analysisView td { border: thin solid black; }

Alerts

Using the JasperReports Server alert feature, you can set alerts for reports from the Report Viewer toolbar. This functionality is available for Tables and Crosstabs. Alerts help you to monitor the data and notifies you when there is any significant change in the data. You can create, view, search, edit, and delete an alert.

For more information, see JasperReports Server User Guide.

Configuring Alerts

The prerequisite to contain configuration settings is that the email needs to be set up for alerting to function properly.

The scheduled alert run for the reports in the background according to a user-defined schedule. Alerts use the same engine and the same misfire policy as the Scheduler. You can configure alerts using the aspects of the Scheduler. For more information, see Configuring the Scheduler.

Performance Impact

The JasperReports Server could experience performance impact since both the Scheduler and Alert functionalities rely on the same engine. This might be due to various scenarios, for example, when multiple alerts are set on the same report the import operation affects server performance.

Recommendation

Whenever there is a performance issue on JasperReports Server, we encourage you to increase the thread count in the `js.quartz.base.properties` configuration file by setting the property `org.quartz.threadPool.threadCount=6`.

Admin Console

The Admin Console is a new page for JasperReports Server that allows system admins (`superuser`) and organization admins (`jasperadmin`) with detailed views of Schedules, Alerts, and Diagnostic data.

Users, other than administrators, do not have access to the Admin Console page and can view the Schedules, Alerts and Diagnostics details within the **View>Schedules and Alerts** page. For more information, see JasperReports Server User Guide.

This chapter contains the following sections:

- Schedules Tab

- Alerts Tab

- Diagnostics Tab

Schedules Tab

The Schedules tab in the Admin Console page is accessible to the system admins (`superuser`) and organization admins (`jasperadmin`).

List of All Schedules

All scheduled jobs that the user has defined appear in the Schedules tab of the **Manage>Admin Console** page. In the Schedules page, you can search, sort, add filter, refresh, and download a scheduled report. For more information on these operations, see List of Scheduled Jobs section in JasperReports Server User Guide.

Figure 83: Schedules Page

Typical users can see only the jobs that they have defined. Administrators can view the jobs defined by all users.

The Schedules page shows the following:

Column Name	Description
Job ID	The unique identifier of a job.
Job name/Description	Name of the scheduled job and description.
Resource/URL	Repository URL of the job.
Status	The status of the job in **NORMAL**, **EXECUTING**, **COMPLETE**, **PAUSED**, **ERROR**, or **UNKNOWN** state.
User	The owner who created the job.
Next run	Filters by date and time to view the count and list of schedules that will run next at the filtered time.
Last run	Filters by date and time to view the count and list of schedules that was run last at the filtered time.
Pause/Activate	Enables or disables the job.
Actions	Edit or delete the job.

Editing a Schedule

To edit a scheduled job for a report

Procedure

1. Click **Manage> Admin Console > Schedules**.

2. Click the Edit icon ✏ in the row of the job that you want to update.

3. Edit the fields in the **Schedule**, **Parameters**, **Output Options**, and **Notifications** tabs.

> 📝 See Creating a Schedule section in JasperReports Server User Guide and repeat from steps 4-10.

4. Click **Save**. The update occurs immediately.

Pausing a Job

To stop a job from running without deleting it, disable the job.

To pause a scheduled job

Procedure

1. Click **Manage>Admin Console > Schedules**.

2. In the row of the job that you want to stop, disable the **Pause/Activate** toggle switch.

> 📝 You can resume the job by enabling the **Pause/Activate** toggle switch. When a stopped job is re-enabled, it waits until the next scheduled time to run.

Deleting a Job

To delete a scheduled job

Procedure

1. Click **Manage> Admin Console > Schedules**.

2. Click the delete icon 🗑 in the row of the job you want to delete. A confirmation dialog with a warning message is displayed to confirm if you want to delete this job.

   If the delete request is sent for the job that is still running, then server completes running the job before deleting it.

3. Click **Delete** to delete the scheduled job else click **Cancel** to cancel the delete action.

4. Click **Close** to close the Scheduled job panel.

Alerts Tab

The Alerts tab is the new feature in the Admin Console page. You can view the list of all the report alerts in the Alerts tab. System admin (superuser) and organization admin (jasperadmin) have complete access to view report alerts in the Admin Console. The superuser can view the list of report alerts for all the users and the jasperadmin can view alerts of their respective organization. Other users do not have access to the Admin Console page and can only view report alerts of the folders with permissions assigned.

> Currently, the Alerts tab in the Admin Console page is a new beta feature that may not be fully stable or supported. For more information on the Alerts beta feature, see Admin Console Beta.

Diagnostics Tab

The Diagnostics tab in the Admin Console page provides a snapshot overview of your entire Jaspersoft system. Only the system admin (superuser) can view the diagnostic information. This user interface lets you quickly and easily view the status of license validity date, repository - database, configuration, size, total count of reports run, product details, and so on.

The following figure shows all the diagnostics information that the user has defined appear in the Diagnostics tab of the **Manage>Admin Console** page.

Figure 84: Diagnostics Page

The Diagnostics page includes the following controls:

Diagnostics Page Controls	Description
↻	Displays the latest updated diagnostics information with the recent date and time.
🔍	Displays the search term.
▤→	Downloads the diagnostic information of the report in CSV format.
See more	Expands the list of information for Report runs by day list, SQL key words, and Repository Details.
See less	Collapse the list of viewed information for Report runs by day list, SQL key words, and Repository Details.

The **See more** and **See less** controls lets you expand or collapse the viewed information in the Diagnostics page.

The information of the report displayed in the Diagnostics tab can be viewed on the **View>Repository** page. For more information on the Diagnostics, refer to the Using the Diagnostic Data in Reports.

Import and Export

The import and export tools enable you to move resources into and out of the JasperReports Server repository. These tools can back up the entire contents of repository, selected organizations, selected folders, or specific resources. These utilities also handle scheduled jobs, users, and roles that the server stores internally. Import and export can be helpful when migrating between versions of JasperReports Server, when managing organizations, or when moving between test and production environments.

JasperReports Server provides both a user interface and command-line utilities to perform import and export. The functionality in the UI is available only to administrators (superuser and jasperadmin) and the command-line utilities require access to the file system where the server is installed.

This chapter contains the following sections:

- Import and Export Catalogs
- Dependencies During Import and Export
- The Import-Export Encryption Keys
- Import and Export Through the Web UI
- Import and Export Through the Command Line
- Alternate Import-Export Scripts

Import and Export Catalogs

The output of the export command and the input to the import command is called a catalog. It's a set of folders and files that comprise the server's internal database, including organizations, users, roles, scheduled jobs, and repository resources such as reports and associated files. When you don't need the entire database, you can specify options to export only the contents you need, for example one role and its users.

Within a catalog, user passwords are encrypted so that they are not visible outside the server. For more details, see The Import-Export Encryption Keys below.

The catalog is usually exported as a single zip file (compressed archive) containing all the files and folders. However, the content of the catalog is not intended for external access. Objects in the database, such as users, roles, and folders are described in XML files, and repository resources are stored in various private formats consisting of data files and subfolders. The XML syntax of the catalog files is not publicly defined, and the data files aren't meant to be accessed.

To access and interact with the server's internal objects, use the REST v2 API. This web service has well defined datatypes and resource descriptors in XML or JSON formats and a complete set of methods for reading and writing objects on the server. For more information, see the JasperReports Server REST API Reference.

> While user passwords and data source passwords are encrypted in export catalogs, you should still take appropriate measures to secure the catalog file from unauthorized access. Catalog files may contain sensitive metadata such as user names, database URLs, and customer information (in the form of organizations). Catalog files may also contain data in the form of report output such as the PDF of an executive report.

Dependencies During Import and Export

The resources in the repository often have dependencies on other resources, for example a report that relies on images, input datatypes, and a data source. When exporting a resource, you have the option of including all of its dependencies, even if they are stored in folders that were not selected in the repository. Importing a catalog that contains such dependencies will re-create the same folder structure in the target repository. Once imported, you can move and redefine the dependencies of these resources.

However, when exporting individual organizations, there are three scenarios that create broken dependencies:

- An organization admin exports a resource with dependencies that he does not have permission to access.

- The export from an organization includes a resource with a dependency in a parent organization. For reasons of consistency, even the system admin (superuser) can't export a dependency in a parent organization if the parent organization is not explicitly selected for export.

- Unchecking the Include dependencies check box when exporting from the repository in the Web UI.

In these cases, you will see an error to inform you that the dependency cannot be exported. If you choose to proceed with the export, it will contain a broken dependency that may block it from being imported.

To avoid dependencies between organizations, you can place shared resources in the Public folder and set permissions to restrict access and visibility. Dependent resources in the Public folders should be visible to organization admins, but not to regular organization users.

The Import-Export Encryption Keys

As of JasperReports Server 7.5, the management of the encryption keys used during import and export has been automated. These encryption keys are used by the server to encrypt user passwords so they are not revealed in export catalogs.

In previous versions of the server, the import-export key had to be configured manually outside the server. Beginning with version 7.5, the server creates and manages its import-export key internally, and also includes UI and REST options for specifying keys during import and export operations. However, there are still cases when you may need to manage keys on the server and specify keys during import-export operations.

The following table summarizes the use of keys when importing catalogs into the current version of the server:

Origin of Export Catalog	Guidelines for Importing into 8.2.0
Prior to version 7.5 With default key	The current version of the server handles previous export catalogs created with legacy keys, and even older catalogs that did not use encryption. When importing through the UI, specify the Legacy key, and when importing through the command line, specify the `deprecatedImportExportEncSecret`.
Prior to version 7.5 With custom import-export keys	The custom keys are not known to your server, so in order to import these catalogs, you will need to share the custom key with the importing server. There are two ways to specify custom keys:

Origin of Export Catalog	Guidelines for Importing into 8.2.0
	Procedure 1. For one-time or occasional imports, you can enter the key's hex value into the UI or the import command-line, or store it in the repository for repeated use. 2. For continual use, you should import the key to server's keystore with the command-line import command. Then the key is available by it's alias either through the UI or the command-line when importing a catalog.
Version 7.5 and later From the same server	When you import a catalog back into the same server, the server uses the same key for export and import and thus can read its own export catalog. When importing through the UI, specify the Server key, and when importing through the command-line, the key is detected automatically. This is also true after upgrading the server, as long as the catalog was exported from version 7.5 or later and the server's keystore was preserved during the upgrade. For more information about the keystore and catalogs during upgrade, see the JasperReports Server Upgrade Guide.
Version 7.5 and later From a different server	Because each server has unique randomly-generate keys, the keys of the export server are not known to your server. However, unlike the custom key scenario above, you do not have direct access to the server's keys. Before you can move export catalogs between servers, such as test and production servers, you will need to generate and share a key between the servers: **Procedure** 1. Generate a key during the export operation on the first server. 2. Import the new key into the second server's keystore with the command-line import command. 3. Import the catalog into the second server and specify the custom key either through the UI or the command-line.

Origin of Export Catalog	Guidelines for Importing into 8.2.0
	If you also import the key back into the originating server, both servers will share the same key. Then if you specify this custom key during import and export, catalogs can be exported from either sever and imported into the other one.

In addition to the default import and export operations, this guide documents the following:

- Importing catalogs from older servers with legacy or custom keys.
- Exporting catalogs with a specific key.

For operations to set up and manipulate keys, see the JasperReports Server Security Guide:

- Specifying custom keys during import and export operations.
- Importing keys used by other servers.
- Exporting keys for use in other servers.
- Sharing custom keys between multiple servers.

Import and Export Through the Web UI

JasperReports Server allows administrators to import and export resources and users through the user interface:

Section	Actions	Users
Exporting from the Repository	Export entire folders Export selected resources	System admins and Organization admins
Exporting from the Settings	Export everything Export selected users or roles Export resources by type	System admins only
Exporting from Organizations	Export entire organizations	System admins and

Section	Actions	Users
	Export selected users or roles Export resources by type	Organization admins
Importing to Organizations	Import into a specific organization	System admins and Organization admins
Importing from the Settings	Import any catalog into the server	System admins only

Exporting from the Repository

System administrators and organization administrators can browse the repository and easily export folders and individual resources from the context menu. System administrators can export any resource from the repository while organization administrators can only export resources within their organization and the Public folder.

> Catalogs may be very large and take a long time to generate and then download. During this time, the export operation may affect server performance.

To export resources from the repository

Procedure

1. Log in as an administrator that has access to the resources you want to export.

2. Select **View>Repository**.

3. Select one or more resources in the main panel, or select a folder in the left-panel.

4. Right-click the selected folder or resources and select **Export** from the context menu. The Export Resources dialog appears:

Figure 85: Export Resources Dialog in the Repository

5. If required, change the default name of the zip file for the exported catalog.

6. Select the import-export encryption key to protect any passwords in the export catalog. For more information, see The Import-Export Encryption Keys.

- Use the **Server Key** if this is a backup to be imported into the same server later.

- Use the **Legacy Key** if the catalog has to be imported into a different server.

- If you have added custom keys to the keystore on this server, select one from the list that appears:

7. Choose the export options:

- **Include report jobs** - When checked, the export includes scheduled report jobs that are associated with any reports in your repository selection.

- **Include alerts** - When checked, the export includes scheduled report alerts that are associated with any reports in your repository selection.

> The exported report includes alerts only for the reports for which the alert is turned on. If the alert is turned off for the report, then the exported report includes the alert with an empty report alerts folder.

- **Include repository permissions** - When checked, the export includes any explicit permissions on all items in your repository selection. When this option is cleared, the exported items inherit the permissions of the repository where they are later imported.

- **Include dependencies** - When checked, the export includes all dependencies for your resources, even if they are not included in your selected folders. For more information, see Dependencies During Import and Export.

- **Include full resource path** - This option appears when the selected files or folders belong to an organization. When checked, the export includes the full file structure of each resource, starting with the Organizations root folder. This helps for archiving items that would be reimported later into the same organization. If you want to import the resources into a different organization, uncheck this box so the export includes only the selected items and their relative path within the organization.

8. Click **Export**. The server generates the catalog zip file. Depending on the size of the repository and the options selected, it may take several minutes to generate the catalog file.

Exporting from the Settings

For more export options, use the server settings page for system administrators. The settings export the following resources:

- Everything - The entire repository, including all resources in all organizations, as well as all users, roles, and other settings stored internally. With the proper options, this creates a backup of the server.

- Any combination of users and roles - Lets you choose from lists of users and roles, with options to include users by role or roles by user.

- All resources of a given type - For example, all reports or all dashboards.

To export resources from the settings page

Procedure

1. Log in as a system administrator (`superuser` by default).

2. Select **Manage>Server Settings**, then click **Export** in the left-hand panel.
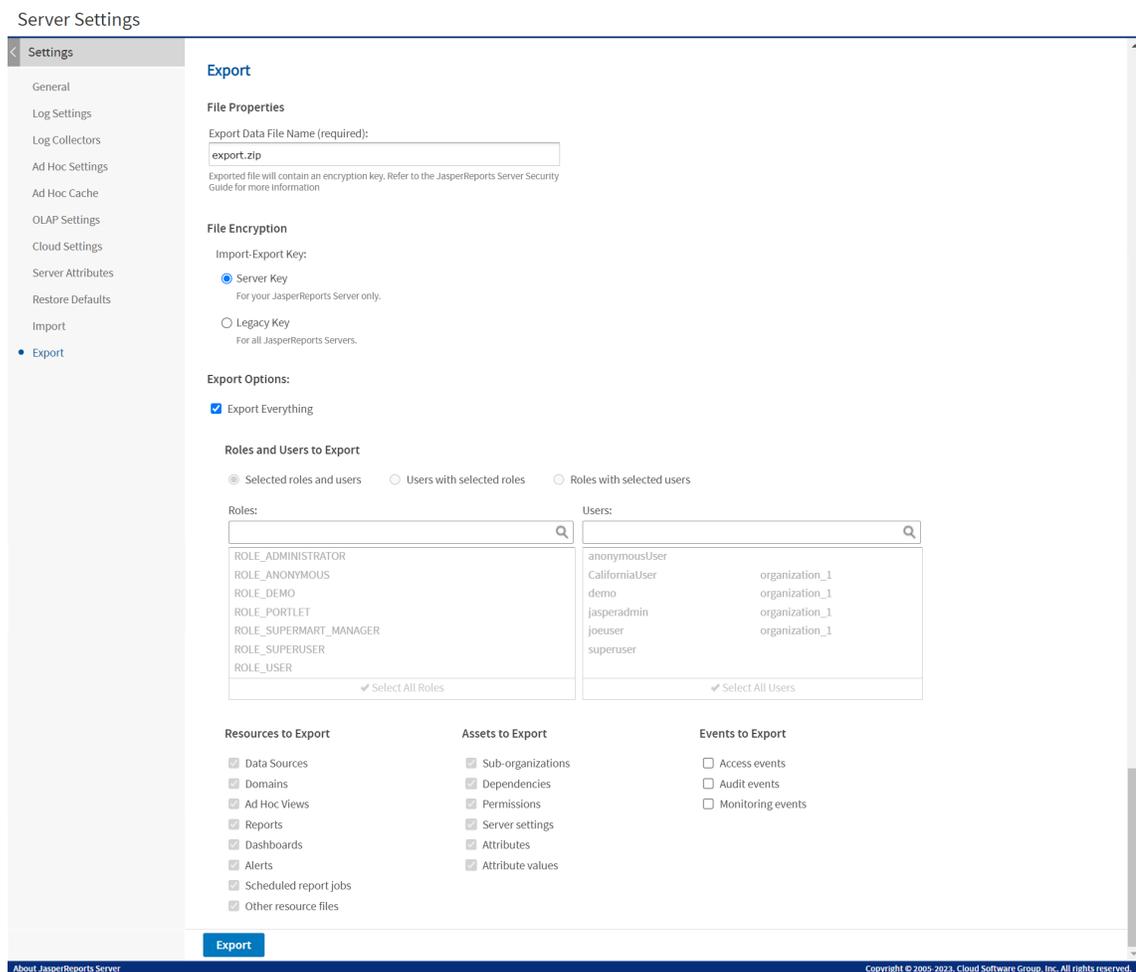
Figure 86: User Interface for Export

3. If required, change the default name of the zip file for the exported catalog.

---

4. Select the import-export encryption key to protect any passwords in the export catalog. For more information, see The Import-Export Encryption Keys.

- Use the **Server Key** if this is a backup to be imported into the same server later.

- Use the **Legacy Key** if the catalog has to be imported into a different server.

- If you have added custom keys to the keystore on this server, select one from the list that appears:
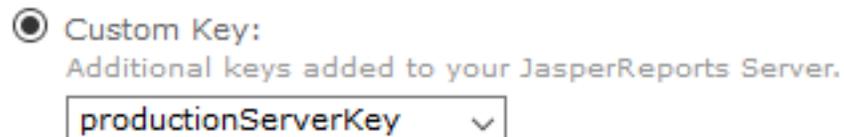
5. Use the checkboxes and radio buttons to choose the contents of your exported catalog file:

Select **Export Everything** (default) to export the entire repository, including all organizations, users and roles, and all types of assets. Select the checkboxes under Events to Export to include the different types of events in your export catalog.

6. Clear **Export Everything** to select users and roles or resource types to export.

a. To export roles and users, choose one of the following radio buttons to select individual users and roles from the lists:

- **Selected roles and users** - Only the roles and users you select explicitly are exported.

- **Users with selected roles** - Select one or more roles, and all users with those roles are exported, along with the selected roles.

- **Roles with selected users** - Select one or more users, and all roles assigned to those users are exported, along with the selected users.

b. In Resources to Export, perform either one of the following:

a. If you only want users and roles, clear all checkboxes.

b. If you only want resources, do not select any users and roles, then select the resource types you want to export.

c. Select the checkboxes under Assets to Export to include these various assets in your export catalog.

d. Select the checkboxes under Events to Export to include the different types of

monitoring events.

e. Select the checkbox under Events to Export to include repository create and modify dates.

7. Click **Export**. The server generates the catalog zip file.

> Resources are exported along with any dependencies, even if they are not included in your repository selection. For more information, see Dependencies During Import and Export.
>
> Depending on the size of your repository, and the options you have selected, it may take several minutes to generate the catalog file. During this time, the export operation may affect server performance.

Exporting from Organizations

Both system admins and organization admins can use the Manage Organizations page to select organizations for export. For example, an organization admin can use this feature to back up his entire organization or to move suborganizations from one parent to another.

To export organizations

Procedure

1. Log in as an administrator that has access to the organizations you want to export.

   For example, to move an organization, you must log in as the administrator of the parent organization.

2. Select **Manage>Organizations** to display the hierarchy of organizations.

3. In the left-hand panel, right-click the organization you want to export and select **Export** from the context menu.

Export Organization

File Properties

Export Data File Name (required):

organization_1_export.zip

Exported file will contain an encryption key. Refer to the JasperReports Server Security Guide for more information

File Encryption

Import-Export Key:

○ Server Key
For your JasperReports Server only.

○ Legacy Key
For all JasperReports Servers.

Export Options:

☑ Export Everything

Roles and Users to Export

○ Selected roles and users   ○ Users with selected roles   ○ Roles with selected users

Roles: Users:

Roles	Users	
	CaliforniaUser	organization_1
	demo	organization_1
	jasperadmin	organization_1
	joeuser	organization_1

✓ Select All Roles   ✓ Select All Users

Resources to Export   **Assets to Export**

☑ Data Sources   ☑ Sub-organizations
☑ Domains   ☑ Dependencies
☑ Ad Hoc Views   ☑ Permissions
☑ Reports   ☑ Attributes
☑ Dashboards   ☑ Attribute values
☑ Alerts
☑ Scheduled report jobs
☑ Other resource files

Export   **Cancel**

Figure 87: User Interface for Organization Export

4. If required, change the default name of the zip file for the exported catalog.

5. Select the import-export encryption key to protect any passwords in the export catalog. For more information, see The Import-Export Encryption Keys.

   - Use the **Server Key** if this is a backup to be imported into the same server later.

   - Use the **Legacy Key** if the catalog has to be imported into a different server.

   - If you have added custom keys to the keystore on this server, select one from the list that appears:
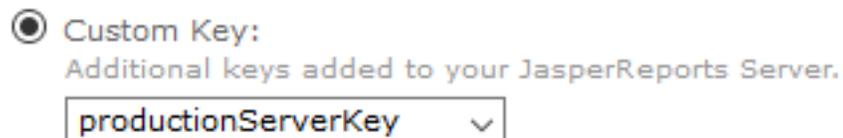
6. Use the checkboxes and radio buttons to choose the items to be exported from this organization.
   Select **Export Everything** (default) to export the entire organization, including all resources, report jobs, users, and roles.

7. Clear **Export Everything** to select users and roles or resource types to export.

   a. To export roles and users, choose one of the following radio buttons to select individual users and roles from the lists:

      - **Selected roles and users** - Only the roles and users you select explicitly are exported.

      - **Users with selected roles** - Select one or more roles, and all users with those roles are exported, along with the selected roles.

      - **Roles with selected users** - Select one or more users, and all roles assigned to those users are exported, along with the selected users.

   b. In **Resources to Export**, perform either one of the following:

      - If you only want users and roles, clear all checkboxes.

      - If you only want resources, do not select any users and roles, then select the resource types you want to export.

      - Select the checkboxes under **Assets to Export** to include these various

resources in your export catalog.

8. Click **Export**. The server generates the catalog zip file. The server displays a message if there are any broken dependencies.

> Resources are exported along with some dependencies. You must have permission to access them or they should be in the parent organization. Broken dependencies may block the export or import operations. For more information, see Dependencies During Import and Export.
>
> When logged on as system admin (`superuser`) and exporting from the root node, the dialog is functionally equivalent to exporting from the Settings page. In this case, there are additional options to export events.
>
> Catalogs may be very large and take a long time to generate and then download. During this time, the export operation may affect server performance.

Importing to Organizations

Both system admins and organization admins can use the Manage Organizations page to import an organization. An import merges the content of the export catalog into the selected organization. When merging organizations with the update option, the contents of the import override the target organization for any user, role, or resource with the same name. If the imported organization has a different ID than the destination, the merged organization takes the organization ID of the imported organization.

To move an organization, create an empty organization as the destination, and then import it into it.

To import organizations

Procedure

1. Log in as an administrator that has access to the destination organization.

2. Select **Manage>Organizations** to display the hierarchy of organizations.

3. In the left-hand panel, right-click the organization you want to import into and select **Import** from the context menu.

Figure 88: User Interface for Organization Import

4. Click **Browse** to choose the catalog zip file to import. The catalog file must be created by the export of an organization.

5. Select the key that was used to export the catalog file. For more information, see The Import-Export Encryption Keys.

   - Use the **Server Key** if the catalog was exported from the same server with the server key.

   - Use the **Legacy Key** if the catalog was exported from any server with the legacy

key, or if it was exported from any server prior to release 7.5.

- If you have added custom keys to the keystore on this server, select one from the list that appears:

◉ Custom Key:
Additional keys added to your JasperReports Server.

| productionServerKey ∨ |

- Use the Key Value and enter a key in hexadecimal notation if you exported with a one-time key (the command-line js-export --genkey option).

- Use the Stored Key and specify its location if you saved the export key in the repository.

6. Use the checkboxes to change the behavior of the import operation:

- **Update** - When checked, it imports only resources that are newer than ones with the same URI in the current organization. The **Skip user updates** field allows you to keep the current definition of any users that also exist in the imported organization.

- **Include alerts** - When checked, it imports all alerts of any users that exist in the imported organization.

- **Include themes** - When checked, imports any themes that exist in the imported organization.

7. Click **Import**.

The server uploads the catalog zip file and imports its contents into the organization. If there are any broken dependencies in the catalog, the server displays a message with three choices:

- Skip - Does not import the resource with the broken dependency, but continues to import other resources.

- Include - Attempts to import the resource with the broken dependency. The import succeeds if there is already a resource in the destination that satisfies the dependency. If the dependency is not satisfied in the destination, the resource is skipped and the import continues.

- Cancel - Stops the import operation.

> 📝 For more information, see Dependencies During Import and Export.
>
> When logged on as system admin (superuser) and importing to the root node, the dialog is functionally equivalent to importing on the Settings page. In this case, there are additional options to import events. But you cannot import an organization to the root.
>
> Catalogs may be very large and take a long time to upload and then process. During this time, the import operation may affect server performance.

Importing from the Settings

The Settings pages for system administrators include a user interface to simplify the import procedure.

This import operates on a running instance of the server, and all imported resources are visible immediately. In addition, any configuration or security settings in the imported catalog take effect immediately, with no need to restart the server.

To import data from the Settings page

Procedure

1. Log in as a system administrator (superuser by default).

2. Select **Manage>Server Settings** and choose **Import** in the left-hand panel.
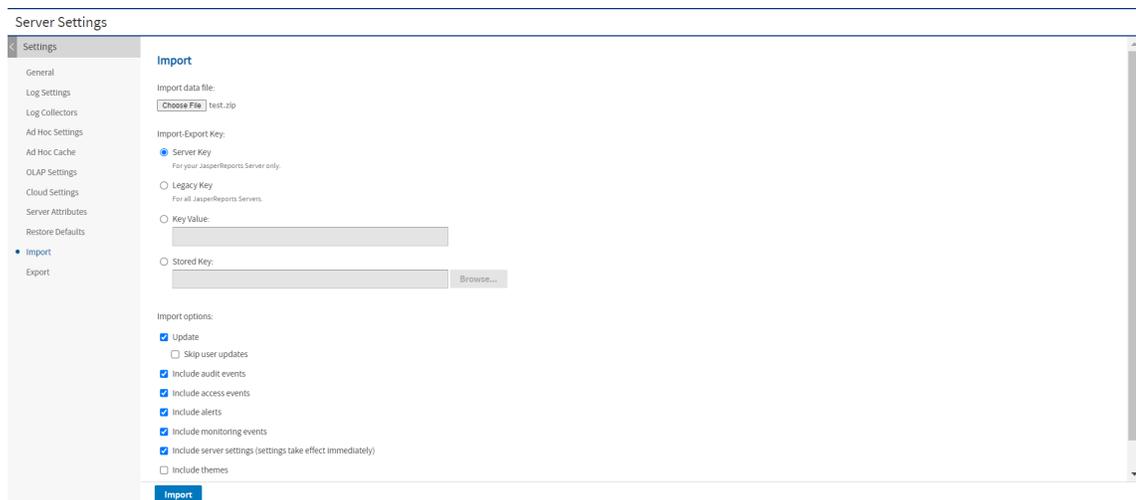
Figure 89: User Interface for Import

3. Click **Browse** to choose the catalog zip file to import. This dialog cannot import a catalog file that was created from the export of an organization.

4. Select the key that was used to export the catalog file. For more information, see The Import-Export Encryption Keys.

- Use the **Server Key** if the catalog was exported from the same server with the server key.

- Use the **Legacy Key** if the catalog was exported from any server with the legacy key, or if it was exported from any server prior to release 7.5.

- If you have added custom keys to the keystore on this server, select one from the list that appears:

  ⦿ Custom Key:
  Additional keys added to your JasperReports Server.

  productionServerKey ▾

- Use the **Key Value** and enter a key in hexadecimal notation if you exported with a one-time key (the command-line js-export --genkey option).

- Use the **Stored Key** and specify its location if you saved the export key in the repository.

5. Use the checkboxes to change the behavior of the import operation:

- **Update** - When checked, it imports only resources that are newer than ones with the same URI in the current repository. The **Skip user updates** field allows you to keep the current definition of any users that also exist in the imported catalog.

- **Include access events** - When checked, it imports the modification times of resources from the catalog. When cleared, resources keep their existing access times if they already exist.

- **Include alerts** - When checked, the import includes scheduled report alerts that are associated with any reports in your repository selection.

  > 📝 Importing alert details for a single report is comparatively quicker than importing alert details for the report having multiple alerts. As including multiple alerts in the report results in downgrading the performance of the server.

- **Include audit events** and **Include monitoring events** - These fields determine

whether access and monitoring events from the catalog, if any, are imported.

- **Include server settings** - This field determines whether the system configuration is updated from the catalog. There are two prerequisites for the catalog to contain configuration settings:

  - The originating server settings must be modified through the UI. Thus, only Log Settings, Ad Hoc Settings, OLAP Settings, and Cloud Settings are affected. For more information, see Configuration Settings in the User Interface

  - The catalog must be exported with the "everything" option or the specific **Server Settings** option.

    When server settings are imported, they take effect immediately and appear in the Settings UI.

- **Include themes** - This field determines whether you want to import themes from the catalog.

6. Click **Import**.

    The server uploads the catalog zip file and imports its contents into the repository. Depending on the size of the catalog, and the options you have selected, it may take several minutes to perform the import.

    Theme files in catalogs exported from older JasperReports Server versions may not be compatible, and can cause HTML display errors. When you import certain catalogs created in older versions, you may need to uncheck the **Include themes** option so that the incompatible files are not imported.

    If you have a custom theme to import, you can use the web UI to download it from the source server and upload it to the target server. You may need to edit the catalog manually. For example, if your theme contains the file pageSpecific.css, you must remove it from the ZIP file before uploading. Then redo changes to the file based on the pageSpecific.css file in the target server. For more information, see Downloading and Uploading Theme ZIP Files.

    Resources are imported along with any dependencies. You must have permission to write at the dependency's location in the organization. Broken dependencies may block the import operation. For more information, see Dependencies During Import and Export.

    Catalogs may be very large and take a long time to upload and then process. During this time, the import operation may affect server performance.

Import and Export Through the Command Line

> ⚠️ If you installed JasperReports Server from the binary installer, the command-line utilities are configured by the installer. If you installed the WAR file distribution, you must follow the instructions in Configuring Import-Export Utilities before you can run the utilities.

The import and export utilities are shell scripts located in the <js-install>/buildomatic folder:

Windows:	<js-install>/buildomatic/js-import.bat
	<js-install>/buildomatic/js-export.bat
Linux:	<js-install>/buildomatic/js-import.sh
	<js-install>/buildomatic/js-export.sh

The examples in this chapter use the shortened Windows commands without the optional .bat extension on the command line. If you are running JasperReports Server on Linux, be sure to add the .sh file extension.

When using the import and export utilities, keep the following in mind:

- JasperReports Server should be stopped when using the import and export utilities. This is very important for the import utility to avoid issues with caches, configuration, and security.

- All command-line options start with two dashes (--).

- You must specify either a directory or a zip file to export to or import from.

- The import and export commands include additional options to import and export keys between servers that need to share export catalogs. The options for sharing keys are documented in the JasperReports Server Security Guide. Once keys have been shared between servers, the commands in this chapter can be used without specifying keys.

- If you are exporting or importing organizations, you must be aware of resource dependencies outside of the organization that may block the operation. For more information, see Dependencies During Import and Export.

- Ensure the output location specified for an export is writable to the user running the command.

- All URIs are repository paths originating at the root (/). In commercial editions of the server, even those with a single default organization, the path must include the organization unless you specify the --organization option, for example:

/organizations/organization_1/reports/interactive/CustomersReport

> The import and export scripts provide access to the repository and internal database of the server. Even though all passwords are encrypted during export, a catalog may still contain sensitive URLs and data. You should set permissions on the host file system and operating system to secure the scripts and any catalogs you export.

Exporting from the Command Line

Usage: js-export [OPTIONS]

> We recommend you stop your server instance before running the export utility. For instructions see the JasperReports Server Installation Guide.

Use this command to export repository resources such as reports, images, dashboards, domains,, or entire folders to a catalog file. You can also export scheduled jobs, organizations, users, roles, and metadata such as repository access times and auditing events. The export output is known as a catalog. It is either a zip archive file or a set of files in a folder structure.

The `js-export` command includes additional options for exporting cryptographic keys. For more information about this special use case, see the JasperReports Server Security Guide.

Options in js-export Command

Option	Explanation
`--everything`	Exports everything except audit and monitoring data: all repository resources, permissions, report jobs, organizations, users, and roles. If any server settings have been modified in the UI, those are also included. May be combined with `--organization` to export everything in just one organization. This option is equivalent to: `--uris --repository-permissions --report-jobs --calendars --users --roles`
`--help`	Displays brief information about the available options.
`--include-access-events`	Exports repository events (date, time, and username of last modification).

Option	Explanation
`--output-dir`	Path of a location to export the catalog in a folder structure.
`--output-zip`	Path and filename to export the catalog as a zip file.
`--report-jobs`	Comma-separated list of repository report unit and folder URIs for which report unit jobs should be exported. For a folder URI, this option exports the scheduled jobs of all reports in the folder and all subfolders.
`--calendars`	When specified, the export includes all calendars of all types (holiday, recurring, ...) defined in the scheduler. When calendars are present in an export catalog, they are always processed and added on import.
`--uris`	Comma-separated list of folder or resource URIs to export from the repository. If the URI specifies a folder, the export operation exports all resources and folders contained in the folder. In addition, it recurses through all its subfolders.
`--resource-types`	Comma-separated list of resource types to export. The available resource types are: adhocDataView, awsDataSource, beanDataSource, customDataSource, dashboard, dataType, file, folder, inputControl, jdbcDataSource, jndiJdbcDataSource, listOfValues, mondrianConnection, mondrianXmlaDefinition, olapUnit, query, reportUnit, virtualDataSource, xmlaConnection.
`--repository-permissions`	This option exports repository permissions with each exported resource or folder. This option should only be used with --uris.
`--skip-dependent-resources`	Exports resources without their dependencies, for example a report without its external data source, input control definitions, or image files.
`--organization`	Specify the ID of an organization to export. When specified, only resources, users, and roles from this organization (and

Option	Explanation
	its suborganizations) are exported. When specified, all URIs are relative to this organization.
`--skip-suborganizations`	When used with `--organization`, specifies that suborganizations (and their resources, users, and roles) should not be exported.
`--roles`	Comma-separated list of roles to export. If no roles are specified with this option, all roles are exported.
`--role-users`	Use only with `--roles`. This option exports all users belonging to each exported role.
`--users`	Comma-separated list of users to export. If no users are specified with this option, all users are exported. Exporting a user includes all user attributes and all roles assigned to each user. When specifying users, you must give their organization ID if applicable, for example: `--users superuser, "jasperadmin|organization_1"`
`--users-roles`	Use only with `--users`. This option exports all roles belonging to each exported user.
`--include-attributes`	Specify this flag to export attributes on any user, organization, or root level that is exported.
`--skip-attribute-values`	When used with `--include-attributes`, specifies that only attribute names are exported, values are null.
`--include-audit-events`	Includes audit data for all resources and users in the export.
`--include-monitoring-events`	Includes monitoring data for all resources and users in the export.
`--include-server-settings`	Includes persistent server settings, such as Log Settings, Ad Hoc Settings, Ad Hoc Cache, and OLAP Settings.

Option	Explanation
`--report-alerts`	Comma separated list of repository report unit and folder URIs for which report unit alerts should be exported. For a folder URI, this option exports the alerts of all reports in the folder and all subfolders.

> User passwords are encrypted during the export by default, but exported catalogs may contain sensitive data. Take appropriate measures to secure the catalog file from unauthorized access.

Examples:

- Export everything in the repository:

```
js-export --everything --output-dir myExport
```

- Export the /reports/interactive/CustomersReport report unit to a catalog folder:

```
js-export --uris /organizations/organization_
1/reports/interactive/CustomersReport --output-dir myExport
```

- Export the /images and /reports folders:

```
js-export --uris /organizations/organization_1/images
/organizations/organization_1/reports --output-dir myExport
```

- Export all resources (except users, roles, and job schedules) and their permissions to a zip catalog:

```
js-export --uris / --repository-permissions --output-zip myExport.zip
```

- Export all resources and report jobs:

```
js-export --uris / --report-jobs / --output-dir myExport
```

- Export the report jobs of the /reports/interactive/CustomersReport report unit:

```
js-export --report-jobs /organizations/organization_
1/reports/interactive/CustomersReport --output-dir myExport
```

- Export all roles and users:

  ```
  js-export --roles --users --output-dir myExport
  ```

- Export the ROLE_USER and ROLE_ADMINISTRATOR roles along with all users belonging to either role:

  ```
  js-export --roles ROLE_USER, ROLE_ADMINISTRATOR --role-users --output-dir
  myExport
  ```

- Export all resources in an organization, but not its suborganizations:

  ```
  js-export --organization organization_1 --skip-suborganizations --uris /
  ```

> The folder named Temp at the root and in every organization is a special folder. None of the folders or resources in a Temp folder are exported.

Importing from the Command Line

See Import and Export Through the Command Line for guidelines when running the command-line utilities.

> When using the js-import command-line utility, the server must be stopped to avoid issues with caches, configuration, and security. For instructions see the JasperReports Server Installation Guide.

Usage: `js-import [OPTIONS]`

Use this command to read the catalog from your file system and create the resources in the JasperReports Server repository. The import command can also create entities such as organizations, users, roles, and attributes. The catalog must be one created by the export interface or the `js-export` command, either as a ZIP archive file or a folder structure.

Exported catalogs may contain encrypted passwords. If you are importing to a different server, you must configure an encryption key on both servers. See The Import-Export Encryption Keys for details.

The js-import command includes additional options for importing cryptographic keys. For more information about this special use case, see the JasperReports Server Security Guide.

Options in js-import Command

Option	Explanation
--help	Displays brief information about the available options.
--input-dir	Path for importing a catalog from a directory.
--input-zip	Path and filename for importing a catalog from a zip file.
--update	Resources in the catalog replace those in the repository if their URIs and types match.
--skip-user-update	When used with --update, users in the catalog are not imported or updated. Use this option to import catalogs without overwriting currently defined users.
--organization	If the import catalog is from an organization, it specifies the target organization where it should be imported. The organization ID should match the ID of the organization in the import catalog, if not, use the --merge-organizaiton option.
--merge-organization	Use this option when --organization is specified but it does not match the ID of the organization in the import catalog. When merging organizations, the contents of the import override the target organization for any user, role, or resource with the same name. A merged organization takes the organization ID of the imported organization.
--brokenDependencies	Specifies the action to take when importing a resource with a broken dependency. One of the following values: • skip – Does not import the resource with the broken dependency, but continues to import other resources. • include – Attempts to import the resource with the broken dependency. The import succeeds if there is already a resource in the destination that satisfies the dependency. If the dependency is not satisfied in the destination, the resource is skipped and the import continues. • cancel – Stops the import operation.

Option	Explanation
`--include-access-events`	Restores access events (date, time, and username of last modification) on imported resources.
`--include-audit-events`	Imports any audit data in the catalog.
`--include-monitoring-events`	Imports any monitoring data in the catalog.
`--include-server-settings`	Determines whether the system configuration is updated from the catalog. There are two prerequisites for the catalog to contain configuration settings: • The originating server settings must be modified through the UI (Log Settings, Ad Hoc Settings, Ad Hoc Cache, and OLAP Settings). For more information, see Configuration Settings in the User Interface. • The catalog must be exported with the "everything'" option from the user interface or the command-line utility. Imported server settings take effect when the server is started.
`--skip-themes`	This flag is required when importing a catalog that includes a theme from some Release 5 server versions. If you need to import a custom theme, use the Theme UI to download it from the source server and upload it to the target server. In some cases, you may need to take more extensive steps. For more information, see Downloading and Uploading Theme ZIP Files.
`--include-alerts`	Includes data alert when importing the report.

Examples:

• Import the myExport.zip catalog archive file:

```
js-import --input-zip myExport.zip
```

- Import the myDir catalog folder, replacing existing resources if their URIs and types match those found in the catalog:

```
js-import --input-dir myDir --update
```

- Import the myExport.zip catalog archive file but ignore any users found in the catalog:

```
js-import --input-zip myExport.zip --update --skip-user-update
```

- Import the myDir catalog folder with access events:

```
js-import --input-dir myDir --include-access-events
```

When a resource in the target repository has the same URI as on that you are importing, the default behavior is left as it is and the existing resource remains unchanged (no overwriting occurs).

To delete the existing resource and replace it with a new one (of the same type and with the same URI), use the --update option. Note that, if the resource in the export catalog is a different type than the existing resource, the server returns an error and skips the update operation.

When you import a user whose roles exist in the repository, the user is given those roles. User properties are imported with the user.

When you import access events, the date and time of the last modification before export is restored on import for every resource. The catalog folder has to be created with access events. If you do not import access events, or if they do not exist in the imported files, then the date and time of the import are used.

Configuring Import-Export Utilities

If you installed JasperReports Server from the binary installer, the import-export utilities are configured by the installer. If you installed the WAR file distribution, you must configure several files before you can use the import-export utilities.

Alternatively, see Alternate Import-Export Scripts because the alternate scripts do not require any configuration, regardless of the installation method.

To configure the import-export utilities

Procedure

1. Depending on the database you use, copy the installation configuration file:

   from: <js-install>/buildomatic/sample_conf/<database>_master.properties

   to: <js-install>/buildomatic/default_master.properties

2. Edit the default_master.properties file to set values specific to your installation. For more information about the settings in this file, see the JasperReports Server Installation Guide.

   > Oracle users can set the sysUsername and sysPassword to the same name as dbUsername and dbPassword in the default_master.properties. The system username and password are not required because js-import and js-export do not make changes to the database schema.

3. Run the following command:

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

   This command generates the following files with the values that you added to the default_master.properties file:

   - <js-install>/buildomatic/build_conf/default/js.jdbc.properties
   - <js-install>/buildomatic/build_conf/default/js.quartz.properties (only for DB2 and PostgreSQL)

4. Make sure that the JDBC driver for your database is located in the following folder:

   <js-install>buildomatic/conf_source/iePro/lib

   If necessary, you can find links for downloading JDBC drivers from the Jaspersoft Community website:

   http://community.jaspersoft.com/wiki/downloading-and-installing-database-drivers

Alternate Import-Export Scripts

Regardless of your installation method, JasperReports Server provides a third way to run import-export commands. Buildomatic is another command-line script that is based on the Apache Ant tool to automate installations. It includes targets (sub-commands) to perform

import and export operations with the same options as the scripts. The following examples compare the two commands:

Shell Scripts: `js-export.sh --everything --output-zip=js-catalog-exp.zip`

Buildomatic: `js-ant export-everything -DexportFile=js-catalog-exp.zip`

Both types of scripts are located in the <js-install>/buildomatic folder.

Running Import from Buildomatic

The import target for ant has the following syntax:

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

Linux and Mac OSX: `./js-ant import -DimportFile=<filename> [-DimportArgs="<import-options>"]`

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

> When performing a large import using js-ant, the server should be stopped (or put into a mode with reduced load) to avoid issues with caches, configuration, and security.

The following examples are typical import commands on Windows:

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

The following examples are typical import commands on Linux:

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

Running Export from Buildomatic

The export target for ant has the following syntax:

Windows:	`js-ant export -DexportFile=<filename> -DexportArgs="<export-options>"`
Linux and Mac OSX:	`./js-ant export -DexportFile=<filename> -DexportArgs="<export-options>"`

The export file format is a ZIP file or a set of files under a new directory name. If you specify the .zip extension for your output filename, a ZIP archive is created automatically. Otherwise, an uncompressed directory with files and sub-directories is created.

The following examples are typical export commands on Linux and Windows:

```
js-ant export-help-pro
js-ant export -DexportFile=my-domains.zip
    -DexportArgs="--uris /organizations/organization_1/datasources"
js-ant export -DexportFile=my-reports-and-users.zip
    -DexportArgs="--uris /organizations/organization_1/reports
    --users jasperadmin|organization_1,joeuser|organization_1"
js-ant export -DexportFile=my-datasources
    -DexportArgs="--uris /organizations/organization_1/datasources --
roles ROLE_USER"
js-ant export -DexportFile=js-everything.zip -DexportArgs="--everything"
```

System Configuration

You can change the default behavior of JasperReports Server by editing the system's configuration. The configuration is defined by a set of properties and their values.

The properties are stored in configuration files located in various folders under the <js-install> directory, which is the root of your JasperReports Server installation. To change the configuration, you edit these files and then restart the server.

A few of the most commonly edited properties are available to the system admin through the user interface (UI). Changes to these properties take effect immediately, are stored in the repository, and override the equivalent values stored in files, even after the server restarts

This chapter describes a subset of the properties in the configuration files. Settings that affect security are covered in *JasperReports Server Security Guide*. Configuration of the auditing and monitoring features is covered in Configuring Auditing and Monitoring. More options are described in the JasperReports Server Installation Guide.

Because the locations of files described in this chapter vary with your application server, the paths specified here are relative to the deployed WAR file for the application. For example, the applicationContext.xml file is shown as residing in the WEB-INF folder; if you use the Tomcat application server bundled with the installer, the default path to this location is:

> C:\Program Files\jasperreports-server-8.2.0\apache-tomcat\webapps\jasperserver-pro\WEB-INF

> ⚠ Use caution when editing the properties described in this chapter. Inadvertent changes may cause unexpected errors throughout JasperReports Server that may be difficult to troubleshoot. Before changing any files, back them up to a location outside of your JasperReports Server installation.
>
> Do not modify settings that are not described in the documentation. Even though some settings may appear straightforward, values other than the default may not work properly and can cause errors.

This chapter contains the following sections:

- Configuration Settings in the User Interface

- Configuration for Using Proxies

- Configuration for Session Persistence

- [Enabling Compression in Tomcat](#)

- [Configuring Ad Hoc](#)

- [Enabling Data Snapshots](#)

- [Enabling Data Staging](#)

- [Configuring Cloud Services](#)

- [Configuring Domains](#)

- [Configuring JasperReports Library](#)

- [Disabling Open In Editor Option](#)

- [Configuring Input Control Behavior](#)

- [Configuring the Scheduler](#)

- [Configuring Report Thumbnails](#)

- [Configuring the Heartbeat](#)

- [Configuring the Online Help](#)

- [Configuring JasperReports Web Studio Access](#)

Configuration Settings in the User Interface

Configuration settings are persistent through server restart, though this wasn't always the case. Some previous versions would reload configuration settings from the server's configuration files.

To make persistent configuration changes through the JasperReports Server user interface

Procedure

1. Log in as system administrator (superuser by default).

2. Select Manage > Server Settings:
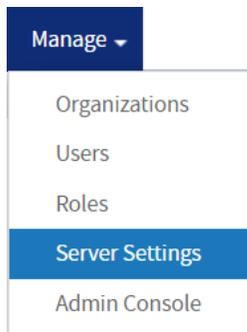
3. Choose a category of settings or administrator actions from the left-hand Settings panel.
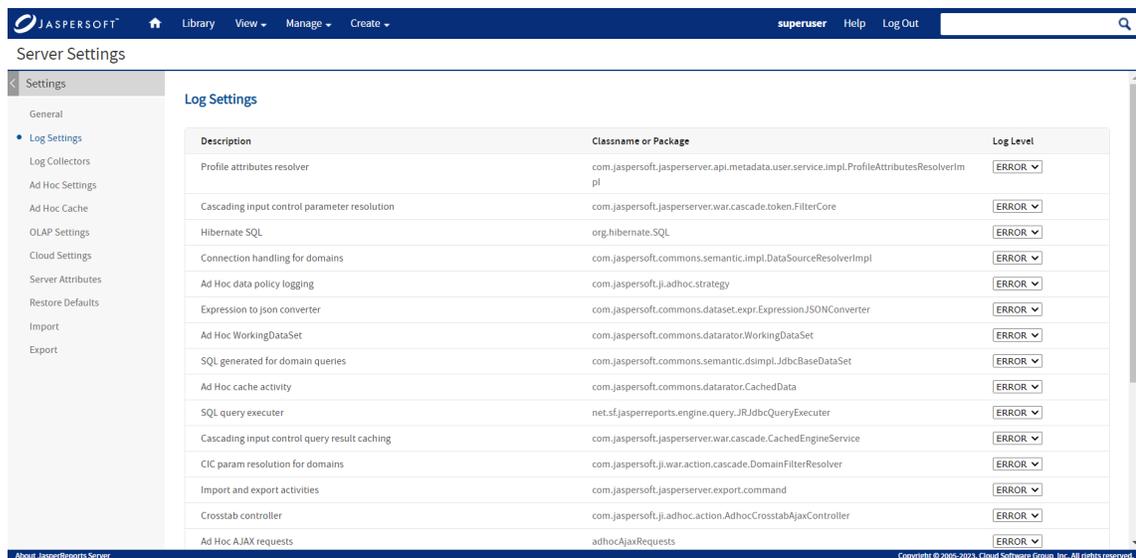
Figure 90: The User Interface for Configuration Settings

4. Find the configuration setting you want to change and edit its value. In the case of log levels, the new value takes effect immediately. In the case of other settings, click Change beside the individual setting.

The settings and administrator actions are documented in their respective sections:

Settings	Documentation
General	Enabling JDBC Driver Uploads
Log Settings	Configuring System Logs

Settings	Documentation
Log Collectors	Using Log Collectors
Ad Hoc Settings	Ad Hoc Settings
Ad Hoc Cache	Ad Hoc Cache Management
OLAP Settings	*Jaspersoft OLAP User Guide*
Cloud Settings	Configuring Cloud Services
Server Attributes	Managing Attributes
Restore Defaults	Restoring Default Settings
Import	Importing from the Settings
Export	Exporting from the Settings

Understanding Persistent Settings

When making changes through the Settings UI, you should understand how persistent settings made in the UI are stored internally and relate to the settings in configuration files:

- The Settings pages display a subset of the settings available in configuration files. Therefore, all settings in the UI also exist in a configuration file.

- By default, the Settings pages display the values of settings that exist in the corresponding configuration file. If you modify only the files and restart the server, your new file settings take effect on the server and are visible in the UI.

- When you change a value on the Settings pages, the new setting takes effect immediately, but the new value is not written to the corresponding configuration file. Instead, it's stored in the server's internal database so the value is persistent when the server is restarted.

> Only configuration settings that have a value modified on the Settings pages of the UI are stored and made persistent in the database.

- When the server restarts, any stored values take precedence over values for the same settings in the configuration files. However, each setting is independent, so a value that's not modified in the Settings UI is read from the configuration files.

- The Settings pages display the values of the settings in effect on the server.

> Be aware that the configuration values that appear on the Settings pages are possibly a mixture of values loaded from configuration files and from the persistent storage.

- Changing a setting that has already been modified updates its value stored internally, even if it is set to the same original value stored in a configuration file. The stored value continues to take precedence over any changes to this setting in the configuration file.

Restoring Default Settings

If a setting has been modified in the UI, it will remain in persistent storage and always override the corresponding setting in a configuration file. If you want to reset a value so that it is read from the configuration file instead, use the Restore Defaults page.

To restore a default setting

Procedure
1. Log in as system administrator (superuser by default).

2. Select Manage > Server Settings and choose Restore Defaults from the left-hand panel.

Figure 91: The Restore Defaults Page Containing Persistent Configuration Settings

The configuration values on the Restore Defaults page represent the settings that have been modified through the UI and are stored in persistent storage.

> The Restore Defaults page also includes JDBC drivers configured during the installation or through the data source creation wizard. Do not remove the drivers with [SYSTEM] values. For more information, see Managing JDBC Drivers.

3. To restore a setting to its configuration file default, click the 🗑 icon beside its current value and confirm.

4. Click Save to make the change permanent.
The setting is removed from persistent storage and the value of the setting is restored to its default value from the corresponding configuration file. The next time the server restarts, its value will be read from the configuration file.

> The settings listed on the Restore Defaults page are those that can be exported to different servers or re-imported after a server upgrade. For more information, see Importing from the Settings.

Configuration for Using Proxies

JasperReports Server exposes some URLs to itself through the UI and when using the scheduler to send emails. When using a proxy, those URLs must be rewritten so that they expose the proxy URL, not actual URL of the server instance. This can be done on the proxy or using JasperReports Server settings.

Configure your proxy or load balancer to set X-Forwarded-* headers (X-Forwarded-For, X-Forwarded-Proto, X-Forwarded-Host, X-Forwarded-Port). For the configuration details, refer to the documentation of the proxy or load balancer that you use.

If you are using multiple proxies, X-Forwarded-* headers on all intermediate proxies should be set to the address of the proxy receiving requests from the user's UI.

> When using a proxy, make sure that the proxy is set up to allow special symbols in URLs. In Apache httpd, this option is called AllowEncodedSlashes On. See the documentation for your application server for more information.

Change the following setting so that JasperReports Server exposes the proxy URL for scheduled reports.

Configuration for Using Proxies for Scheduled Reports	
Configuration File	
.../WEB-INF/js.quartz.properties	
Property	**Description**
report.scheduler.web.deployment.uri	This is the base URL used by the scheduler to generate links to reports in emails. Set this property to the full URL, including application name, that you expose through your proxy. For example: `http://bi.example.com/jasperserver` Specify http or https in your URL.

Configuration for Session Persistence

Many application servers can store user sessions while a web app is offline, for example when changing JasperReports Server configuration files. The app server remembers all the information about a user's session, such as the session ID and what page was being viewed. When the web app restarts, the user session is restored. The user does not have to log in again, and often may not even notice that the server was temporarily unavailable. This is called session persistence.

> 📝 Sessions do not persist when redeploying a web app in the application server, only when restarting the web app.

JasperReports Server supports a limited form of session persistence. When session persistence is enabled in the app server, sessions can be restored in the following cases:

- Browsing the repository, expanding folders in the repository tree, and viewing folder contents.
- Searching the repository, including all search filters and results.
- Repository permissions dialog, including state and selections.
- Add folder dialog.
- Add resource dialogs, including adding or editing a data source, JasperReport, and other repository objects.
- Copy, cut, and paste resources in the repository.
- Scheduling a report, including all information such as a schedule and notifications.

If the server becomes unavailable when using the pages or dialogs above, the user can see the pause only when performing an action on these pages, such as submitting. When the server has finished restarting, the user can continue interacting with these UI elements. If the user takes no action while the server is unavailable, they may not even notice the server restart.

However, for other interactive dialogs in JasperReports Server, the state is too large to store in the user session. These features do not support session persistence and unsaved data can be lost:

- Ad Hoc Editor - The state of the report layout and data in the Ad Hoc Editor cannot be restored.
- Dashboard designer - The contents and state of the canvas cannot be restored.

- Interactive report viewer - The data in the report, as well as the state of column sorting and filtering cannot be saved.

- Domain designer - Any tables, joins, filters, calculated fields, and display names cannot be restored.

- OLAP viewer - The data in the OLAP view and current MDX expression cannot be restored.

- Administration dialogs - When creating or editing users, organizations, or roles, the information entered in a dialog cannot be restored if it was not submitted.

In the cases listed above, the user's work is interrupted, and any unsaved work is lost. However, when the server restarts, the user does not have to log in again, the server displays a message about the session that could not be fully restored, and redirects the user to the home page. The user must relaunch the interactive feature and recreate any unsaved work.

Session persistence also affects web service calls. The REST API supports a login to store a session ID, and with persistence enabled, that session ID will still be valid when the application server restarts. This simplifies the code when you need to handle timeouts. In general, web service calls do not support interactive work like designing an Ad Hoc view, a dashboard, a Domain, or exploring data in OLAP, so they are not affected by the lack of session persistence in most cases. However, web service calls are affected in the following case:

- Report execution - All asynchronous API calls for running and exporting reports rely on the large JasperPrint object that cannot be persisted. When the server restarts, the asynchronous calls return errors because the reports could not be saved in the session. Your application needs to detect this error and include code for rerunning the report.

> 📝 JasperReports Server also supports session replication among multiple instances of the server in a cluster. However, session replication has the same limitations because it is based on session persistence. For more information, see the JasperReports Server Ultimate Guide.

The following procedure gives the configuration in JasperReports Server and the Apache Tomcat application server to enable session persistence. For another application server, refer to that server's documentation.

To configure JasperReports Server and Apache Tomcat for session persistence

Procedure

1. Edit the file .../META-INF/context.xml to comment out the Manager property as follows:

   ```
   <!-- Manager pathname="" /-->
   ```

2. Edit the file .../WEB-INF/web.xml to make the following changes.

   a. Locate the ClusterFilter given in the comments and uncomment it as follows:

   ```
   <filter>
           <filter-name>ClusterFilter</filter-name>
           <filter-
   class>com.jaspersoft.jasperserver.war.TolerantSessionFilter</filter-class>
       </filter>
   ```

   b. Locate the corresponding mapping for the ClusterFilter and uncomment that, too. Also uncomment the <distributable> element below it as follows:

   ```
   <filter-mapping>
       <filter-name>ClusterFilter</filter-name>
       <url-pattern>/*</url-pattern>
   </filter-mapping>
   <distributable/>
   ```

3. Add the following property to your JVM environment:

   ```
   -Dorg.apache.catalina.session.StandardSession.ACTIVITY_CHECK=true
   ```

4. Restart your Apache Tomcat application server.

Enabling Compression in Tomcat

The Tomcat app server can be configured to compress web content sent to browsers in order to reduce bandwidth usage and reduce loading times. Tomcat compression has been tested with JasperReports Server and found to reduce transferred data to about one third of the original size. For example, the login page and its scripts and images are about 3 MB normally and 1 MB compressed; the Domain designer page and scripts are 6.2 MB normally and 1.7 MB compressed. Large reports with lots of data may compress even more.

The performance benefit of compression depends upon the content and your client's connection. For small pages on a local network, the data transfer time is minimal, and the overhead of decompression may actually cause pages to load a few milliseconds longer. For large pages on a local network, compression can help load a few milliseconds faster. However, compression can improve loading times noticeably on networks with low latency or bandwidth, for example if you access reports on the server from a mobile device. In this case, loading time can be reduced by seconds, for example from 10 to 5 seconds.

Compression can also be enabled on cloud or hosted services to reduce bandwidth usage and costs.

Compression is a configuration on the Tomcat app server that is off by default. To enable compression, shut down your Tomcat instance and modify the following file:

Compression in Tomcat

Configuration File

.../apache-tomcat/conf/server.xml

Property	Description

<pre><Connector
compressibleMimeType="text/html,text/xml,text/css,
 	
text/javascript,application/javascript,application/json"
compression="on"
compressionMinSize="128"
connectionTimeout="20000"
noCompressionUserAgents="gozilla, traviata"
port="8080"
redirectPort="8443" protocol="HTTP/1.1" /></pre>
	Add this connector to the Catalina service. You can modify the settings as needed, but these default values have been tested to work.

Restart the Tomcat app server after saving the file.

Compression is not compatible with the use of sendfile to reduce processor load in Tomcat. If you specify the useSendFile parameter, it takes precedence and content will not be compressed. For more information, see the Tomcat documention.

Configuring Ad Hoc

Ad Hoc settings help you fine-tune the Ad Hoc Editor and how users interact with Ad Hoc views, in particlar the cache settings that may affect performance for large data sets. Users with ROLE_SUPERUSER can manage the Ad Hoc settings and cache through the UI. Less used settings are located in configuration files.

When opening Ad Hoc views or reprts that are based on Domains, data structures are verified against those in the Domain and may cause errors if the view or report reference items that are no longer in the Domain. To modify this behavior, see Setting the Level of Referential Integrity.

This section covers the following ways to configure Ad Hoc:

- Ad Hoc Settings

- Ad Hoc Data Policies

- Ad Hoc Data Policies for Big Data

- Ad Hoc Templates and Report Generators

- Ad Hoc Configuration File

- Ad Hoc Cache Management

- Ad Hoc OLAP Filter Configuration

Ad Hoc Settings

> The Ad Hoc settings apply to Ad Hoc views based on Topics or Domains. Ad Hoc views based on OLAP connections use the OLAP settings described in the Jaspersoft OLAP User Guide.

The Ad Hoc settings include the following:

- General Settings to modify the Ad Hoc Editor user interface:

  - Show Duplicate Table Rows. It sets the default behavior for Show duplicate rows in the Ad Hoc Editor. Users with ROLE_SUPERUSER can edit the Ad Hoc settings. This setting is applied by default when the Ad Hoc view is created. If a user logs in and changes the default setting of Show duplicate rows in the Ad Hoc Designer > Format Visualization > Appearance, then the server setting gets

overridden for a particular Ad Hoc view.

○ Configure View Query. Determines whether users can see a button in the Ad Hoc Editor to display the SQL or MDX query generated for the view. This can be useful for advanced users, but you should consider your data security before enabling this. System admins can always view queries in the Ad Hoc Cache (see Ad Hoc Cache Management).

○ Display Null as Zero. Determines the appearance of null values in crosstabs and time-series charts. By default, this setting is disabled and null values are displayed as empty cells in crosstabs or missing points in time series charts (thus causing irregular intervals). When enabled, null values are displayed as zero in crosstabs and in time-series (creating regular intervals). Regardless of this setting, null values are always displayed as zero in all other chart types.

○ Always show input controls. By default, this option is not enabled. When this option is selected, it displays the Input Controls dialog to prompt the user when the Ad Hoc report runs. Input Controls dialog appears only for the Ad Hoc reports created when the option is selected, while Ad Hoc reports created when the option was disabled, do not get affected. When this option is not selected, the Input Controls dialog still appears for Ad Hoc reports created when this option was enabled.

○ Default Ad Hoc Visualization Type. Sets the default visualization type when a new Ad Hoc view is created. A superuser can select the default visualization type from the drop-down list.

• Query Limits to preserve resources used by queries when Ad Hoc views are designed and run:

○ Ad Hoc Filter List of Values Row Limit. The maximum number of items that should be displayed in the Condition Editor when a user defines filters for an Ad Hoc view based on a Domain. If this limit is exceeded when users define filters, JasperReports Server displays a message to that effect. Setting this to a lower value can improve performance.

> If the input control has more options than the limit given in this setting, the **Equals** input control changes into a free input field in the Ad Hoc report. Therefore, the field does not display the options to choose from and the user has to enter the option manually.

○ Ad Hoc Dataset Row Limit. The maximum number of rows that an Ad Hoc view will request in a query. Be aware that JasperReports Server truncates the data

      displayed in the report when the limit is reached. Setting this to a lower number may improve performance, but your reports may not reflect the full data set.

- Ad Hoc Query Timeout. The number of seconds the server should wait before timing out an Ad Hoc view while running its query. Setting this to a lower number may prevent exceptions when users run Ad Hoc views. Setting this to a higher number may prevent complex calculations from timing out, but may use more database connections.

- Data Policies that determine how JasperReports Server handles data loading and processing for certain kinds of Ad Hoc views. See Ad Hoc Data Policies in the next section.

To configure the Ad Hoc query settings

Procedure

1. Log in as system administrator (superuser by default).

2. Select Manage > Server Settings and choose Ad Hoc Settings in the left-panel.

3. Set the configuration values as described above.

4. Click Change beside each value you modified to make your changes effective. Or click Cancel to reset it to the previously saved value.

Ad Hoc Data Policies

> Data snapshots, described in Data Snapshots and Enabling Data Snapshots, apply only to reports displayed in the report viewer. This section covers data policies that apply only to views in the Ad Hoc Editor.

Data policies determine how data is cached and where certain calculations occur. All Ad Hoc work is based on a query, either from a Domain or Domain Topic, or from the JRXML of a plain Topic. Data policies determine whether the Ad Hoc engine uses the query as-is and processes the data in memory, or rewrites the query so that the database processes data and returns only what Ad Hoc needs to display.

By default, the data accessed by Domain-based reports is grouped, sorted, and aggregated in the database, rather than processed in memory on the server. This way the server retrieves only the columns that appear in the report rather than the entire set of fields in

the Domain. Calculated fields are also processed by the database. For Topic queries based on JDBC (and JNDI) data sources, the default behavior is to request the entire result set and process the columns for display in memory. Note that independent check boxes control the behavior, one check box for Domains and another for JDBC data sources.

When these check boxes are cleared, the server loads the entire set of fields associated with a Domain or Topic into memory, then applies the necessary calculations, grouping, sorting, and aggregation. This is also the case for Ad Hoc views that do not rely on Domains or JDBC data sources; in these cases, the server processes the data in memory.

Generally, we recommend that these settings be enabled, especially when working with large datasets. In deciding whether JasperReports Server should process the data in memory or push that processing to the database, consider these factors:

- The size and complexity of your reports. Reports with calculated fields, complex sorting, grouping, or aggregation may perform better when the server optimizes the queries so that the database performs the work.

- The amount of data in your data sources. If your data sources include a great deal of data, reports against them may perform better when the server optimizes the queries.

- The number of users editing and running Ad Hoc views. If you have a large number of users creating and running Ad Hoc views, performance may be better when the server optimizes the queries. Implementations with fewer users may perform better when the options are disabled.

- The performance characteristics of your data source. If the database or other data source is tuned for maximum performance, Ad Hoc views may perform better when the server optimizes the queries.

- If your data source is hosted by MySQL, we recommend that you keep the default (unchecked) for the Optimize Queries for JDBC-based Reports setting. MySQL has poor performance with the nested queries that this setting would generate.

- The amount of memory allocated to JasperReports Server's Java Virtual Machine (JVM). If the JVM of the application server hosting JasperReports Server is allocated plenty of memory, Ad Hoc views may perform better when JasperReports Server optimizes the queries. This is especially true if your data source tends to be slow.

To decide whether JasperReports Server should optimize queries for Ad Hoc views, we recommend disabling the settings, opening and saving some representative reports, and testing their performance. If the performance improves, leave the settings disabled and open and save the remaining reports.

The data polices you can set are:

- Optimize Queries for JDBC-based Reports. When checked, Ad Hoc rewrites the query to calculate, filter, group, sort, and aggregate columns when using Topics based on JDBC and JNDI data sources. Otherwise, the queries run unaltered and calculated fields, filtering, grouping, sorting, and aggregation take place in memory.

- Optimize Queries for Domain-based Reports. When checked, Ad Hoc rewrites the query to calculate filter, group, sort, and aggregate columns when using Domains or Domain Topics. Otherwise, the queries run unaltered and calculated fields, filtering, grouping, sorting, and aggregation take place in memory.

> These data policy settings do not retroactively update the existing reports created from Ad Hoc views in your repository. To change the data policy for an existing report, select the appropriate policy setting, open the corresponding view in the Ad Hoc Editor, and save the report again.

To configure the Ad data policies

Procedure

1. Log in as system administrator (superuser by default).

2. Select Manage > Server Settings and choose Ad Hoc Settings in the left-hand panel.

3. Select Optimize Queries for JDBC-based Reports to optimize and rewrite queries for JDBC-based Topics.

4. Select Optimize Queries for Domain-based Reports to optimize and rewrite queries for Domain-based reports.

5. Click Change beside each value you modified to make your changes effective. Or click Cancel to reset it to the previously saved value.

Ad Hoc Data Policies for Big Data

When handling large datasets (big data) from a Domain source in the Ad Hoc Editor, fields summarized by distinct count are computationally intensive. The server optimizes the distinct count by requesting distinct count calculations from the data source, as opposed to performing the calculations in memory. Database servers are usually optimized for these calculations, which improves the overall performance of the Ad Hoc Editor.

Ad Hoc for Big Data		
Configuration File		
.../WEB-INF/applicationContext-adhoc-dataStrategy.xml		
Property	Bean	Description
calcMethod	Distinct Count	This property's default value is sqlUnionAll. The UnionAll is the optimized query that provides distinct count computed by the database. If this setting causes issues with your database, you can reset its value to sqlGroupBy. If you make this change, redeploy the web app or restart the application server.

Performing distinct count aggregates in the database applies only in the following cases:

- Crosstabs based on Domains contain measures aggregated by distinct count.

- Tables based on Domains contain groups aggregated by distinct count, but no detail rows.

This setting has no effect when there is a row or column group involving a time, timestamp, or date. In this case, Ad Hoc performs the distinct count summary calculations in memory, regardless of the calcMethod setting.

Ad Hoc Templates and Report Generators

Ad Hoc templates are JRXML files in the repository that define the format for reports generated from Ad Hoc views. Report generators are custom beans that create custom output from Ad Hoc views.

In the user interface, users can select either the default template, browse for a different template, or select a report generator, if any are defined. For more information about the report template interface, see the JasperReports Server User Guide. This section explains how to configure these controls.

Changing the Default Ad Hoc Template

The default Ad Hoc template creates a report that resembles the Ad Hoc user interface. It's meant for online viewing and doesn't restrict the size of the contents for printing. To set a different template, modify the following property:

Default Ad Hoc Template		
Configuration File		
.../WEB-INF/applicationContext-adhoc.xml		
Property	Bean	Description
defaultTemplateUri	util:map id="report Generator DefaultConf"	Specify the repository path of your new default template. Make sure the template has permissions so it's accessible to all users. The default value is /public/templates/actual_ size.510.jrxml.

Modifying the Ad Hoc Template Folders

Ad Hoc templates must be uploaded to specific folders in the repository. The default folders are /templates (in every organization) and /public/templates. To set different folders, modify the following property:

Ad Hoc Template Folders		
Configuration File		
.../WEB-INF/applicationContext-adhoc.xml		
Property	Bean	Description
templateURIParent SQLPatternList	adhocEngine Service	Add, change, or remove the values in this list to specify the folders in the repository where the server looks for Ad Hoc templates.

The default Ad Hoc template locations contain the default template. If you move or delete those folders in the repository, be sure to update the definition of the default Ad Hoc template as described in the previous section.

Also, organization templates include the adhoc/templates folder so it appears in every new organization. You should update all organization templates if you change or remove the /templates value. For more information, see Default Folders for Organizations.

Adding Ad Hoc Report Generators

By default no custom report generators defined on the server, and none appear in the user interface.

To add a custom report generator

Procedure

1. Create a Java class that implements the com.jaspersoft.ji.adhoc.service.AdhocReportGenerator interface.

2. Compile the class and place the resulting JAR file in .../WEB-INF/lib.

3. Open the file .../WEB-INF/applicationContext-adhoc.xml for editing and register your class as a Spring bean as shown in the following example:

```
<bean id="myCustomReportGenerator" class="com.example.myCustomReportGenerator">
    <property name="id" value="my-custom-generator"/>
    <property name="..." value="..."/>
    ...
</bean>
```

4. In the same file, update the reportGeneratorFactory bean to include your custom generator bean:

```
<bean id="reportGeneratorFactory"
class="com.jaspersoft.ji.adhoc.service.ReportGeneratorFactoryImpl">
    <property name="reportGenerators">
        <list>
            <ref bean="myCustomReportGenerator" />
            <!--<ref bean="actualSizeReportGenerator" />-->
            <!--<ref bean="letterPortraitReportGenerator" />-->
            <!--<ref bean="letterLandscapeReportGenerator" />-->
            <!--<ref bean="a4PortraitReportGenerator" />-->
            <!--<ref bean="a4LandscapeReportGenerator" />-->
        </list>
    </property>
</bean>
```

5. Edit the .../WEB-INF/bundles/adhoc_messages.properties file to add a UI label for your custom generator. The key has the form ADH_REPORT_GENERATOR_<generator-id>. Add the same key to other language bundles if you want to support other languages.

```
ADH_REPORT_GENERATOR_my-custom-generator=Corporate Template
```

6. Restart the server or redeploy the JasperReports Server web app. The label for your custom generator appears in the list of report generators when users create and save a report from an Ad Hoc view.

Ad Hoc Configuration File

The following properties are among those that can be configured in the WEB-INF/applicationContext-adhoc.xml file:

Configurable Properties in WEB-INF/applicationContext-adhoc.xml

Property	Description
JrxmlScriptURI	The location in the file system of the state2jrxml.js script, which generates the JRXML report based on the current Ad Hoc Editor selections. By default, this file is located in the /adhoc folder of the repository.
realmsURI and realmsURIParent SQLPatternList	The repository locations where Topics should reside. The defaults are /adhoc/topics and /public/adhoc/topics.

Property	Description
defaultTheme	The name of the default style for Ad Hoc views. This name must match a style defined in both a CSS and a JRXML file. The default is default.
aruFolder	The repository location where users are allowed to save their Ad Hoc views. The default is /. This allows your users to save Ad Hoc views anywhere. If you have a folder specifically for user content, specify this folder, for example /userviews.
tempFolderName	The repository location where JasperReports Server saves reports created from Ad Hoc views. The default is /temp relative to root and to every organization. The server allows users with ROLE_ADMINISTRATOR or ROLE_SUPERUSER to view the temporary folders and their contents. The server manages these temporary files automatically, but files may accumulate in certain cases. As part of regular maintenance, you should periodically delete the files in these folders.
maxSafeGroupMembers	The maximum number of row groups or column groups a crosstab can display before the editor prompts the user for confirmation. This limit is a safeguard to avoid performance issues when grouping a field with too many values. The default is 100. Set it higher to allow more groups to appear without prompting users.
createColumn CrosstabHeaders	This property is located in the actualSizeReportGenerator bean. With the default setting of false, column group headers appear in the Ad Hoc view but not in the report generated from the view (this is the historical behavior). When set to true, column group headers appear in both the Ad Hoc view and in the generated report. When column group headers are included in a crosstab, an extra row is added below the column headers for spacing.

Property	Description
resolveRuntimeAdhocFilterLabels	This property resolves the inconsistency between Adhoc View Filters and Filters/Input Controls labels in dependent Adhoc Reports/Dashboards/Visualize where operators were not previously shown, and it is disabled by default.
	To have Adhoc Designer resolve Ad Hoc filter labels at runtime, set this property to *true* and restart the application server. Otherwise (with default setting), you will need to re-save all the AdHoc views in order for the operator to be displayed in the dependent report, dashboard and in Visualize.
	Note: If the property is set to true (which means the UI ignores any custom labels given by user on the UI) and the existing dashboard has 2 views/reports with the same filters name, the UI uses the label sent by the server. As a result, those filters will have the same labels. For example, creating new dashboard with *Country* filter for both will display them as *Country* and *Country2*.

The repository URI locations are relative to each and every organization in the server instance. For example, for a user in the default organization, the URI /adhoc actually refers to /organizations/organization_1/adhoc.

Ad Hoc Cache Management

> 📝 The Ad Hoc cache applies to Ad Hoc views based on Topics or Domains, and any reports generated from those Ad Hoc views. Ad Hoc views based on OLAP connections use the OLAP cache. For a comparison of the two caches, see Comparison with Jaspersoft OLAP Cache. For instructions on setting the OLAP cache, see the Jaspersoft OLAP User Guide.

JasperReports Server can temporarily cache Ad Hoc query result sets for re-use. The cache is populated by the data that results from queries when creating or running Ad Hoc views. The datasets are uniquely identified by a key that references the query itself, the data source URI, and parameters used when the query was issued.

Caching reduces database loads and delivers frequently-used datasets to the user quickly. Caching applies when reports are created and when they're run. The cache implementation uses Ehcache, which allows the administrator to view cache entry memory size and set limits on memory use. Configure the Ad Hoc cache to optimize memory use and response time for your use patterns.

Setting the Cache Granularity

By default, datasets for each user are cached separately. A parameter in the cache key identifies the user. This per-user caching can result in duplicate datasets and impaired performance when different users run the same query. You can configure JasperReports Server to share cached datasets across users by editing the .../WEB-INF/applicationContext-datarator.xml file.

The following code configures the cacheKeyInterceptor bean to ignore logged-in users' credentials when creating the cache keys:

```
<property name="ignoredParameters">
    <list>
        ...
        <value>LoggedInUser</value>
        <value>LoggedInUsername</value>
    </list>
</property>
```

Restart JasperReports Server after saving the modified file.

Configuring the Cache

Caching improves overall performance of data retrieval and sorting, but unused datasets can consume memory and cached data can become stale. To address these concerns, the cache automatically removes datasets periodically. By default, datasets are removed from the cache if they are not accessed for 30 minutes. They are also cleared after 90 minutes, regardless of how recently they were accessed.

To configure the frequency with which the cache is automatically cleared, edit the following configuration file:

Ad Hoc Cache Expiration

Configuration File

.../WEB-INF/adhoc-ehcache.xml

Property	Default Value	Description
maxBytesLocalHeap	400M	The maximum memory use of the entire cache, by default 400 MB. Use K for kilobytes, M for megabytes, and G for gigabytes.
		If left unbounded, the Ad Hoc cache can use up all available memory in your JVM. Set this value according to your server's available memory, the size of your datasets, and the number of cache entries you expect based on concurrent Ad Hoc use and the time settings below. We recommend setting this value to about half of the maximum heap size you configured for the JVM (-Xmx setting).
timeToIdleSeconds	1800	The number of seconds to wait after a dataset is has been accessed before removing it from the cache. The default is equivalent to 30 minutes. Use 0 (zero) for no limit.
timeToLiveSeconds	5400	The maximum time that a dataset is stored in the cache, even if it is being repeatedly

Ad Hoc Cache Expiration	
	accessed. Ensures that stale data is periodically replaced. The default is equivalent to 90 minutes. Use 0 (zero) for no limit.

Restart JasperReports Server after modifying these values.

Manually Clearing the Cache

Administrators can also use the server interface to view the queries whose datasets are in the cache. Administrators can see the full query but never the contents of the dataset. The Ad Hoc cache page also displays performance data about each query. This information can be helpful when trying to resolve performance issues. The interface displays several values for every query:

- Age (min:sec) – Time since the dataset was first stored in the cache.

- Query (msec) – Time in milliseconds from when query was sent to the data source (database) until the first row was received.

- Fetch (msec) – Time in milliseconds from when first row was received from the data source (database) until the last row was received.

- Memory used (MB) – Size in megabytes of the resulting dataset being stored in the cache entry.

The Ad Hoc cache page also allows administrators to manually remove datasets if necessary. Removing a dataset from the cache forces the server to get fresh data the next time a user creates or runs an Ad Hoc view with that query.

To view queries and manually clear the Ad Hoc cache

Procedure

1. In JasperReports Server, log in as system administrator (superuser by default).

2. Click Manage > Server Settings and choose Ad Hoc Cache in the left panel.
   The Ad Hoc Cache page appears, displaying all the datasets in the cache, sorted by age.

Figure 92: Ad Hoc Dataset Caching Administration Page

Each dataset is listed by its corresponding query and data source. Recall that Ad Hoc Topics have user-defined queries, so they tend to be short, whereas the query for Domains are generated from the design of the Domain and user selections in the Data Chooser dialog. The Ad Hoc Cache page displays only the first few lines of a query, as well as the data source.

3. To remove all datasets, click Clear All at the bottom of the Ad Hoc Cache page. This also clears the Teiid cache used by virtual data sources, including a virtual data source that wraps a data source for big data.

4. To remove a specific dataset from the cache, and click Clear beside the corresponding query.

5. To find a specific cache entry, you can change the sorting in the upper-left by clicking Age, Last Used Time, Memory Used, or Data Source URI.

6. To view the details of a specific query, including the full query string, click the query itself in the Query & Source column. The Detail page appears, displaying additional information for the selected query, such as the number of rows in the cached dataset.
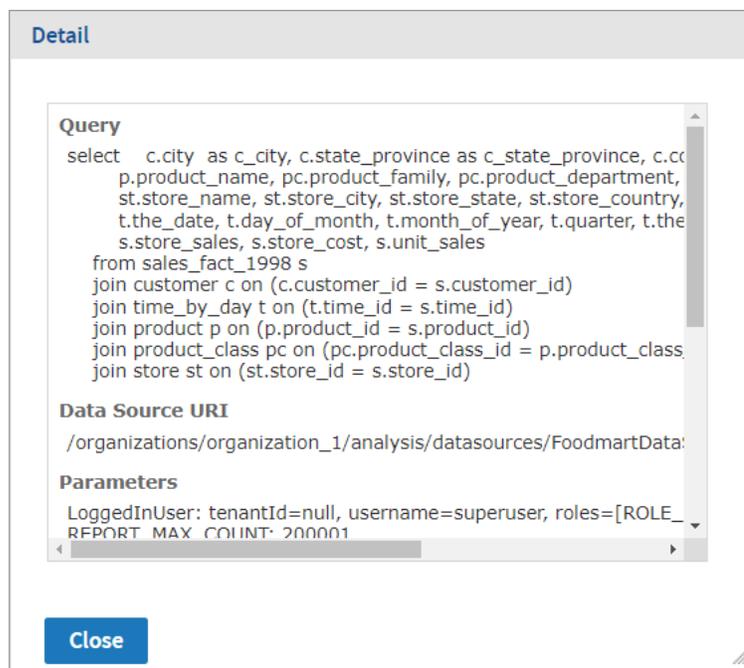
Figure 93: Typical Dataset in Ad Hoc Cache

Comparison with Jaspersoft OLAP Cache

The following table contrasts the key features of the Ad Hoc cache in JasperReports Server and Jaspersoft OLAP.

Ad Hoc Caching in JasperReports Server and Jaspersoft OLAP

Cache Feature	JasperReports Server	Jaspersoft OLAP
Structure of cache	Result caches are held at the query level: query text and language, plus data source URI and query parameters.	Result caches are held at the analysis connection level: schema plus database connection.
Sharing	Not by default, but can be enabled as described in Setting the Cache Granularity, above.	There is only one cache; it is shared across all queries and users.
Security	Applied to cache control so that users are not allowed to see privileged	

Cache Feature	JasperReports Server	Jaspersoft OLAP
	data.	
Populating	Queries populate the cache. You can also schedule reports to pre-populate the cache during off-hours.	
Size	Limited by available JVM memory (heap). Not configurable. Running out of memory is unusual. It can only happen if a single query returns too many elements for available memory. The report fails with an out-of-memory error.	
Automatic time-based cache policy	Configurable, as described in Configuring the Cache above.	In low-memory situations, cached items are removed automatically by JVM garbage collection; the least-recently-used items are cleared first. There is no way to remove data based on how long it has been in memory.
Clearing selected datasets manually	Configurable, as described in Manually Clearing the Cache above.	Cache regions can be defined and cleared programmatically with APIs.
Clearing all datasets manually	Configurable, as described in Manually Clearing the Cache above.	In JasperReports Server, select Manage > Server Settings, then select OLAP Settings and click Flush OLAP Cache. For additional methods, see the Jaspersoft OLAP Ultimate Guide.

Disabling the Ad Hoc Cache

> Disabling either cache is a server-wide setting that applies to all data sources or connections used in any Ad Hoc view. Make sure that other views aren't negatively affected by this change.

There are two reasons to consider disabling the Ad Hoc cache:

- You have a high-performance database that returns results so fast that additional caching in the server does not improve response times. In this case, the slight overhead of the cache may actually impact performance.

- Your database manages real-time data, and you create Ad Hoc views that present up-to-the-minute information from this data source. In this case, you don't want to retrieve old data out of the cache.

To disable the Ad Hoc cache for Topics and Domains, set the value of maxBytesLocalHeap to 1 (1 byte). For instructions, see Configuring the Cache. This effectively turns off the cache so that every query is retrieved directly from the data source.

To disable the OLAP cache for OLAP connections used in the Ad Hoc Editor, check the mondrian.rolap.star.disable-Caching setting on the Manage > Server Settings > OLAP Settings page. For more information, see the Jaspersoft OLAP User Guide.

In addition, if you have modified any of these three properties in applicationContext-adhoc.xml, set them back to false:

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

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

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

Ad Hoc OLAP Filter Configuration

When using filters in Ad Hoc OLAP, the server queries the database to display a list of values to select from. To avoid performance issues, the number of items in a filter is limited. By default, the limit is 250 possible values.

If your filters reach this limit and your list of values is truncated, you should first consider using a different filter operation. For example, instead of "city is one of <list>," use "City starts with <letter>." If you still need to change this limit, modify the following property:

Ad Hoc OLAP Filter Limit		
Configuration File		
.../WEB-INF/applicationContext-adhoc-dataStrategy.xml		
Property	Bean	Description
maxFilterValues	mdxDataStrategy	Set the value to the maximum number of filter values you expect. Setting this value higher than the default of 250 may cause performance issues.

Enabling Data Snapshots

The data snapshot feature stores report data in the server, which can change in the user experience significantly:

- Without data snapshots – Whenever users run a report, the server queries the data source and displays the latest data. When the same report is run over and over, the data source is often returning the same data every time. This is the default behavior.

- With data snapshots – The first time a report runs, it queries the data source and stores a copy of the data with the report in the repository. Users who view the report later see the data from the saved snapshot, not from querying the data source. Reports accessed through REST APIs and Visualize.js are also based on the saved snapshot. For large reports or frequently viewed reports, the persisted snapshot provides a significant performance gain and reduces load on your data sources. Every user who has access to the report will see the data from the same snapshot. For users who require it, the report viewer provides a button to manually refresh the data snapshot anytime. In addition, when the scheduler runs a job on a report it always updates the snapshot. Enable data snapshots if you'd like to use them.

We encourage enabling data snapshots with the following recommendations:

- If you have a new installation of JasperReports Server, enable snapshots to get the full server functionality. In the future, persistent data snapshots may be enabled by default.

- If you're upgrading from an early release that predates data snapshots, first follow the upgrade procedure and verify the outcome, as instructed in the JasperReports Server Installation Guide. Then, before enabling data snapshots, notify your users about this new functionality.

- Data snapshots are stored in the server's repository, which must be sized accordingly. If you have a large number of reports, or very large reports, consider the performance of your repository database before enabling snapshots. If your users rely on data that changes frequently or if they expect to see real-time data when opening a report, do not enable snapshots. Alternatively, you can enable snapshots selectively as described below.

Global Data Snapshot Configuration

The server-level settings determine whether the snapshot feature is available on the server.

Data Snapshots Server-Level Configuration		
Configuration File		
.../WEB-INF/applicationContext-data-snapshots.xml		
Property	Bean	Description
snapshot Persistence Enabled	dataSnapshot Service	When set to true, it allows the JasperReports Server report viewer to save data snapshots in the repository and open them the next time the report is run. By default, this is set to false.

> There is also a property named snapshotRecordingEnabled that caches a snapshot in the report viewer memory when sorting and filtering columns interactively. This allows the report viewer to refresh the display without querying the database every time. Regardless of persistence, snapshotRecordingEnabled improves report viewing performance and decreases database load, so it should remain set to true.

Report-level Data Snapshot Configuration

You can disable snapshots on a specific report by setting the following property in the report's JRXML:

```
net.sf.jasperreports.data.cache.persistable=false
```

> 📝 This report-level property depends on the snapshot mechanism in JasperReports Server. This property has no effect in other report viewers without such a mechanism, like the viewer integrated in Jaspersoft Studio.

There are two ways to control snapshots at the report level. In the case above:

- Data snapshots are enabled on the server, so most reports use them.

- Reports that don't benefit from data snapshots can explicitly disable snapshots in their own JRXML.

As with all report-level properties, you can set server-wide default values, as described in Configuring JasperReports Library:

Data Snapshots Default Report-Level Configuration	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	Description
net.sf.jasperreports.data.cache.persistable=false	When set to false, the server-wide default for reports is to not use data snapshots, however, they are still available if a report overrides this value to true in its own JRXML.

Because the report-level property takes precedence over the server-level property, this enables a second way to control snapshots:

- Data snapshots are enabled on the server.

- But the server-wide default is set to false, so most reports don't use them.

- Reports that benefit from data snapshots can explicitly enable snapshots in their own JRXML with:

net.sf.jasperreports.data.cache.persistable=true

Data Snapshots in the Scheduler

Scheduled jobs always run the report by accessing the data source, so the output has up-to-the-minute data. When data snapshots are enabled on reports, the job always updates the data snapshot with this new data after it runs. This way, when you schedule a report, it also refreshes the data snapshot periodically: hourly, daily, weekly, or whatever suits your data requirements.

When data snapshots are enabled on the server, the scheduler interface has an extra option to output the data snapshot. This option, as shown in the following figure, generates a copy of the report with the new data snapshot. This copy is stored in the repository as a JasperReport, identical to the report being run. Over time, this will create an archive of your report data.

If you clear the Data Snapshot Output Format option, no copy of the report is saved with the new data snapshot, but the data snapshot on the original report is still updated when the job runs. Also, you must select at least one other output format to schedule the report.
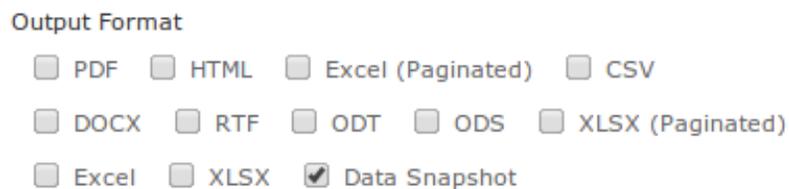
Figure 94: The Data Snapshot Output Option in the Scheduler

Finally, when data snapshots are enabled, you can also update them through REST web services calls. When specifying the report to run with the rest_v2/reportExecutions service, you can add arguments to explicitly update or not update the associated data snapshot. For more information, see the JasperReports Server REST API Reference.

Enabling Data Staging

Data staging is a new feature in JasperReports Server 6.0 that can optimize the performance of certain Ad Hoc views and reports. With data staging, the entire dataset for a Domain Topic is indefinitely cached in the server's Ad Hoc cache. When the entire dataset is pre-loaded in the server, Ad Hoc views and reports from that Domain Topic run faster

because they fetch data from the local cache, not by querying the production database. To keep data fresh, you can specify a reloading interval so that the server periodically accesses the database in the background to reload the entire dataset.

By default, data staging is not enabled on the server. You must enable data staging first, then each creator of a Domain Topic can choose to configure it for their Domain Topic.

When a Domain Topic uses data staging, it no longer uses any Domain-based optimizations (data policies) configured in the server because those apply only to database queries. Datasets that are staged are also exempt from the Ad Hoc cache time limits such as idle time and time-to-live. Data policies and Ad Hoc cache settings still apply to Domain Topics that do not use data staging.

When deciding to enable data staging, you should consider the nature of your data, its size, your database performance, and your access patterns to determine whether staging will help your views and reports run faster. You should also perform realistic usage tests to determine the amount of memory and size of data that give you the best performance. You can see staged datasets on the Ad Hoc cache page (Manage > Server Settings > Ad Hoc Cache), which displays their size and fetch time.

To get the full benefit of data staging, you must ensure your server has enough physical memory allocated to the Java virtual machine so that the cached dataset stays in memory. The cache for data staging will use disk storage when memory is full, but response times will suffer. Therefore, you must apply data staging to carefully selected Domain Topics so that the resulting sum of all staged datasets does not overwhelm your cache or your Java virtual machine (JVM) memory limits.

The cache for data staging also uses disk storage so that staged datasets are persistent during server restarts. Another advantage of staged datasets is that they allow fast reporting even when your original database has latency or downtime issues.

> When you enable Data Staging, Ad Hoc views based on Domain topics may return duplicate data.

Global Data Staging Configuration

The following setting determines whether the data staging feature is available on the server:

Data Staging Server-Level Configuration

Configuration File
.../WEB-INF/applicationContext-adhoc-dataStrategy.xml

Property	Bean	Description
`enabled`	`stagingService`	When set to true, it allows Domain Topic creators to configure data staging on Domain Topics. When a Domain Topic uses data staging, its entire data set is stored permanently in the Ad Hoc cache. By default, this is set to false.

Configuration File
.../WEB-INF/adhoc-ehcache.xml

Property	Bean	Description
`maxBytesLocalHeap`	`ehcache`	The total size of the heap shared between the Ad Hoc cache and the staging cache. The default is 400 MB. Use K for kilobytes, M for megabytes, and G for gigabytes. If you expect to have very large datasets being staged (hundreds of MB), you should increase this number to accommodate them. This value should still be about half of the maximum heap size you configured for the JVM (-Xmx setting).
`maxBytesLocalDisk`	`ehcache`	The maximum size of the staging cache stored on disk. The default is 2 GB. Use K for kilobytes, M for

Data Staging Server-Level Configuration	
	megabytes, and G for gigabytes. If you expect to have extremely large data sets being staged, you should increase this value as well.

Topic-level Data Staging Configuration

To use data staging, you must first create a Domain Topic by creating an Ad Hoc view and selecting a Domain as its source. After selecting a Domain, use the Data Chooser dialog to specify the sets and fields of the Domain in your Topic, along with any pre-filters and display names. The items chosen from the Domain, the filters in the Domain, along with any pre-filters in the Domain Topic all determine the dataset that data staging will store in the cache.

When data staging is enabled on the server as shown above, Domain Topics offer two new settings in the Save as Topic tab of the Data Chooser dialog.

Figure 95: Data Staging Options in Domain Topics

The two settings control data staging behavior:

- Enable staging — Turns on data staging for this Domain Topic. All Ad Hoc Editor actions, Ad Hoc views, and Ad Hoc reports based on this Domain Topic will use the staged data in the cache.

- Refresh interval for cached data — Determines the refresh interval of the staged data. The default minimum interval is 10 minutes. The maximum interval you can specify is 7 days.

Each time the specified interval is reached, the server refreshes the staged data by querying the data source for the entire dataset again. The new staged data replaces the old staged data in the cache so that the staging lasts indefinitely, but the dataset is renewed.

Staged data is refreshed asynchronously every interval by the server, independently of any access to the staged data by an Ad Hoc view or report.

> To ensure data security, data staging is disabled when your Domain is based on a data source that is defined with attributes. Server-level attributes are the exception and are allowed. Therefore, when a data source is defined with organization attributes or user attributes, the data staging settings are not available in the UI. For more information, see Attributes in Data Source Definitions.

Data Staging Dependencies

Data staging stores a dataset based on the items and filters selected in the Domain and Domain Topic. Views and reports based on the Domain Topic rely on this data being staged, that is, available in the cache.

When data staging has been turned on and views or reports have been created based on this Domain Topic, you can edit the Domain Topic as follows:

- You can add items from the Domain to the Domain Topic. When you save the Domain Topic, the server immediately retrieves the new dataset with the new items and stores it in the cache.

- You cannot remove items from the Domain Topic. If you try to remove items, the Data Chooser dialog will give an error. Look in the log file to find details about these dependencies.

- You cannot turn off data staging for this Domain Topic unless all Ad Hoc views and reports based on it have been deleted.

- You can change the refresh interval of the staged data. When you save the Domain Topic, the server immediately retrieves the dataset again and applies the new refresh interval in the future.

Configuring Cloud Services

The following settings control how JasperReports Server interacts with a cloud service provider, such as AWS and Microsoft Azure:

- The Cloud Settings page enables you to create and change firewall rules without restarting the server. For the Microsoft Azure service, these rules are called "access rules". For AWS, they are called "security groups".

- The AWS configuration file allows you to change the JDBC driver used for AWS data sources.

- The Azure configuration file allows you to change the JDBC driver and URL template used for Azure data sources.

For more information about AWS and Azure data sources, see AWS Data Sources and Azure SQL Data Sources.

Changing Cloud Services Settings

To change cloud services settings

Procedure

1. Log in as a system administrator (superuser by default).

2. Click Manage > Server Settings.

3. Click Cloud Settings in the left-menu.
   The Cloud Settings panel appears.

Figure 96: Cloud Settings Page

> We set up one AWS DB Security Group (using IP address) in each RDS region, per JasperReports Server instance. The security group allows connections from the specific JasperReports Server instance to the specified AWS database instance.

4. Modify the following settings and click Change after each modification. Changes are effective immediately on the server:

- Automatically Set Up an Access Rule for JasperReports Server: This checkbox is generally left checked. When checked the JasperReports Server will automatically create and update an access rule that allows connections from JasperReports Server to the database hosted by the cloud service provider. If you want to manage the access rules manually, uncheck this box.

- Access Rule Name: When JasperReports Server creates access rules to support

cloud-based data sources on this instance, it uses this name as the basis of the access rule name. When the JasperReports Server instance is running on AWS EC2, the EC2 instance ID is appended. When running outside of AWS EC2, you must make sure that the name is unique among JasperReports Server instances (that is, each instance should have its own name), so the IP addresses are properly granted access to the appropriate database instances.

- Access Rule Description: This text is used as the description for the access rule.

- JasperReports Server Public IP: Enter the public IP address for JasperReports Server. Most users on AWS EC2 should leave this field empty and let JasperReports Server determine the IP address automatically. It is possible with complex EC2 topology involving Virtual Private Clouds (VPCs) that you need to provide your IP address manually.

- Suppress EC2 Credentials Warning: If your JasperReports Server instance was created with no IAM role, when you go to the data source wizard to add an AWS data source with EC2 credentials there will be a warning message saying there is no proper role set. Checking this box suppresses the warning and disables the option.

Changing the Default JDBC Driver for AWS Data Sources

When adding an AWS data source, JasperReports Server uses the JDBC driver specified in the .../WEB-INF/applicationContext-webapp.xml file. You can configure JasperReports Server to use a different JDBC driver.

To change the JDBC driver used with AWS data sources

Procedure

1. Open the file .../WEB-INF/applicationContext-webapp.xml for editing.

2. Locate the jdbcConnectionMap bean and the key of your AWS database type within it. Modify this key to specify a different JDBC driver. For example, the default driver for MySQL databases is set to the MariaDB driver:

```
<entry key="mysql">
<util:map>
    ...
    <entry key="jdbcUrl" value="jdbc:mysql://$[dbHost]:$[dbPort]/$[dbName]"/>
    <entry key="jdbcDriverClass" value="org.mariadb.jdbc.Driver"/>
    ...
</util:map>
```

3. Save the file and restart JasperReports Server.

Changing Azure SQL Data Source Defaults

When creating an Azure SQL data source, JasperReports Server uses the JDBC driver and JDBC connector URL template specified in the .../WEB-INF/applicationContext-azure-sql-datasource.xml configuration file. You have the choice of changing the URL string and using the Microsoft JDBC driver for SQL Server when creating the data source or you can change the default driver and URL string in the configuration file.

Azure Data Source Defaults	
Configuration File	
.../WEB-INF/applicationContext-azure-sql-datasource.xml	
Bean	Description
defaultAzureJdbcDriverClassName	The JDBC driver is used for Azure SQL Server data sources. The default is the native JDBC driver for SQL Server (com.microsoft.sqlserver.jdbc.SQLServerDriver).
defaultAzureJdbcUrlSyntax	The template for the JDBC connector URL for Azure SQL Server data sources. The connector string contains properties required for the JDBC driver to access the Azure data source. The default string specifies the server name, database name, and the use of SSL as an encryption method.

Azure Data Source Defaults	
Configuration File	
.../WEB-INF/applicationContext-webapp.xml	
Bean	Description
defaultAzureKeyStoreType	The file format for storing the certificate key exchange files used for Azure data sources. The default file format is pkcs12.

Configuring Domains

Advanced uses of Domains may consider these configurations:

- Disabling the Domain Validation Check
- Setting the Level of Referential Integrity
- Optimizing Snowflake Schema Joins

When you use Domains with certain database constructs, you may need to configure JasperReports Server:

- Enabling Oracle Synonyms
- Enabling CLOB Fields
- Enabling Proprietary Types
- Extending JDBC Type Mapping
- Accessing Materialized Views
- Modifying Domain Calculated Field Variable Behavior

Disabling the Domain Validation Check

By default, JasperReports Server validates a Domain against its data source to ensure that the Domain design maps properly to the underlying tables. This validation occurs when a

Domain design file is uploaded to the server. If your data source is very large and complex, this validation can take time. If the validation takes too long, you can disable it. In this case, JasperReports Server assumes the Domain design is valid, and simply uploads it without the check.

To disable the validation edit the following configuration file:

Configuring Domain Validation Check		
Configuration File		
.../WEB-INF/applicationContext-semanticLayer.xml		
Property	Bean	Description
skipDomainDatabase Validation	slConfig	Set this property to TRUE to disable the validation check.

> 📝 If the tables and fields referenced in the Domain design don't exist in the data source when skipDomainDatabaseValidation is set to TRUE, the Domain wizard won't detect the problem, but the Choose Data wizard returns errors when your end users work with the Domain.

Setting the Level of Referential Integrity

Referential integrity is the verification process that occurs when opening an Ad Hoc view or a report that is based on a Domain. The server checks that all fields and measures referenced by the view or the report match those defined in the Domain. This ensures that out-of-date views or reports cannot be run if they rely on items in the Domain that were modified or removed.

In practice, when you modify a Domain to remove fields or measures, you must also modify all Ad Hoc views that depend on those fields and measures, and you must regenerate any reports based on those views. In case you don't, referential integrity will give an error so that the user knows why the report did not run, instead of failing at runtime or showing empty data. When opening an Ad Hoc view that has similar referential integrity errors, the server displays a list of missing fields and measures and prompts you to removes the items so the view is coherent when it opens.

However, in cases where Domain security makes a field unavailable to a user, this will also be detected as a referential integrity error upon opening a view or running a report. Some

views and their corresponding reports may be designed for multiple users with different security access to data, and you intend to display the report with blank or zero in place of certain data. In such cases, you should change the default setting for referential integrity so that users can open views and run reports without causing an error.

Configuring Referential Integrity

Configuration File
.../WEB-INF/applicationContext-adhoc-dataStrategy.xml

Property	Default	Description
checkSourcesInStrictMode	true	By default (when true) when opening an Ad Hoc view, any mismatch between its fields and those in the Domain it references trigger a missing data dialog in the Ad Hoc editor and prompt the user to remove those fields. When this configuration is set to false, the following referential integrity errors are allowed: • The item's table ID or join ID does not match the one in the Domain, provided its unique item ID still matches one in another set or subset. • The item's type does not match the one in the Domain, provided the item is not used in any filter, calculated field, or summary function that is incompatible with the altered type. Referential integrity issues that do not meet these criteria will still trigger a missing data dialog in the Ad Hoc editor, even if this setting is false. For example, if an aggregation on a numerical field in the view

Configuring Referential Integrity		
		is set to Average, but the field in the Domain is now a string, an error is triggered because strings cannot be averaged.
skipCheckSourcesForViews	false	When rendering an Ad Hoc view based on a Domain, the server checks that all fields and measures in the view match those of the Domain. By default (when this setting is false), if there missing fields, the sever will display an error and not display the view.
		When this setting is true, the server will render the view even if some fields are missing for any reason, for example if they are hidden due to security in the Domain. This setting is more permissive of referential integrity errors than checkSourcesInStrictMode above.
skipCheckSourcesForReports	false	When running a report based on an Ad Hoc view based on a Domain, the server checks that all fields and measures in the report match those of the Domain. By default (when this setting is false), if there is a mismatch, the sever will display an error and not run the report.
		When this setting is true, the server will run the report with missing fields and the result depends on the severity of any mismatch. Fields that are hidden due to security in a properly designed report will appear blank or zero. A report where the missing field leads to a computatuion error will cause the running report to fail.

> 📝 If you change the referential integrity settings in order to allow certain reports to work with Domain security, you should make sure those reports work as intended for all users and ensure all other reports run correctly. If you later remove fields or measures from your Domains, you should also be diligent in manually updating all views and reports that depend on those Domains.

Optimizing Snowflake Schema Joins

When creating a Domain on top of a snowflake schema, the default joins generated when using the Domain in the Ad Hoc Editor may take a long time and include dimensions not used in the report. For example, a schema with over a hundred dimension tables mostly connected to a subset of 5-10 fact tables may cause such behavior. The following setting can be enabled to optimize the joins generated for such a snowflake database schema. The default setting has better performance in the more common cases with fewer tables.

Configuring Domain Join Optimization		
Configuration File		
.../WEB-INF/applicationContext-semanticLayer.xml		
Property	Bean	Description
specialOptimizationOn	graph Operations	The default setting of false handles typical cases of Domains based on 10-100 tables. For snowflake schemas that typically have 100 or more tables, or for database topologies that cause slow join performance in Ad Hoc views, set this property to true to optimize the joins in the Ad Hoc Editor.

Enabling Oracle Synonyms

By default, Domains can't access synonyms in an Oracle database. Settings to enable them vary depending on your application server. The settings shown here apply to Apache Tomcat. Your application server may require different values. Set the following property to

enable Oracle synonyms on Tomcat. If you access your Oracle database through JNDI, you also need to configure the JNDI connection.

Be aware that the Oracle metadata service is significantly slower when synonyms are in scope.

Enabling Oracle Synonyms

Configuration File

…/WEB-INF/applicationContext-jdbc-metadata.xml

Property	Bean	Description
includeSynonyms ForOracle	jdbcMeta Configuration	Set the value to true: <value>true</value>

Configuration File

…/META-INF/context.xml

Property	Bean	Description
accessToUnderlying ConnectionAllowed	<Resourcename= "jdbc/oracle"…	If you use JNDI, add the following property: accessToUnderlying ConnectionAllowed="true"

Enabling CLOB Fields

Support for CLOB (Character Large Object) fields is dependent on your database and must be enabled manually. If you want to access CLOB fields in JasperReports Server, set the following options according to your database.

The Oracle JDBC driver implementation uses the CLOB JDBC type for CLOB fields.

CLOB Support for Oracle

Configuration File
.../WEB-INF/applicationContext-jdbc-metadata.xml

Property	Bean	Description

jdbc2JavaType Mapping	jdbcMeta Configuration	This property contains a map of database field types to Java types. Find the line for CLOB that is commented out:
		
		`<!--entry key="CLOB" value=""/-->`
		
		Modify it as follows:
		
		`<entry key="CLOB" value="java.lang.String"/>`

The MySQL JDBC driver implementation uses either the CLOB JDBC type, the LONGVARBINARY JDBC type, or both to represent CLOB fields, depending on their length.

CLOB Support for MySQL

Configuration File
.../WEB-INF/applicationContext-jdbc-metadata.xml

Property	Bean	Description

jdbc2JavaType Mapping	jdbcMeta Configuration	This property contains a map of database field types to Java types. Find the following lines:
		
		`<!--entry key="CLOB" value=""/-->`
		`<!--entry key="LONGVARBINARY"`

CLOB Support for MySQL

```
                                              value=""/-->
```

And modify them as follows:

```
<entry key="CLOB"
value="java.lang.String"/>
<entry key="LONGVARBINARY"
value="java.lang.String"/>
```

📝 Due to a limitation, CLOB and NTEXT fields cannot be sorted or compared in a query. For example, trying to add a CLOB field as a crosstab column with domain query optimization enabled will result in an error message.

Enabling Proprietary Types

JasperReports Server provides a JDBC-to-Java type mapping for all standard JDBC column types for use in Domains. However, some databases have proprietary types that you may need to map to a Java type with a special configuration. Some proprietary types, such as NVARCHAR2 for Oracle, are already mapped by default.

As a prerequisite, the proprietary type must be logically equivalent to one of following Java classes:

`java.lang.Boolean`	`java.lang.Integer`	
		`java.sql.Date`
`java.lang.Byte`	`java.lang.Long`	
		`java.sql.Time`
`java.lang.Character`	`java.lang.Short`	
		`java.sql.Timestamp`
`java.lang.Double`	`java.lang.String`	
		`java.util.Date`
`java.lang.Float`	`java.math.BigDecimal`	

There are two ways to create a mapping for a proprietary type, as shown in the following table:

- Modify the generic mapping for NUMERIC types. By default, any numeric type that doesn't match one of the other types is mapped to BigDecimal.

- Create a secondary mapping under the special OTHER key, where the secondary key can be your custom type name.

Proprietary Database Type Mapping		
Configuration File		
.../WEB-INF/applicationContext-jdbc-metadata.xml		
Properties	Bean	Description

jdbc2Java TypeMapping	jdbcMeta Configuration	If your proprietary type is not already defined in this file, you can add it:
		
		• To modify the generic mapping, edit this line:
		
		```<entry key="NUMERIC" value="java.math.BigDecimal"/>```
		
		• To add a secondary key to the OTHER key, follow this example:
		
		```<entry key="OTHER"> <map> <entry key="NVARCHAR2" value="java.lang.String"/> </map></entry>```

Extending JDBC Type Mapping

Some database types are not even mapped to a JDBC type. In particular, Oracle uses TIMESTAMP WITH TIME ZONE and TIMESTAMP WITH LOCAL TIME ZONE that must be mapped in order to appear in JasperReports Server. If there are other types in your

database, you can override or extend the JDBC type mapping with the following configuration:

Extending JDBC Type Mapping		
Configuration File		
.../WEB-INF/applicationContext-jdbc-metadata.xml		
Property	Bean	Description

codeToJdbcType Mapping	jdbcMeta Configuration	This property contains a map of database type codes to JDBC types. By default the codes for Oracle TIMESTAMP types are mapped:
		
		`<entry key="-101" value="TIMESTAMP"/>`
		`<entry key="-102" value="TIMESTAMP"/>`
		
		Add or replace these entries to map additional types from your database.

Overriding JDBC Type Mapping for Selected Database

The `applicationContext-jdbc-metadata.xml` file contains `jdbcMetaConfiguration`. It is global for all drivers and contains the following two properties, that are used to map JDBC type to Java types:

- `jdbc2JavaTypeMapping`

- `codeToJdbcTypeMapping`

In some cases, database or JDBC driver vendors may use the same Type Code or JDBC type for different data types. To address this, JasperReports Server allows the definition of custom mappings for specific databases while keeping the global configuration intact.

For example, the native MS SQL Server driver might return code 92 for a column of type `Time(3)`. By default, JasperReports Server maps code 92 to `java.sql.Types.TIME`. However, different JDBC drivers may interpret this code differently; for instance, the

Progress SQL Server driver maps it to `java.sql.Types.TIMESTAMP`. To adjust MS SQL Server behaviour, a specific configuration for the MS SQL Server driver can be added to override the default mapping.

Follow these steps to customize the configuration:

1. Find the `<bean class="com.jaspersoft.jasperserver.api.engine.common.domain.JdbcDriverMetaConfigurationImpl">` in the `../WEB-INF/applicationContext-jdbc-metadata.xml` file.

2. Uncomment the bean and set its properties:

```
<bean
class="com.jaspersoft.jasperserver.api.engine.common.domain.Jdbc
DriverMetaConfigurationImpl">
    <property name="databaseProductName" value="Microsoft SQL
Server"/>
    <property name="codeToJdbcTypeMapping">
        <map>
            <entry key="92" value="TIMESTAMP"/>
            <entry key="8" value="FLOAT_OR_DOUBLE_TYPE"/>
        </map>
    </property>
</bean>
```

Ensure that the class is set to `com.jaspersoft.jasperserver.api.engine.common.domain.JdbcDriverMetaConfigurationImpl`, indicating that this bean can override the default configuration per driver type. The databaseProductName value should be set to the JDBC Driver Name found in the driver's official documentation or retrieved by executing getDatabaseProductName() in Java code.

3. Restart JasperReports Server.

In this specific configuration for the MS SQL Server driver, the `codeToJdbcTypeMapping` property is adjusted to override the default mappings for code 92 and code 8.

In certain scenarios, MS SQL Server may return code 8 for columns of the Float type. This default code is later mapped to `java.sql.Types.DOUBLE`. However, challenges arise when the same code is used for other data types like Real or Double, and this behavior can vary between JDBC driver vendors. For example, the Progress JDBC driver interprets such columns as `java.lang.Float`.

To resolve this difference, a specialized mapping is introduced by assigning code 8 to `FLOAT_OR_DOUBLE_TYPE`. This addition adds an extra check performed later in the `jdbc2JavaTypeMapping` property:

```
<property name="jdbc2JavaTypeMapping">
    <map>
        <!-- ... other mappings ... -->
        <entry key="FLOAT_OR_DOUBLE_TYPE">
            <map>
                <entry key="float" value="java.lang.Float"/>
                <entry key="otherColumnTypes" value="java.lang.Double"/>
            </map>
        </entry>
    </map>
</property>
```

In this configuration:

- For columns identified as float, JasperReports Server maps the type to `java.lang.Float`.

- For other floating-point types like double, real, or numeric, identified by the value `otherColumnTypes`, the mapping is set to `java.lang.Double`.

Accessing Materialized Views

Domains access tables and views by default, but some databases support other structures such as materialized views. These alternate table structures don't show up by default, but you can often configure Domains to display and access them.

If the JDBC driver for your database assigns a standard table type identifier to the materialized view, you can access it in Domains, Ad Hoc views, and reports. To find the table type, use a JDBC client such as the SQuirreL tool to view your database schema. In SQuirreL, use the "Objects" tab to browse the tables and views organized by table type. Look for your materialized view and note its table type.

The table type values are defined in the DatabaseMetaData.html.getTables() documentation. When you know the string corresponding to your table type, add it to the following configuration value:

Accessing Materialized Views		
Configuration File		
.../WEB-INF/applicationContext-jdbc-metadata.xml		
Property	**Bean**	**Description**
tableTypes	jdbcMeta Configuration	Uncomment the JDBC table type corresponding to the materialized view or other table structure in your databases, for example: <value>LOCAL TEMPORARY</value>

Modifying Domain Calculated Field Variable Behavior

The property *resolveDomainCalcFieldVariableInAdvance* included in WEB-INF/applicationContext-adhoc-dataStrategy.xml for bean id="commonDomainDataStrategy" allows you to calculate a domain calculated field in two ways. The default value is set to *false*, and the behavior overrides the lookupExpression(Variable var) in ColumnCalculation.

For example, if you have a domain calculated field expression such as "coverage / allowance", it will translate to sum(coverage) / sum(allowance) instead of sum(coverage / allowance).

You can change the default domain calculated field variable behavior to use a sum aggregation (the original behavior prior to JasperReports Server 7.8) with the following configuration:

Modifying Domain Calculated Field Variable Behavior
Configuration File
.../WEB-INF/applicationContext-adhoc-dataStrategy.xml

Modifying Domain Calculated Field Variable Behavior		
Property	Bean	Description
resolveDomainCalcFieldVariableInAdvance	commonDomainData Strategy	Set value to *true*: <property name="resolveDomainCalcFieldVariableInAdvance" value="true"/>

Configuring JasperReports Library

JasperReports Server's reporting features are built on the JasperReports Library, which is embedded in the server. Many of the options you can configure to change the server's functionality are actually JasperReports Library options. The configuration options can control many aspects of the server's behavior, from the way reports are exported into different file formats, to the default font.

These options can be set at different levels:

- Global – Applies to all reports generated by the server. Global JasperReports properties are defined in the .../WEB-INF/classes/jasperreports.properties file.

- Report – Defined in the JRXML of the report and applies to that specific report.

- Element – Defined in the JRXML and applies to specific elements of the report.

For more information about JasperReports Library configuration, see http://jasperreports.sourceforge.net/config.reference.html.

The following sections highlight a few of the available options:

- Extending JasperReports Library

- Changing the Crosstab Limit

- Setting a Global Chart Theme

- Disabling Interactivity in the Report Viewer

- Disabling Chart Types in Dashboards

- Changing the Pro Charts Rendering Engine

- Configuring a JavaScript Engine for Graphical Report Rendering

- [Static Export Properties for High Charts](#)

- [Enabling PDF Accessibility Features in Tables](#)

- [Enabling HTML in Tooltips](#)

Extending JasperReports Library

You can extend JasperReports Library by implementing the public interfaces it exposes.

Such an implementation is usually stored in a JAR (Java Archive) that contains a file called jasperreports_extension.properties, and specifies a factory class used to instantiate an extension registry. The extension registry specifies one or more extension objects, each of which corresponds to a JasperReports Library extension point represented by a Java interface.

Place this JAR on the JasperReports Library classpath, and your extension is automatically available.

For more information, refer to JasperReports Library Ultimate Guide.

Changing the Crosstab Limit

If you use crosstab reports, you may experience Out of Memory errors if the reports are very large or complex. You can configure JasperReports Server to return a message instead of memory errors when users run such crosstabs. To do so, enable the net.sf.jasperreports.crosstab.bucket.measure.limit property and set its maximum value in the following configuration file:

Crosstab Report Configuration Option	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	Description
net.sf.jasperreports.	This value effectively limits the number of cells in a

Crosstab Report Configuration Option	
crosstab.bucket. measure.limit	crosstab, which can be computed as follows: (number of crosstab rows) x (number of crosstab columns) x (number of user-defined measures + 1) The default value is 100000. Enter large values to allow your users to create larger, more complicated crosstabs; enter small values to restrict them. If you experience OutOfMemoryExceptions after changing this value, try setting it to a smaller number, or configure your JVM to allow more memory to be used.

Setting a Global Chart Theme

Chart themes control the look and feel of the charts generated by JasperReports Server. Chart themes can be applied at the level of either the server or the individual report:

- To apply a theme at the report level, select it when designing the report in Jaspersoft Studio. Note that you can also apply a theme to individual chart elements. Note that a chart theme can be included in a report unit as a resource. In this case the theme is available only to charts in that report unit.

- To apply a theme at the server level, copy the chart theme JAR to the correct location and edit its configuration file.

A chart theme is a JAR file that defines the look and feel of a chart. Once you have created the chart theme JAR file, copy it to the .../WEB-INF/lib directory. Chart themes in this location are available to any chart in the instance of the server. They can also be set as the global chart theme.

To set a theme as the default chart theme, edit the following configuration file:

Global Report Theme
Configuration File

Global Report Theme	
.../WEB-INF/classes/jasperreports.properties	
Property	Description
net.sf.jasperreports. chart.ChartTheme	The name of a chart theme that is in the .../WEB-INF/lib directory.

We recommend that you create your chart themes in Jaspersoft Studio. Click File > New > Other > Chart Theme, then use Jaspersoft Studio to archive the new chart theme as a JAR.

> Chart themes do not apply to Ad Hoc chart views.

Disabling Interactivity in the Report Viewer

By default, the report viewer's interactivity is enabled, and reports with interactive elements (such as the table component) are interactive when run in the web server and displayed in the viewer. If you don't want your reports to be interactive, you can disable interactivity across all reports by editing the following configuration file.

Interactivity in the Report Viewer	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	Description
net.sf.jasperreports. components.table.interactive	By default, this property is set to true; in this case, interactivity is enabled in the report viewer. Set it to false to disable interactivity.

> Changing this setting in this configuration file changes the behavior for the entire server. To configure this behavior at the report, table, or column level, edit the report's JRXML properties in Jaspersoft Studio.

Disabling Chart Types in Dashboards

By default, interactivity is enabled in charts that appear in dashboards, and users can change the chart type by clicking on the chart type selector (gear icon). If you don't want users to modify the chart types, you can disable the chart type selector on all dashboard charts by editing the following configuration file.

Chart Type Selector in Dashboards	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	Description
com.jaspersoft.jasperreports. highcharts.interactive	By default, this property is set to true; in this case, chart types can be modified in dashboards. Set it to false to prevent users from changing dashboard charts.

> Changing this setting in this configuration file changes the behavior for the entire server. To configure this behavior at the report level, edit the report's JRXML properties in Jaspersoft Studio.

Changing the Pro Charts Rendering Engine

By default, JasperReports Server renders Pro Charts (those based on Fusion Charts) using HTML5. If your browser doesn't support HTML5, the chart won't be rendered unless you change the configuration. Though we don't recommend it, you can instead have these elements rendered by Adobe Flash; in some circumstances, Fusion's Flash solution may pose a security risk.

Note that Pro Charts are available only in the JasperReports Server Professional edition.

To render Pro Charts using HTML5, edit the following configuration file:

Pro Charts Renderer	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	Description
com.jaspersoft.jasperreports. fusion.charts.render.type	Determines which of the following renderers is used: • html5 is the default renderer for Pro Charts. It is the preferred renderer. • flash is a deprecated renderer for Pro Charts.

Note that this property applies only to reports that rely on Pro Charts and affects only the HTML preview and export.

Typically, this property is set at the server level; to override the server-level setting for a specific Pro Chart report, set this property at the report level, and also specify a second property as shown:

```
net.sf.jasperreports.print.transfer.fusion=com.jaspersoft.jasperreports.
fusion
```

This allows the reporting engine, JasperReports Library, to recognize the Fusion settings. If this property isn't set, the `com.jaspersoft.jasperreports.fusion.charts.render.type` property is ignored at the report level.

Configuring a JavaScript Engine for Graphical Report Rendering

Depending on the circumstances, a given graphical element (such as a chart, a map, or a widget) in a report can be rendered in two ways:

• When run directly in the web UI, the browser itself renders the chart.

- When scheduled to run later or run in the background, an external engine renders the chart.

JasperReports Server uses the Chromium JavaScript engine to generate graphical reports that are run in the background or scheduled. You can use Chrome, Chromium, or any other browser based on Chromium like Microsoft Edge. For information on the compatible versions of browsers, see the *Jaspersoft Platform Support Guide*. Download the correct version of Chrome/Chromium for your environment and install it on the computer hosting JasperReports Server.

Once Chrome/Chromium is installed, point JasperReports Server to its location. You can configure several processes to use Chrome/Chromium: HighCharts generation, Pro Charts generation (including Pro Widgets and Pro Maps), and exporting dashboards. Chrome/Chromium can be used to render reports and dashboards when exporting them to a PNG, PDF, ODT, or DOCX file.

> These are server-wide settings. In a given server, all charts of the same type (HighCharts or Fusion (Charts Pro, Maps Pro, or Widgets Pro) use the same JavaScript engine.

To configure JasperReports Server to use Chrome/Chromium for HighCharts generation, Pro Charts generation (including Pro Widgets and Pro Maps), and exporting dashboards, edit the following properties:

JavaScript Engine Configuration for HighCharts, Pro Charts, and Dashboards	
Configuration File	
.../WEB-INF/js.config.properties	
Property	Description
chrome.path	This property points to the engine the server should use to generate dashboards when exporting and to generate HighCharts-based and Pro Charts-based charts in reports run in the background or scheduler.
	Specify the location for Chrome/Chromium, for example:
	chrome.path = C:/Program Files (x86)/Google/Chrome/Application/chrome.exe

JavaScript Engine Configuration for HighCharts, Pro Charts, and Dashboards

	Alternatively, to use another browser, such as Edge, specify the respective path.
	When you use binary installer to install JasperReports Server, you can give the path to Chrome or Chromium in installer during installation and the path gets automatically set in js.config.properties file.
	When you use WAR file to install JasperReports Server, this property needs to be set in default_master.properties file. When configured in default_master.properties, it gets automatically set in js.config.properties file on deployment.
chrome.page.timeout	The maximum number of seconds to wait for output from Chrome\Chromium before the chart times out. The default is 150.

Configuration File

.../WEB-INF/classes/jasperreports.properties

Property	Description
net.sf.jasperreports.chrome.argument.no-sandbox	If Tomcat is run as root on Linux, the export of reports, Ad Hoc views, and dashboards fails. To avoid this, set this property to true: net.sf.jasperreports.chrome.argument.no-sandbox=true By default, this is set to false.

After setting these properties, restart JasperReports Server to enable them.

> If you are using JasperReports Server and Chrome/Chromium on a Windows platform, exporting large Fusion reports may cause issues. In those cases, we recommend editing the property as follows:
> - Set chrome.page.timeout to 600 to increase the timeout period.

Static Export Properties for High Charts

When you create interactive JasperReports that use Highcharts, and those reports are exported to HTML, the <script> elements in the HTML must reference the correct JavaScript libraries. The following JasperReports Library properties define the library paths:

Static Export Properties for Highcharts

Configuration File

.../WEB-INF/classes/jasperreports.properties

Property

```
# Highcharts static export properties
com.jaspersoft.jasperreports.highcharts.render.require.js$context.url
com.jaspersoft.jasperreports.highcharts.jquery.js$context.url
com.jaspersoft.jasperreports.highcharts.highcharts.js$context.url
com.jaspersoft.jasperreports.highcharts.highcharts.more.js$context.url
com.jaspersoft.jasperreports.highcharts.highcharts.heatmap.js$context.url
com.jaspersoft.jasperreports.highcharts.data.service.js$context.url
com.jaspersoft.jasperreports.highcharts.default.service.js$context.url

com.jaspersoft.jasperreports.highcharts.item.hyperlink.service.js$context.
url
com.jaspersoft.jasperreports.highcharts.y.axis.service.js$context.url
com.jaspersoft.jasperreports.highcharts.dual.pie.service.js$context.url
com.jaspersoft.ji.adhoc.highcharts.setting.service.js$context.url

com.jaspersoft.jasperreports.highcharts.highcharts.chart.producer.js$conte
xt.url
```

Each property that ends in $context.url may also have a similar property ending in $url. The $context.url property is used when a request object is available to resolve a context path for the HTML export, and the $url property is used as a fallback.

Not all library paths are defined in the jasperreports.properties file. Some are provided by the environment (such as com.jaspersoft.jasperreports.highcharts.jquery.ui.js), and others are set programmatically by the bean com.jaspersoft.ji.adhoc.jr.AdhocHighchartsSetting-ServiceBundle (in the .../WEB-INF/applicationContext-adhoc.xml file).

Enabling PDF Accessibility Features in Tables

JasperReports Library supports properties and metadata that allow people to create accessible PDF documents that can be read by screen readers. These properties help meet the requirements specified in the Section 508c of the United States Rehabilitation Act of 1973.

JasperReports Library provides the ability to automatically add 508c metadata to table components. When this features is enabled, tables in Ad Hoc reports and your own JasperReports contain the necessary metadata when exported to PDF.

Enabling PDF Accessibility Features in Tables	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	Description
net.sf.jasperreports.components.table.generate.pdf.tags	When set to true, metadata for accessibility is automatically generated in JasperReports tables.

This property can also be set at the report level or table component level to control the automatically generated metadata.

Enabling HTML in Tooltips

To enable HTML in tooltips, edit the following configuration file:

Enabling HTML in Tooltips	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	Description

Enabling HTML in Tooltips

com.jaspersoft.jasperreports.fusion.value.encoder=legacy	When set to *legacy*, HTML is enabled for tooltips in Maps and Charts. **Note**: When this functionality is enabled, visualizations are vulnerable to security issues, such as HTML injection attacks.

Disabling Open In Editor Option

By default, the Open In Editor property is enabled. It appears on the context menu when you right-click the report or files in the repository. Using the Open In Editor option, you can edit the files including CSS, XML, JSON, and JRXML in the text editor and JRXML editor respectively. You can disable Open In Editor and hide it from the context menu by editing …/WEB-INF/applicationContext-search.xml file.

To disable the open in editor

Procedure

1. Open the file …/WEB-INF/applicationContext-search.xml file for editing.

2. Locate the searchActionModel bean.

3. Set the openInEditor property to false and save the file.

```
<bean id="searchActionModel"
class="com.jaspersoft.jasperserver.search.model.SearchActionModelSupport">
   <propertyname="proVersion"value="${isProVersion}"/>
   <propertyname="openInEditor"value="false"/>
</bean>
```

4. Restart JasperReports Server.

Configuring Input Control Behavior

When defining text input controls, the default server behavior allows empty strings, even if you have configured a regular expression and made the input control mandatory. Use this setting to enforce the regular expression even on empty strings. This forces the user to provide a conforming value for the input control.

Input Control Behavior	
Configuration File	
.../WEB-INF/applicationContext-cascade.xml	
Bean	Description
applyRegexpToEmptyString	The default value of false gives the traditional behavior: even if a regular expression is defined, it is not applied to empty strings. If you want to strictly enforce the regular expression, even on empty input strings, set this property to true.

You can also configure the default value that appears in each type of input control. This is the value that is displayed when the input control is not given any value. By default, the display value is ~NULL~.

Edit the file .../WEB-INF/applicationContext-cascade.xml to change the following entries. The examples in comments show how you can use the default value to suggest a pattern for the input. To make an input control appear blank when no value is given, set value="" (an empty string).

```
<util:map id="globalDefaultValues" value-type="java.lang.String" key-type="java.lang.Byte">
    <!-- if DataType isn't defined-->
    <entry key="-1" value="~NULL~"></entry>
    <!--TYPE_TEXT = 1-->
    <!--<entry key="1" value="Enter value"></entry>-->
    <entry key="1" value="~NULL~"></entry>
    <!--TYPE_NUMBER = 2-->
    <!--<entry key="2" value="0"></entry>-->
    <entry key="2" value="~NULL~"></entry>
    <!--TYPE_DATE = 3-->
    <!--<entry key="3" value="2020-03-12"></entry>-->
    <entry key="3" value="~NULL~"></entry>
    <!--TYPE_DATE_TIME = 4-->
```

```
        <!--<entry key="4" value="2015-09-22T05:26:16"></entry>-->
        <entry key="4" value="~NULL~"></entry>
        <!--TYPE_TIME = 5-->
        <!--<entry key="5" value="13:37:54"></entry>-->
        <entry key="5" value="~NULL~"></entry>
    </util:map>
```

Configuring Case Sensitivity

You can configure the following property to turn on/off the case sensitivity for input control behavior.

Configuring the Case Sensitivity for Input Control Behavior	
Configuration File	
.../WEB-INF/js.config.properties	
Property	Description
inputControl.handler.values.caseSensitive	This property is used to set input control or filter values to case insensitivity for case-insensitive databases.
	By default, the value is inputControl.handler.values.caseSensitive=true. You can set this property to false to ignore case sensitivity.
	When set to false, the parameters or input controls validator becomes case insensitive.
	For example, if inputControl.handler.values.caseSensitive=true providing country="usa" triggers a validation exception.
	However, if inputControl.handler.values.caseSensitive=false
	providing any of the following values is acceptable:
	• country="USA"

Configuring the Case Sensitivity for Input Control Behavior
• country="UsA" • country="usa"

Note: The parameter *caseSensitive* is also added to the existing JSON model for ExistingFilters (DynamicFilterObject) in Crosstab/Table Model.

Configuring the Scheduler

The scheduler runs reports in the background according to a user-defined schedule (also called a job). You can configure the following aspects of the scheduler:

- Configuring the Scheduler Misfire Policy
- Configuring Scheduler Failure Notifications
- Restricting File System Output
- Removing Report Scheduling Interval Options
- Adding a Holiday Exclusion Calendar
- Changing the Default Output Folder
- Configuring the Scheduler for Dashboards
- Configuring the Scheduler to Fetch Attributes
- Setting Properties in the jasperserver_config.properties File

Configuring the Scheduler Misfire Policy

A scheduler misfire occurs when the scheduler cannot run a report at the designated time, for example because JasperReports Server is offline, its database is offline, or the number of threads is limited. In this case, you can configure the behavior of the scheduler to retry the report or skip the scheduled run.

You can set a different misfire policy for each kind of job schedule: single job, repeating job, and calendar job. Misfire policies are defined in the Quartz Scheduler documentation and other online resources:

https://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/tutorial-lesson-05.html

https://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/tutorial-lesson-06.html

https://nurkiewicz.com/2012/04/quartz-scheduler-misfire-instructions.html

Configuring Scheduler Misfire Policy

Configuration File

`.../WEB-INF/js.quartz.properties`

Property	Description
`report.quartz.misfirepolicy.singlesimplejob`	Sets the misfire policy for single jobs to one of the following: • `SMART_POLICY` • `MISFIRE_INSTRUCTION_FIRE_ NOW` • `MISFIRE_INSTRUCTION_IGNORE_ MISFIRE_POLICY`
`report.quartz.misfirepolicy. repeatingsimplejob`	Sets the misfire policy for repeating jobs to one of the following values: • `SMART_POLICY` • `MISFIRE_INSTRUCTION_FIRE_ NOW` • `MISFIRE_INSTRUCTION_IGNORE_ MISFIRE_POLICY` • `MISFIRE_INSTRUCTION_ RESCHEDULE_NEXT_WITH_ EXISTING_COUNT` • `MISFIRE_INSTRUCTION_ RESCHEDULE_NOW_WITH_ EXISTING_REPEAT_COUNT` • `MISFIRE_INSTRUCTION_ RESCHEDULE_NOW_WITH_`

Configuring Scheduler Misfire Policy

	REMAINING_REPEAT_COUNT

`report.quartz.misfirepolicy.`
`calendarjob`
	Sets the misfire policy for jobs with calendar recursion to one of the following values:
	
	• `SMART_POLICY`
	
	• `MISFIRE_INSTRUCTION_IGNORE_`
	`MISFIRE_POLICY`
	
	• `MISFIRE_INSTRUCTION_FIRE_`
	`ONCE_NOW`
	
	• `MISFIRE_INSTRUCTION_DO_`
	`NOTHING`

When importing scheduled jobs, all calendar jobs with misfired triggers are re-executed. This results in unwanted and unplanned executions. This can be overcome by changing the Misfire policy on the server. But, that will impact all other jobs along with the calendar trigger.

To avoid changing the misfire policy globally for all jobs with trigger, two new properties `calenderTrigger.resetStartTimeOnImport` and `simpleTrigger.resetStartTimeOnImport` are added in the `js.quartz.properties` file. The new properties skip firing calendar jobs right after import, and job is executed at the next trigger (calendar or simple) time.

Configuring Scheduler Misfire Policy

Configuration File
`.../WEB-INF/js.quartz.properties`

Property	Description
`calenderTrigger.resetSt` `artTimeOnImport`	By default, `calenderTrigger.resetStartTimeOnImport=false`. If set to `true`, then while importing the job, start time is recalculated to the import time. And, the job execution starts in the future based on the condition set in calendar trigger.

Configuring Scheduler Misfire Policy

`simpleTrigger.resetStartTimeOnImport`	By defaults, `simpleTrigger.resetStartTimeOnImport=false`. If set to `true`, then job start time is refreshed to current time.

> 📝 For old jobs with simple triggers that were completed, but remained in the system, the jobs are executed again because the start time recalculated to the current time.

Configuring Scheduler Failure Notifications

By default, if a scheduled report runs but causes an error, the scheduler sends an email to the schedule owner and to all JasperReports Server administrators in the same organization. This is in addition to any failure notification addresses specified on the Notifications tab of the scheduler wizard. To receive these scheduler failure alerts, administrators must have valid email addresses defined in their user accounts.

You can also configure the scheduler to send failure notifications to different users based on roles, or turn off failure notifications.

Configuring Scheduler Failure Notifications

Configuration File

.../WEB-INF/applicationContext-report-scheduling.xml

Entry Key	Bean	Description
administrator Role	quartz Scheduler	This setting determines the role to which the scheduler failure notifications will be sent. All users in the organization with this role and a valid email address defined in their user profile will receive the email notification. By default, this setting is ROLE_ ADMINISTRATOR.
disableSending	quartz	Disables or allows failure notifications to be sent to

Configuring Scheduler Failure Notifications		
AlertToAdmin	Scheduler	the role in the previous setting. By default, this setting is false, meaning that notifications are sent. Set this value to true to disable scheduler failure notifications being sent to administrators (or the role defined above).
disableSending AlertToOwner	quartz Scheduler	Disables or allows failure notifications to be sent to the schedule owner. By default, this setting is false, meaning that notifications are sent. Set this value to true to disable scheduler failure notifications to the schedule owner.

Restricting File System Output

The scheduler outputs reports through several channels. Most reports are emailed, but reports can also be written to FTP folders. You can also configure the scheduler to write reports to the server's local file system. This option is disabled by default for security reasons.

> ⚠ If you turn on scheduler file system output, make sure you have configured user and folder access rights to make sure that malicious files cannot be written to your file system. The process that writes the files is the same user that runs the application server hosting JasperReports Server.

Scheduler File System Output
Configuration File
.../WEB-INF/applicationContext.xml

Property to Update	Description
`enableSaveToHostFS`	Set the value from "false" (the default) to "true".
	When true, the user interface for the scheduler displays active fields that allow the schedule creator to specify a folder in the server's file system. The scheduler will write

Scheduler File System Output

files to this location every time it runs the schedule for this report.

This property also determines the scheduler's overall access to the file system. When true, any schedule configured with a file system folder will write to the file system. When false, no scheduled reports will write output to the file system (FTP and email output are not affected). However, any file system output specified in a schedule remains defined and will again trigger file system output when this property is true again.

Removing Report Scheduling Interval Options

Note: The information covered in this section only applies to jobs with simple recurrence, not calendar recurrence type.

Users can schedule reports to run at regular intervals. The default interval can be expressed in weeks, days, hours or minutes. To prevent users from scheduling frequent reports, you can limit the intervals to days or hours by editing the following configuration file:

Scheduling Interval Options

Configuration File

.../WEB-INF/flows/reportJobBeans.xml

Section to Update	Description
recurrenceIntervalUnits	Comment out the intervals you want to disable.

To remove a temporal interval, enclose the corresponding bean in comment characters. For example, to prevent users from scheduling reports at minute intervals, comment out the bean containing the INTERVAL_MINUTE field:

```
<!--
<bean class="com.jaspersoft.jasperserver.war.dto.ByteEnum">
    <property name="code">
        <util:constant static-field="com.jaspersoft.jasperserver.api.engine.scheduling.
            domain.ReportJobSimpleTrigger.INTERVAL_MINUTE"/>
    </property>
    <property name="labelMessage">
        <value>job.interval.unit.minute.label</value>
    </property>
</bean>
-->
```

Adding a Holiday Exclusion Calendar

The scheduler supports exclusion calendars to specify days or times when no report should be run, even if scheduled. For example, you might not want a report to run on a bank holiday when the financial data would be meaningless.

The scheduler maintains a list of named calendars, and the user interface allows the schedule creator to select a calendar whose dates will be excluded from the schedule.

Currently, the only method to define a holiday calendar is through the REST API. You can use any browser plug-in that acts as a REST client and can send PUT requests to JasperReports Server. Using such a plug-in, compose and send the following REST request (header and body) to your server:

```
PUT http://<host>:<port>/jasperserver[-pro]/rest_v2/jobs/calendars/2014FrenchHolidays
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<reportJobCalendar>
    <calendarType>holiday</calendarType>
    <description>2014 French Holidays</description>
    <excludeDays>
        <excludeDay>2014-01-01</excludeDay>
            <excludeDay>2014-04-18</excludeDay>
        <excludeDay>2014-04-21</excludeDay>
        <excludeDay>2014-05-01</excludeDay>
        <excludeDay>2014-05-08</excludeDay>
        <excludeDay>2014-05-29</excludeDay>
        <excludeDay>2014-06-09</excludeDay>
        <excludeDay>2014-07-14</excludeDay>
```

```
            <excludeDay>2014-08-15</excludeDay>
            <excludeDay>2014-11-01</excludeDay>
            <excludeDay>2014-11-11</excludeDay>
            <excludeDay>2014-12-24</excludeDay>
            <excludeDay>2014-12-25</excludeDay>
        </excludeDays>
        <timeZone>GMT+01:00</timeZone>
    </reportJobCalendar>
```

For example, using the Poster plug-in for Firefox, you can submit this request as shown in the following figure. The figure also shows the successful reply from the server.
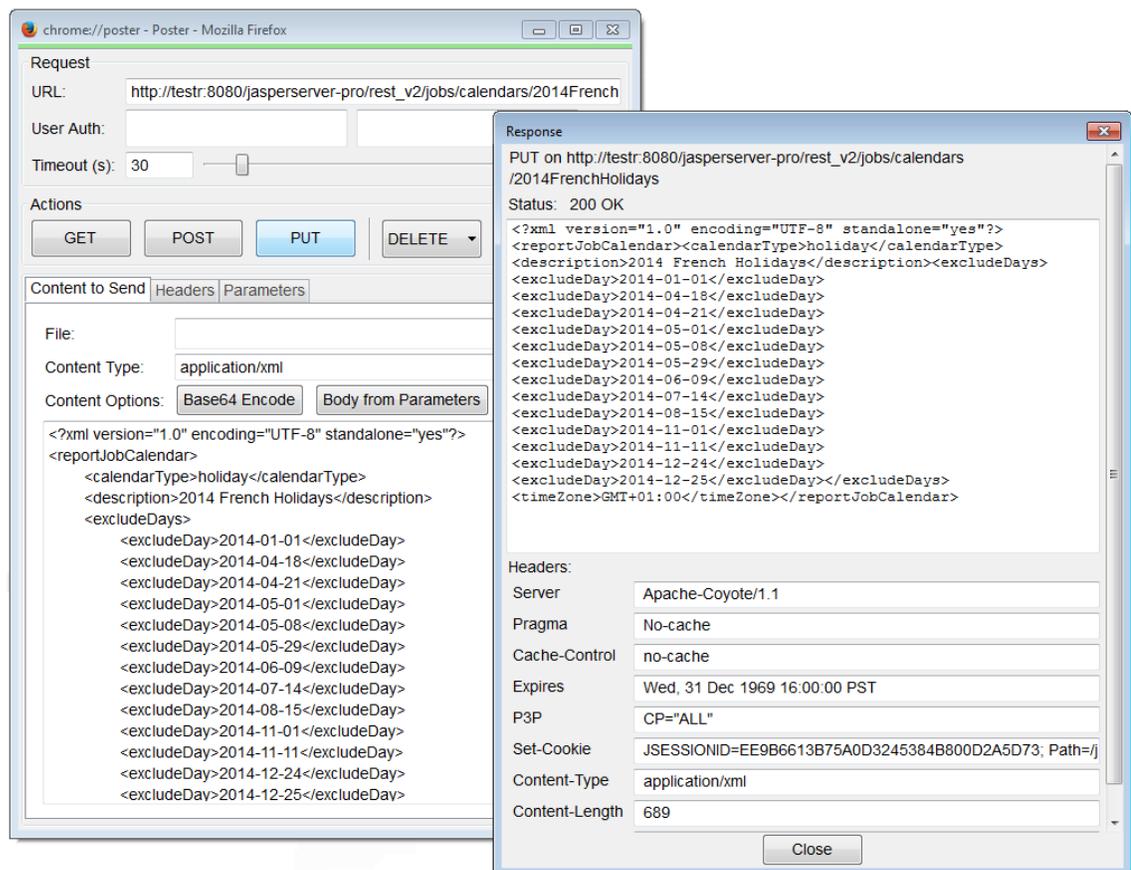
Figure 97: Creating a Holiday Calendar with REST Web Services

Then you should see your new calendar in the list of calendars in the Schedule tab.

Figure 98: Selecting a Custom Holiday Calendar in the Scheduler

The REST API supports other types of calendars, however, the user interface lists only calendars of type holiday. Using the REST API, you can create and manage any number of calendars and update any schedule to use them. For more information, see the JasperReports Server REST API Reference.

Changing the Default Output Folder

By default, the scheduler will save the output of scheduled reports to the directory in which the report resides and scheduled dashboard exports to the /public/Samples/Dashboards folder. You can change this default location to another folder in the repository by editing the applicationContext-report-scheduling.xml configuration file on the server.

To enable file data sources in the UI

Procedure

1. Open the file <js-install>/WEB-INF/applicationContext-report-scheduling.xml for editing.

2. Locate the element <util:map id="reportJobDefaults">.

3. Update <entry key="scheduler.job.repositoryDestination.folderURI" value="/job_output" /> and replace "/job_output" with the URI for the new default folder in the repository.

4. Restart the server or redeploy the JasperReports Server web app. The new default folder appears on the Output Options tab when you try to schedule a report or dashboard.

Configuring the Scheduler for Dashboards

You can configure the following property to enable the scheduler and headless browser to load the dashboard on the server side and export it.

Configuring the Scheduler for Dashboards	
Configuration File	
.../WEB-INF/js.config.properties	
Property	Description
deploy.base.local.url	This property is used by scheduler and headless browsers to load the dashboard on the server side and export it. **Note**: You will need to set this property when the **Export** option "detailed" is selected. Set this property to the localhost URL. This URL must include the application name. For example: http://localhost:8080/jasperserver-pro Specify http or https in your URL.

Configuring the Scheduler to Fetch Attributes

You can configure the following property to enable the scheduler to fetch attributes.

Configuring the Scheduler to Fetch Attributes
Configuration File
.../WEB-INF/js.quartz.properties

Configuring the Scheduler to Fetch Attributes	
Property	Description
notification.service.multiTenant.config	This property is used by scheduler to fetch attributes. This property can be set to any of the following values:
	• none
	• tenant
	• hierarchical
	By default, notification.service.multiTenant.config=none
	If this property is set to hierarchical, the attribute is referred across all levels, starting from the user level. The server searches for an attribute with the given name in the following order, stopping and returning the first value that it finds:
	• At the user level, the server searches the attributes of the logged-in user and creates the bean with those attributes.
	• If attribute is not found at user level, the server searches at organization or tenant level. in the organization attributes of the logged-in user's organization. The server also searches in all parent organizations and creates the bean with those attributes.
	• If attributes are not found at user and organization or tenant level, attributes are searched at the server level and create the bean with those attributes.

If properties are configured for tenant or hierarchical, the system creates beans (JavaMailSender and GraphServiceClient) with tenant-specific or hierarchical-specific attributes, respectively. Else, it creates beans using the default properties configured in the js.quartz.properties file.

Setting Properties in the jasperserver_ config.properties File

You can set the following properties while composing emails:

Property	Description
graph.service.body.scheduler.template=<html><head><meta http-equiv='Content-Type' content='text/html; charset=utf-8'><title>My Page</title></head><body> ${linkDescription} ${resourceURI} ${preamble}</body></html>	Defines the HTML template used for composing emails with links when scheduling jobs or alerts in the scheduler using graph API.
graph.service.body.scheduler.notification.template = ${preamble} ${jobinfo} Label = ${label} ID = ${id} Description = ${description} Status = ${status} 	Defines the template used for composing notification mails when scheduling notification using graph API.
mail.service.body.scheduler.notification.template = ${preamble}${jobinfo}\nLabel = ${label}\nID = ${id}\nDescription = ${description}\nStatus = ${status}${preamble}	Defines the template used for composing notification mails when scheduling notification using mail.

📝 The template used to composing notification mails contains placeholders for dynamic content to be added. The placeholders include ${label}, ${jobinfo}, ${id}, ${description}, and ${status}.

Configuring Report Thumbnails

The JasperReports Server REST API has the ability to generate and export thumbnail images (small preview images) of reports and dashboards. The JasperMobile apps for Android and iOS use this feature to display small tiles with the report or dashboard image.

For more information about the REST API, see the JasperReports Server REST API Reference.

By default, report thumbnails are not active. If you use the JasperMobile apps with your server, you should turn on the report thumbnails:

Report Thumbnails	
Configuration File	
.../WEB-INF/js.spring.properties	
Property	Description
property.reportThumbnailServiceEnabled=false	When this property is set to true, JasperReports Server generates and sends report thumbnails in response to REST API requests. When it is set to false, report thumbnails are not generated. The default is false.

Configuring the Heartbeat

During installation (or the first time you log in as administrator), you're prompted once to participate in the Heartbeat program, which reports technical information to the JasperReports Server product team about your implementation, such as the operating system, JVM, application server, database (type and version), data source types, and server edition and version number.

If you change your mind, you can change the heartbeat behavior by editing the following configuration file:

Heartbeat Options	
Configuration File	
.../WEB-INF/js.config.properties	
Property	Description

Heartbeat Options	
heartbeat.enabled=true	When this property is set to true, JasperReports Server reports information about your environment to us once a week. When it is set to false, information is not sent.
heartbeat.askForPermission.enabled	Determines whether the administrator is prompted (the next time he logs into the web UI) about whether to allow heartbeat data to be sent. Typically, there is never cause to edit this property directly.
heartbeat.permissionGranted.enabled	Indicates whether a user has granted the server permission to send Heartbeat data. Setting this property to false prevents data from being sent.

All of these settings are properties that are substituted into the heartbeatBean in the .../WEB-INF/applicationContext-heartbeat.xml file.

Configuring the Online Help

JasperReports Server professional edition includes an online help system that describes the web interface. If your users don't have Internet connectivity, or if you don't want to provide access to this system, you can configure the server to hide the help links completely.

Online Help Configuration Options		
Configuration File		
.../WEB-INF/applicationContext-webHelp.xml		
Property	Bean	Description
showHelp	webHelp	Determines whether the help links are displayed in the

Online Help Configuration Options		
		JasperReports Server web UI. Valid values are true and false.
		The Help link appears at the top right corner of the web UI's pages.
hostURL	webHelp	Indicates the name of the computer hosting the web server where the help is running. The value depends on the version of JasperReports Server. Do not change this value.
pagePrefix	webHelp	Defines the default page name to pass to the web server hosting the help system. The only valid value is Default_CSH.htm for this property.
helpContextMap	webHelp	Maps contexts in the application to topic identifiers in the help system. Many pages in the web application are configured for context-sensitivity. When a user clicks Help on such a page, JasperReports Server loads a specific topic in the help system. The topic that appears is determined by a map in the applicationContext-webHelp.xml file. The only valid values are the defaults.

Configuring JasperReports Web Studio Access

JasperReports Web Studio is the visual designer for creating and editing report templates for the JasperReports Library reporting engine and the whole Jaspersoft family of products that use this open source library to produce dynamic content and rich data visualizations. It comes as a web-based alternative to Jaspersoft Studio, the desktop application, which is the most complete and powerful designer for JasperReports templates.

JasperReports Web Studio can be run on its own or added to an existing JasperReports Server 8.2.0 instance as a pluggable report editing feature.

When JasperReports Web Studio is added to an existing JasperReports Server instance, two new JasperReports Web Studio applications jrio and repo are deployed along with jasperserver-pro in the application server.

jrio enables you to render reports and is used by JasperReports Web Studio for report preview.

repo is a JasperReports Server repository plug-in used by JasperReports Web Studio to store reports and files.

JasperReports Server and JasperReports Web Studio jrio/repo deployment

JasperReports Web Studio jrio and repo and JasperReports Server can be deployed either in a public network or in a private network.

In a public network, the JasperReports Server and JasperReports Web Studio jrio and repo are deployed on the same application server. jrio and repo are connected to JasperReports Server using an internal proxy server. JasperReports Server can directly access jrio and repo.
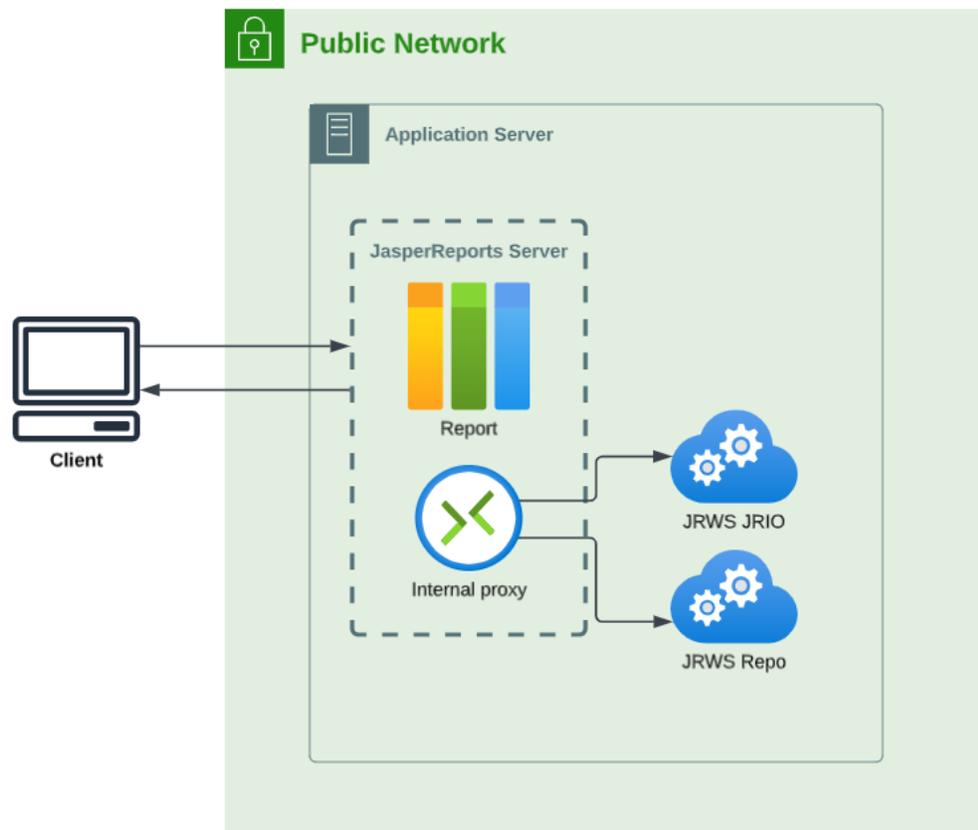
Figure 99: Deployment on public network

In a private network, the JasperReports Server and JasperReports Web Studio jrio and repo are deployed on remote application servers, connected using an internal proxy server. This is useful when deploying in a cluster where jrio and repo applications can be accessed only by JasperReports Server instance , and not users.

JasperReports Server can access jrio and repo via backend. Hence, the connection between the JasperReports Server instance and jrio and repo must be set.
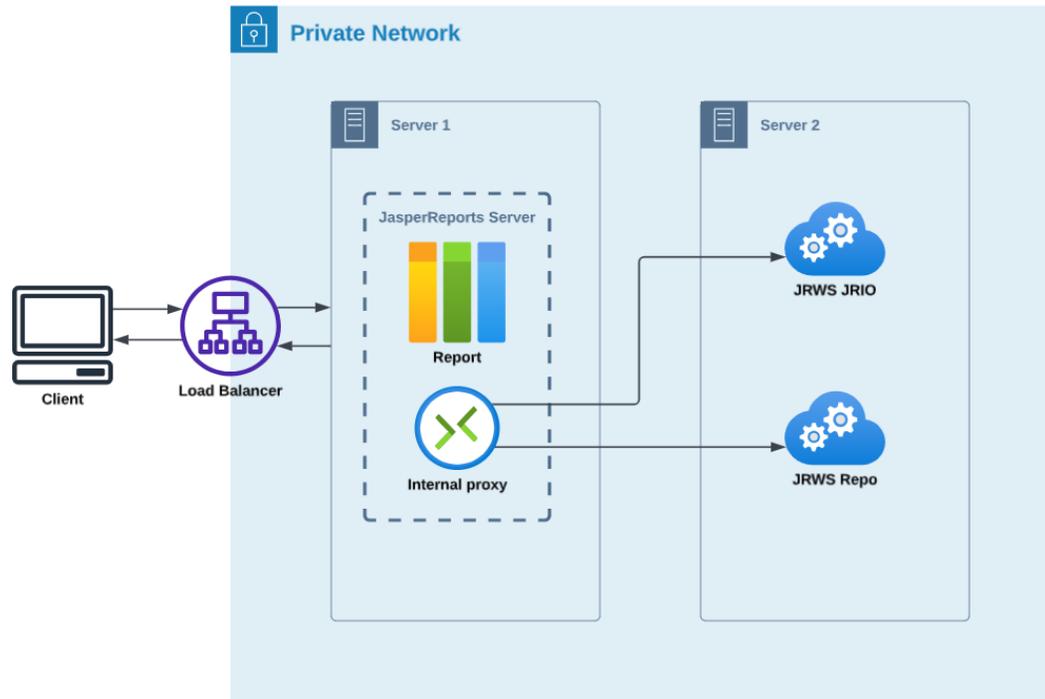
Figure 100: Deployment on private network

The following table describes the JasperReports Web Studio properties:

JasperReports Web Studio Properties	
Configuration File	
`.../WEB-INF/js.config.properties`	
Property	**Description**
jrws.jrio.url	This URL is used to control where jrio is deployed. By default, the value is http://localhost:8080/jrio
	You can set this property for JasperReports Server by providing an environment variable, using the following syntax:
	JRWS_JRIO_URL=http://<hostname>:<port>/jrio

JasperReports Web Studio Properties	
jrws.repo.url	This URL is used to control where repo is deployed. By default, the value is http://localhost:8080/repo
	You can set this property for JasperReports Server by providing an environment variable, using the following syntax:
	JRWS_REPO_URL=http://<hostname>:<port>/repo

If environment variables are provided, they take the precedence and the values set in the js.config.properties file are not considered.

You can deploy the jrio and repo applications on separate server, or service too. It is useful when JasperReports Server cluster is deployed and you want to keep jrio and repo applications aside on a separate service or instance.

Configuring JasperReports Web Studio Access

To disable access to JasperReports Web Studio access based on role, remove that role from the following property, located in the *…/WEB-INF/applicationContext-security-pro-web.xml* file:

```
<security:intercept-url pattern="/webstudio/**" access="ROLE_USER,ROLE_
ADMINISTRATOR" />
```

OpenTelemetry

OpenTelemetry (OTel) is an observability framework and toolkit designed to create and manage telemetry data such as traces, metrics, and logs. It is a collection of APIs, SDKs, and tools to instrument, generate, and export telemetry data. It provides an overview of the performance and behavior of your application. OTel analyzes the internal state of an application by collecting the data, analyzes the data(or logs), and shows the traces of the logs in Jaeger.

OpenTelemetry performs the following functionality:

- **Instrument data** - The data is instrumented by adding Annotations or new lines in the code, which help to instrument the code. The instrumentation can be either Manual or Automated. For more information, see Instrumentation.

- **Generate data** - The instrumented data is captured using a javaagent jar, which is specific to the Java language. The javaagent jar collects the instrumented data emitted by the code.

- **Export data** - The collected data, which is emitted by the code is then exported and visualized using the Jaeger exporter tool.

Traces

Traces are helpful in finding out the reason behind the slow performance of the application. Traces help you to understand the complete path and find out what happens when an API request is made to an application. It captures the exact spans or places in the code where the problem occurred.

Why tracing is necessary?

Tracing helps to investigate performance problems with JasperReports Server. Typically, we have noticed that the response time for running a report in JasperReports Server application was taking longer than usual. This problem might be because of the slow query, which is taking time to execute and deliver back the response. To discover the root cause of the problem in JasperReports Server, it is necessary to implement tracing.

For example, an application performance is slow than usual for a longer time. An application can be slow due to any of the reasons such as it took a lot of time to retrieve data from the network or database took some time to run queries. To find out the reason behind an application slow performance, logs are analyzed to see if there are any errors.

When no errors could be found in the logs, and the root cause remain unknown, then enabling tracing helps to detect the unknown issues. Tracing identifies the slow queries, captures the exact places (spans), and the time spent in the specific area of the code. Therefore, configuring OTel optimizes all such unknown issues and helps in improving the overall code quality and performance of the application.

You can configure OpenTelemetry in the JasperReports Server to overcome the performance issues in the application by enabling tracing.

See Use cases to understand how enabling tracing helped to improve the performance of the JasperReports Server application for Scheduler and Input Control dialog.

Before enabling tracing, you must first download and run a third-party agent, collector, and aggregator. For more information, see Configuring OpenTelemetry and Jaeger Agent.

Spans

Spans represent smaller chunks of operation or a unit of work performed within an API request. All operations within an API request are considered as span. OpenTelemetry uses a visualization tool called Jaeger to view the smaller chunks of operation or spans. Spans include the total API execution time, start time of methods, and hidden errors of the code in the logs.

For more information on OpenTelemetry, visit the official website of https://opentelemetry.io/docs.

Use cases

OpenTelemetry can be used to trace issues of Scheduler and Input Controls. It can be used to trace logs and capture errors of the Scheduler while running a report. This section includes screenshots of the traces captured in JasperReports Server during report execution in the Jaeger UI.

Tracing Report execution triggered by Scheduler

The following figure shows an example of the errors captured for the JasperReports Server Scheduler in Jaeger UI along with the logs. It shows the list of errors captured in Jaeger UI. It includes details of span name, API execution time, span execution time, start time of logs. On expanding the log for an error, you can see the exception caused in each log along with the traces.
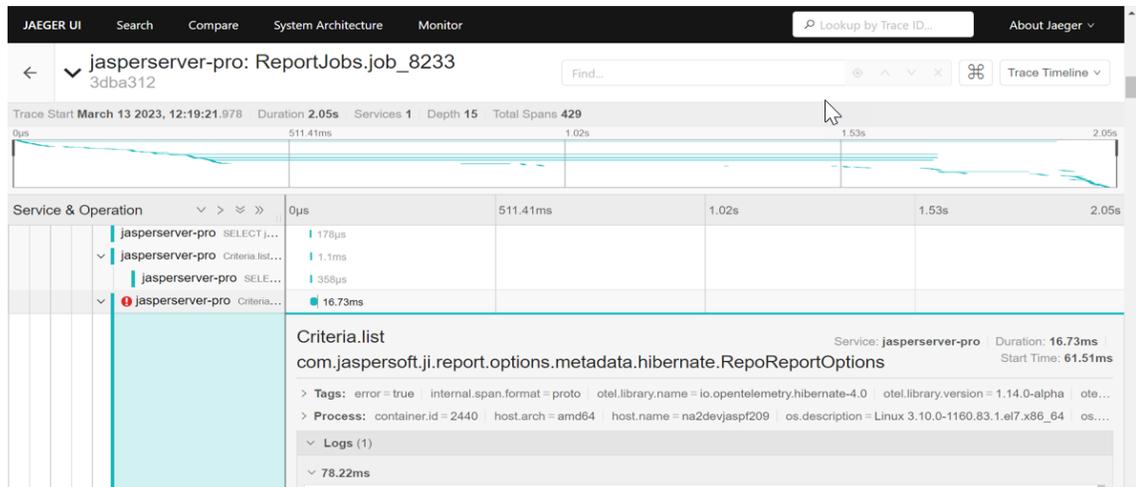
Figure 101: Example of Error Captured by OpenTelemetry in Jaeger UI
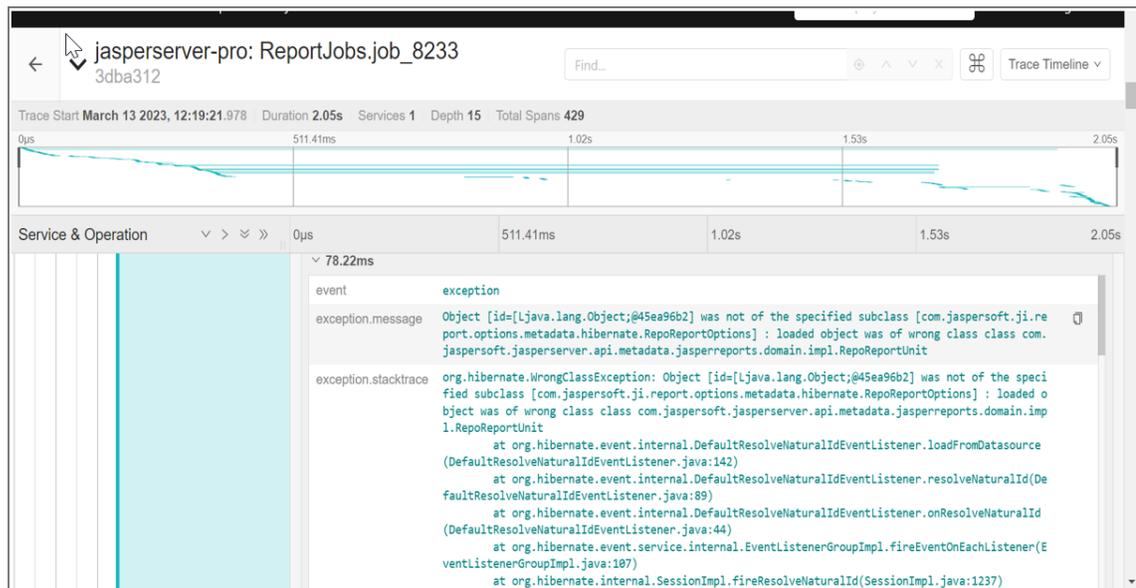
Figure 102: Example of Logs Captured by OpenTelemetry in Jaeger UI

Tracing the execution of Report with Input Controls

Enabling tracing of the report execution helps to find slow SQL queries, and can help to determine if these queries were sent by report or Input Controls. In Jaeger UI you can look for slow queries by checking the start time, total execution time, and the duration of each span.

To enable tracing, you need to configure OpenTelemetry and Jaeger agents on the JasperReports Server application and cluster. For more information, refer to Configuring OpenTelemetry and Jaeger Agent.

OpenTelemetry Architecture

This section provides an overview of the setup of the OpenTelemetry framework in JasperReports Server application. This framework shows how JasperReports Server is deployed in the cluster mode to generate and view traces in Jaeger. Before configuring the OpenTelemetry framework for the JasperReports Server application, you should be familiar with the components of OTel agent and Jaeger.

The OpenTelemetry framework consists of the following components:

- **OTel agent** - The OTel agent resides in the JasperReports Server application, which helps to generate traces.

- **Tracer** - Tracer is a pipeline, which is responsible for starting the tracer.

- **Span processor** - The span processor is responsible for each span lifecycle, which processes the spans.

- **Span exporter** - The span exporter exports the data processed by the span processor to the configured endpoint.

- **OpenTelemetry collector** - The OpenTelemetry collector is used to load balance or streamline the traces collection.

- **Jaeger** - Jaeger is an open-source distributed tracing system tool to visualize traces.

- **Jaeger collector** - The Jaeger collector is an endpoint, which receives traces from Jaeger agents and runs them through a processing pipeline.

- **Jaeger query** - Jaeger query is a service that retrieves traces from the database(or storage) and hosts a UI to display them.

OpenTelemetry Deployment

JasperReports Server is instrumented to generate traces using the OpenTelemetry framework.

When JasperReports Server is instrumented with the OTel agent, the agent resides in the JasperReports Server application and generates traces using the OpenTelemetry framework. The Tracer and Span pipeline starts with the Tracer and Span processor. The Span processor is responsible for each span lifecycle, and it processes the spans and exports them to the configured endpoint.

There are two options to configure the exports to Jaeger:

- **Configuring directly to the Jaeger collector**: In this option, the Jaeger collector endpoint is directly configured and traces can be directly collected at Jaeger.

- **Configuring via OpenTelemetry collector**: In this option, an OpenTelemetry collector is configured between JasperReports Server exporter and Jaeger collector. This helps to load balance or streamline the traces collection (This is completely optional).

The following illustration shows the architecture diagram of JasperReports Server instrumentation.
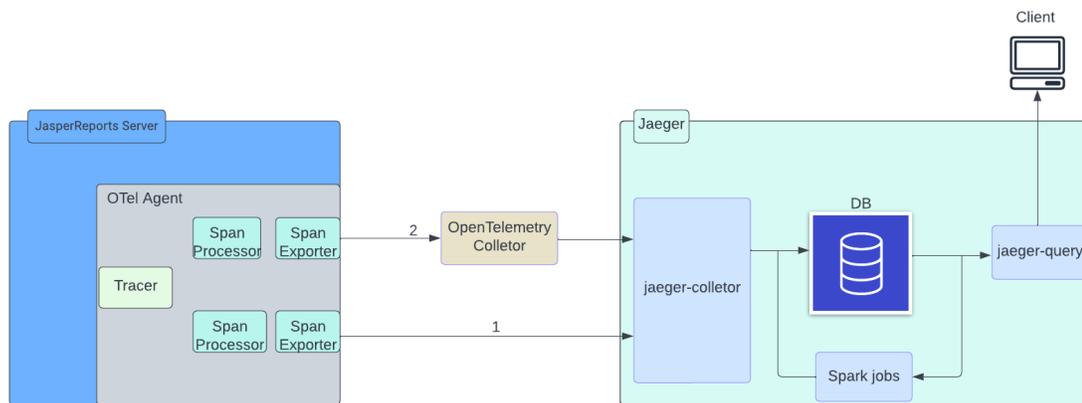
Figure 103: OpenTelemetry Framework in JasperReports Server

Instrumenting JasperReports Server for Tracing

Instrumentation

Instrumentation is the process of adding annotations or new lines in the code to view traces when an issue has occurred within the application. It helps to identify major issues within the code by instrumenting specific areas in the code.

There are two types of instrumentation:

- **Auto Instrumentation** - Auto instrumentation uses the opentelemetry-api to view traces within the code. It creates a span for a method using the @WithSpan annotation for which it is annotated with within the code.

- **Manual Instrumentation** - Manual instrumentation involves manually adding new lines of code. It uses opentelemetry-api and opentelemetry-sdk to view traces within the code.

The following span name and properties were instrumented in the code to view traces of spans in Jaeger:

- @WithSpan - Adds annotation to view method names in Jaeger UI. This is an auto-instrumentation process for viewing method names.

- Span span = Span.current() - It helps to create a span. This is a manual instrumentation process for creating span names.

- span.updateName() - It changes the name of the method that appears in Jaeger.

- span.setAttribute() - It filters out the spans in Jaeger based on the attributes.

- export OTEL_INSTRUMENTATION_HIBERNATE_ENABLED=false - It suppresses the unwanted hibernate spans, which resulted in increasing the number of spans by setting this value to false in tomcat\bin.

Configuring OpenTelemetry and Jaeger Agent

You can configure the OpenTelemetry and Jaeger agent to view the traces in the JasperReports Server application and JasperReports Server cluster.

The configuration of the OpenTelemetry and Jaeger agent includes the following steps:

Procedure

1. Download opentelemetry-javaagent.jar.

2. Download platform-dependent **OpenTelemetry Collector** from
   https://github.com/open-telemetry/opentelemetry-collector-releases/releases.

3. Download Jaeger Tracing from https://www.jaegertracing.io/download/ and run the
   Jaeger aggregator.

To configure step by step, see Setting up OpenTelemetry and Jaeger on JasperReports
Server Application and Setting up OpenTelemetry and Jaeger on JasperReports Server
Cluster.

Setting up OpenTelemetry and Jaeger on JasperReports Server Application

This section includes the steps for setting up the OpenTelemetry and Jaeger agent on the
JasperReports Server application. This configuration helps to view traces and spans of the
JasperReports Server application.

Prerequisites

The below prerequisites must be satisfied before configuring OpenTelemetry and Jaeger on
the JasperReports Server application:

- JasperReports Server setup.

To configure OpenTelemetry and Jaeger in the JasperReports Server, do the following

Procedure

1. Stop Tomcat service.

2. Download opentelemetry-javaagent.jar and then place this javaagent on the
   JasperReports Server node. The javaagent involves some useful libraries, which help
   in instrumenting the code.

3. Go to the Tomcat bin directory of each JasperReports Server node and update the

following arguments in the setenv.sh file of the Tomcat bin:

- export JAVA_OPTS: Provide the correct path of the javaagent where the javaagent is placed.

- export OTEL_TRACES_EXPORTER: Provide Jaeger exporter name as we are using Jaeger exporter to visualize the trace. You can provide the name of the respective exporter that you are using to visualize the traces.

- export OTEL_EXPORTER_JAEGER_ENDPOINT: Provide the correct hostname with port number. If you are using localhost, then provide the localhost name else provide the hostname directly.

- export OTEL_RESOURCE_ATTRIBUTES=service.name: In service name, provide jasperserver-pro service name.

- export OTEL_JAVAAGENT_DEBUG: The default value is false for this argument and is not editable.

- export OTEL_METRICS_EXPORTER: The default value is none for this argument and is not editable.

```
export JAVA_OPTS="$JAVA_OPTS -javaagent:/opt/jaeger/opentelemetry-javaagent.jar"
export OTEL_TRACES_EXPORTER=jaeger
export OTEL_EXPORTER_JAEGER_ENDPOINT=http://hostname:14250
export OTEL_RESOURCE_ATTRIBUTES=service.name=jasperserver-pro
export OTEL_JAVAAGENT_DEBUG=false
export OTEL_METRICS_EXPORTER=none
```

Ensure that the correct path of the javaagent and hostname is provided in the stenv.sh file.

4. Download Jaeger using https://www.jaegertracing.io/download/ on the JasperReports Server node. As a result, Jaeger gets downloaded in ZIP format.

5. Extract the Jaeger package and go to the location under which the jaeger-all-in-one is placed.

6. Open the command prompt and run ./jaeger-all-in-one --collector.zipkin.host-port=:9411 & command to start the Jaeger application.

Before setting up the javaagent and Jaeger file, ensure that the root user has setup OTel and Jaeger in the respective folders and assigned permissions for the jasperserver user. If no files are set up, then the application misbehaves. In such cases, start the service by logging as a jasperserver user.

7. Start the Tomcat service to access Jaeger using the JasperReports Server URL: http://<hostname>:16686 to see traces of the JasperReports Server application node. As a result, Jaeger gets successfully configured with JasperReports Server and Jaeger running in two separate tabs in the browser.

> To view traces of the load balancer, replace the hostname(http://<hostname>:16686) with the load balancer URL.

Setting up OpenTelemetry and Jaeger on JasperReports Server Cluster

This section includes steps for setting up the OpenTelemetry and Jaeger agent on the JasperReports Server cluster environment. This configuration helps to view the traces and spans for each node of the JasperReports Server cluster.

Prerequisites

The below prerequisites must be satisfied before configuring OpenTelemetry and Jaeger on the JasperReports Server cluster environment:

- JasperReports Server cluster setup.

To configure OpenTelemetry and Jaeger in the JasperReports Server cluster environment, do the following

Procedure

1. Stop Tomcat service.

2. Download opentelemetry-javaagent.jar and then place this javaagent on all the JasperReports Server User Guidenodes of the cluster. The javaagent involves some useful libraries that help in instrumenting the code.

3. Go to the Tomcat bin directory of each JasperReports Server node and update the following arguments in the setenv.sh file of the Tomcat bin:

   - export JAVA_OPTS: Provide the correct path of the javaagent where the javaagent is placed.

- export OTEL_TRACES_EXPORTER: Provide Jaeger exporter name as we are using Jaeger exporter to visualize the trace. You can provide the name of the respective exporter that you are using to visualize the traces.

- export OTEL_EXPORTER_JAEGER_ENDPOINT: Provide the correct hostname with port number. If you are using localhost, then provide the localhost name else provide the hostname directly.

- export OTEL_RESOURCE_ATTRIBUTES=service.name: In service name, provide jasperserver-pro service name.

- export OTEL_JAVAAGENT_DEBUG: The default value is false for this argument and is not editable.

- export OTEL_METRICS_EXPORTER: The default value is none for this argument and is not editable.

> Ensure that the correct path of the javaagent and hostname is provided in the stenv.sh file.

```
export JAVA_OPTS="$JAVA_OPTS -javaagent:/opt/jaeger/opentelemetry-javaagent.jar"
export OTEL_TRACES_EXPORTER=jaeger
export OTEL_EXPORTER_JAEGER_ENDPOINT=http://hostname:14250
export OTEL_RESOURCE_ATTRIBUTES=service.name=jasperserver-pro
export OTEL_JAVAAGENT_DEBUG=false
export OTEL_METRICS_EXPORTER=none
```

> In the above code snippet, ensure the OTEL_EXPORTER_JAEGER_ENDPOINT option should have the URL of the JasperReports Server Load balancer and the -javaagent option should have the correct path where the OpenTelemetry javaagent is stored.

4. Login to Load balancer node and download Jaeger on the JasperReports Server node using - https://www.jaegertracing.io/download/. As a result, Jaeger gets downloaded in ZIP format.

5. Extract the Jaeger package and go to the location under which the jaeger-all-in-one is placed.

6. Open the command prompt and run ./jaeger-all-in-one --collector.zipkin.host-port=:9411 & command to start the Jaeger application.

> Before setting up the javaagent and Jaeger files, ensure that the root user has setup OTel and Jaeger in the respective folders and assigned permissions for the jasperserver user. If no files are set up, then the application misbehaves. In such cases, start the service by logging as a jasperserver user.

7. Start the Tomcat service to access Jaeger using the JasperReports Server URL: http://<hostname>:16686 to see traces of the JasperReports Server application node. As a result, Jaeger gets successfully configured with JasperReports Server and Jaeger running in two separate tabs in the browser.

> To view traces of the Load balancer, replace the hostname (http://<hostname>:16686) with the load balancer URL.

By default, Jaeger shows traces of all the nodes of the JasperReports Server cluster. To view the spans and traces of a specific node of the cluster, segregate the traces of the specific node using the host.name tag under the tags option in Jaeger UI.

For example, to view traces of the specific node in the Jaeger UI, enter the "host.name=na2devjaspf831" tag under the tags option. As a result, the traces and spans of the mentioned node having the " host.name=na2devjaspf831" tag is displayed.

Methods for Report Scheduling and Report Execution

In our current implementation, we have changed the Scheduler method names for scheduling and execution of the report. These method names are also known as span names of the Scheduler. By default, the method names are the combination of <className>.<methodName>. You can modify or rename the span name. Based on the changed name, the updated names appear in the Jaeger UI.

The below table includes the list of the changed span or method names of Scheduler. With the changed method names, you can now search traces of logs specific to report scheduling and execution in Jaeger UI.

Scheduler Method Names for Report Scheduling and Execution

Original Method Names	Updated Method Names
JobsJaxrsService.scheduleJobWithProcessedParameters	scheduleJob

Original Method Names	Updated Method Names
JobsServiceImpl.scheduleJob	scheduleJobService
ReportSchedulingFacade.scheduleJob	scheduleJobDAO
HibernateReportJobsPersistenceService.saveJob	scheduleJobDAOImpl
ReportJobsQuartzScheduler.scheduleJob	scheduleQuartzJob
ReportExecutionJobMailNotificationImpl.sendMailNotification	sendMailNotification
ReportExecutionJobAlertImpl.sendAlertMail	sendAlertMail
InternalReportExecutor.createOutputs	createReportOutput
`InternalReportExecutor.executeReport`	executeReportInternal
`EngineServiceImpl.java.runReport`	runReport

The following figure shows the list of span names of Scheduler for report scheduling in Jaeger UI. You can search a span name by searching "schedule" in Operations to view span names specific to report scheduling in Jaeger UI.
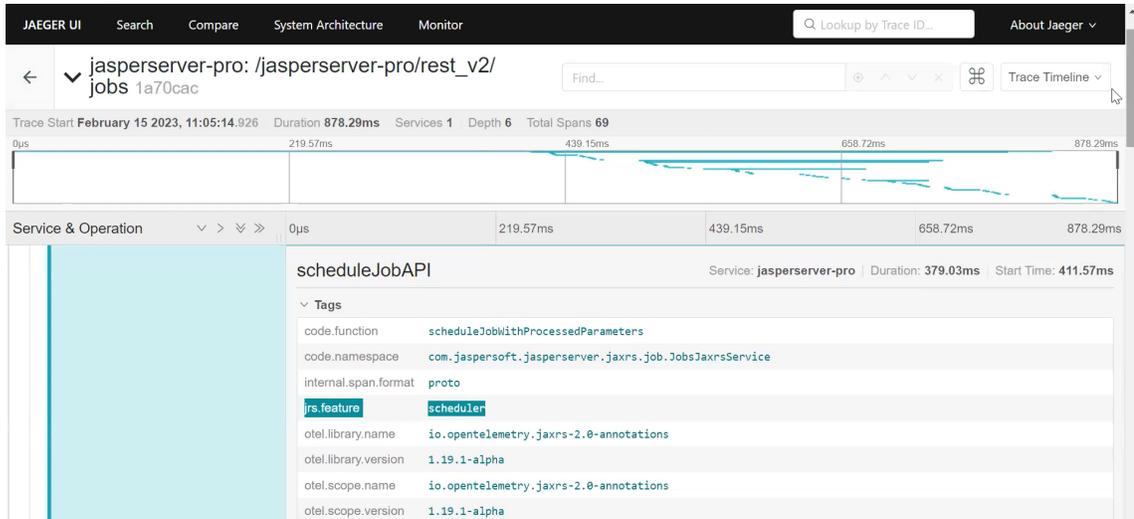
Figure 104: Scheduler Span Names for Report Scheduling

The report execution happens when a job is scheduled for a report. The following figure shows the list of span names of Scheduler for report execution in Jaeger UI. You can search a span name by searching "execute" in Operations to view span names specific to report execution in the Jaeger UI.
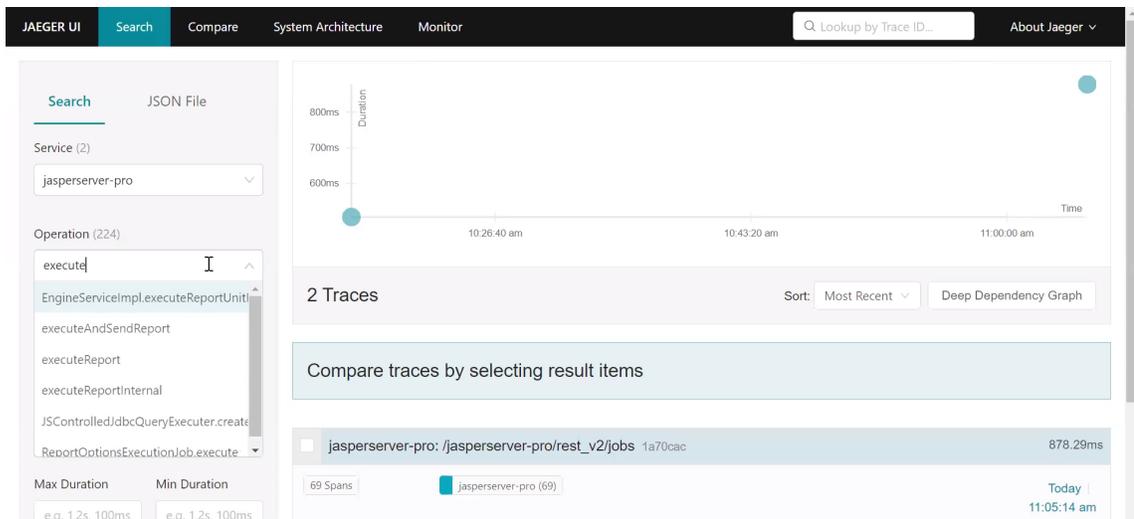
Figure 105: Scheduler Span Names for Report Execution

Viewing and Analyzing Traces

The section describes the process of searching, sorting, and reading traces of logs in the Jaeger UI. Consider an example of the rest_v2/jobs API service, which gets triggered to create a scheduled report execution. The below section includes steps to search, sort, and read traces of Scheduler for the rest_v2/jobs API service.

Viewing and Reading Traces

The following figure shows various attributes of spans in the Jaeger UI.
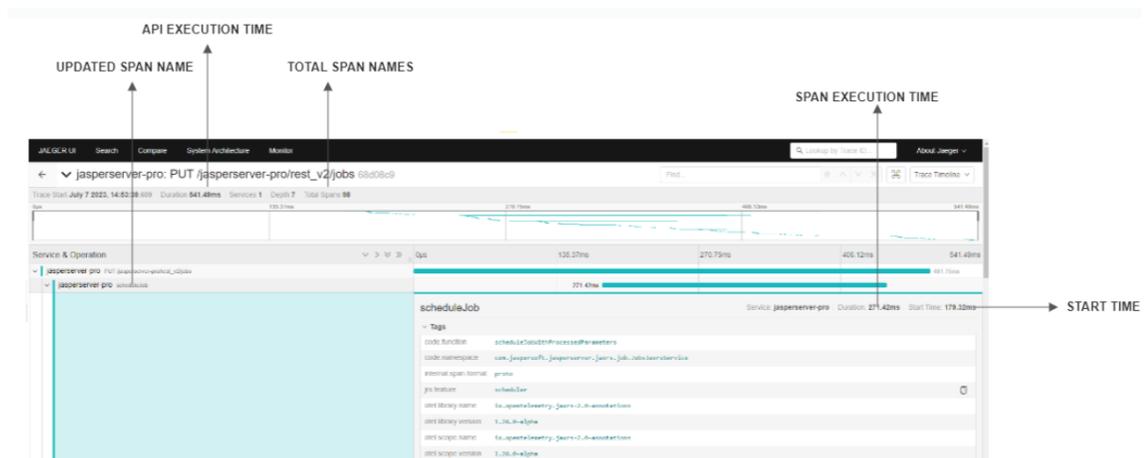
Figure 106: Viewing Spans on Jaeger

Example of Viewing traces of Report Scheduling

To search and read traces of the scheduled report in Jaeger UI, do the following:

Procedure

1. Login to JasperReports Server and create a schedule (For more information, see JasperReports Server User Guide).

2. Login to Jaeger UI using http://<hostname>:16686.

3. Choose the scheduleJobAPI method name in operation. This method is responsible

for scheduling a report.

4. Click Find Traces.

Figure 107: Viewing Traces of Report Scheduling

5. Choose from any result items to view traces of rest_v2/jobs.

For filtering and viewing only Scheduler methods you can search jrs.feature="scheduler" in the tags. This filters out the span names of only Scheduler methods and displays them in the Jaeger UI.

The following figure shows the traces of report job and scheduler. Choose any trace and then click Tags, to see which tag was added for filtering Scheduler traces.

Figure 108: Filtering Traces of the Scheduler

Example of Viewing traces of Report Execution

To search and read traces of report execution in Jaeger UI, do the following:

Procedure

1. Login to JasperReports Server and create a schedule (For more information, see JasperReports Server User Guide).

2. Login to Jaeger UI using http://<hostname>:16686.

3. Choose the executeReport method name in operation. This method is responsible for executing a report.

4. Click Find Traces.

Figure 109: Viewing Traces of Report Execution

Suppressing Span

Suppression of unwanted spans is required to remove the unnecessary database calls, which gets added in traces. While viewing traces in Jaeger, we observed unnecessary database calls, which resulted in increasing the span number within the trace.

You need to suppress the spans to remove the unwanted hibernate spans that are not required to view only relevant spans. For suppressing span, instrument the span by setting export OTEL_INSTRUMENTATION_HIBERNATE_ENABLED=false in tomcat/bin.

Server Diagnostics

JasperReports Server provides comprehensive diagnostics so administrators can monitor the health and performance of the running server. The following tools can help you troubleshoot issues and analyze performance:

- System logs capture any errors in the code running the server, but can be configured to capture more information for debugging. If you suspect a problem or cannot pinpoint an error, you can turn log collectors on and off to capture logs while you test your issue.

- Auditing stores key events of interest to system administrators, such as login time, log out time, user, report generated, report details, and object sizes. The audited events can be saved and moved to an archive automatically after a specified number of days.

- Monitoring is based on audit events and provides a multi-dimensional Domain to create Ad Hoc views. This helps you generate reports that contain key metrics such as which reports use the most resources. It allows you to find places to optimize your data sources and your reports.

- Diagnostics provide run-time data that shows the overall health of your server instance. For example, it includes values such as logged in users, currently running reports, scheduled reports, and memory use and performance in the Ad Hoc cache. The diagnostic data is exposed in a custom data source and can be easily viewed in a report or integrated with industry standard management consoles.

This chapter contains the following sections:

- Configuring System Logs
- Using Log Collectors
- Auditing and Monitoring Events
- Configuring Auditing and Monitoring
- Using the Audit Data
- Using the Monitoring Data
- Importing and Exporting Event Data

- Real-Time Diagnostics

- Exposing Diagnostics Through Jaspersoft's JMX Agent

- Using the Diagnostic Data in Reports

- Excluding Diagnostic Attributes

- Disabling Real-Time Diagnostics

- Server Monitoring

Configuring System Logs

The log files contain important information about how the server is running. If you suspect a problem or cannot pinpoint an error, you can change log levels to capture verbose messages while you test your issue.

JasperReports Server now uses the Apache Log4j 2 package to generate log files. JasperReports Server also uses the SLF4J facade (Simple Logging Facade for Java) and the Log4j 2 SLF4J Binding.

- The default log file is .../WEB-INF/logs/jasperserver.log.

- The default log configuration file is .../WEB-INF/log4j2.properties.

To view the log file, you must have access to the file system where JasperReports Server is installed. This section describes the settings that control what information JasperReports Server writes to its logs.

> Because of the Log4j 2 upgrade, JasperReports Server logs now have a slightly different time format than versions 7.2 and before. If necessary, you can change the output format by configuring properties in the log4j2.properties file. For information about date and time formats, see http://logging.apache.org/log4j/2.x/manual/layouts.html#PatternLayout.

Log Levels

A logger is a Java class that implements a logging interface for sending messages related to internal events of the server. The various features or internal modules of the server have separate loggers that can be controlled independently. The level of a logger determines the number and types of events that it writes to the log file.

Log Level	Amount of Information
ERROR	Write's minimal information to the log, only when describing serious program faults.
WARN	Write's error and warning messages to the log. Warning messages contain cautionary information to help you to decide whether the logged events require your attention.
INFO	Write's error, warning, and informational messages to the log describing significant events, such as those that affect application performance.
DEBUG	Write's error, warning, informational, and additional messages to the log. Debug messages are very detailed and often voluminous. Use this setting only to diagnose a problem. DEBUG can impact system performance and should not be used in production environments. If several loggers are set to DEBUG, the server may generate huge logs, and performance can suffer.

Log levels may be set in two places, either the log4j2.properties configuration file or the Log Settings UI page, or both. Because there may be multiple settings for a given logger, the effective level is determined in the following order:

1. The root log level defined in log4j2.properties and inherited by all loggers. By default, it is set to ERROR, the least verbose.

2. A logger level defined in the log4j2.properties file.

3. A logger level set on the Log Settings UI page and stored in the repository.

All loggers that are not explicitly set to another level has the root level. Jaspersoft does not recommend changing the root level because any other level than ERROR generates huge log files and affect performance. Setting a log level in the log4j2.properties configuration file requires restarting the server. Setting a log level on the Log Settings UI page takes effect immediately and override any level set in the log4j2.properties file.

These settings are described in the following sections.

Editing the Log Configuration File

The log configuration file can be used to define new loggers, logging levels, log output, and log format, but you must restart the server for your changes to take effect.

> 📝 If you defined a custom logger in JasperReports Server 7.2 or earlier versions, you must update its definition in the log4j2.properties file using the new syntax below.

The configuration file for logging depends on whether you want to define loggers for import or export operations or all other server functionality.

Functionality to Log	File Location
Import or Export	<js-install>/buildomatic/conf_source/iePro/log4j2.properties
JasperReports Server	.../WEB-INF/log4j2.properties

Loggers are based on the Java classname of the functionality that you want to log. To find an existing logger or create a new one, you must know the corresponding classname. A logger in the configuration file is defined by three related definitions with the following syntax:

```
logger.<logger-name>.name=<Java-classname>
logger.<logger-name>.level=<log-level>
logger.<logger-name>.appenderRef.<output-type>.ref=<output-name>
```

where:

- <logger-name> is the name of the logger and usually based on the Java classname. The name cannot contain dots (.), so the convention is to use the entire classname with dots replaced by underscores (_).

- <Java-classname> is the name of the Java class that you want to monitor.

- <log-level> is error, warn, info, or debug.

- <output-type> is a Log4j 2 output type, either stdout or rolling. The respective <output-name> is either stdout or the name of a file output defined in the log4j2.properties file. There can be multiple output types, as shown in the following example:

```
logger.net_sf_jasperreports_engine_query_
JRJdbcQueryExecuter.name=net.sf.jasperreports.engine.query.JRJdbcQueryExecuter
logger.net_sf_jasperreports_engine_query_JRJdbcQueryExecuter.level=debug
logger.net_sf_jasperreports_engine_query_JRJdbcQueryExecuter.appenderRef.stdout.ref = stdout
logger.net_sf_jasperreports_engine_query_JRJdbcQueryExecuter.appenderRef.rolling.ref =
fileout
```

Some common loggers are given as comments in the configuration file, simply remove the comment character (#) from each line to add the logger. Otherwise, add all three properties to create a new logger.

Restart the server for your changes to take effect.

If you have made modifications in the Log Settings UI, those settings are persistent in the repository and take precedence over the configuration files. However, changes in the UI are not written to the configuration files. Each setting is independent. For more information, see Log Levels.

Configuring the Log Format

The available parameters for JasperReports Server logs include:

- **USER_ID** - The username with which JRS is logged in.

- **SESSION_ID** - Session ID of the logged in User.

- **RESOURCE_URI** - URI of the Resource opened.

- **REQUEST_TYPE** - The HTTP request type, for example: GET, POST, DELETE or PUT.

- **REQUEST_STATUS** - The HTTP request status code, such as 200 for OK.

- **TIME_TAKEN** - Time taken by a request in milliseconds.

JasperReports Server logs follow the following format, for example, for Log4j:

<Log4j pattern> <Request type> <Session Id> <User Id> <Resource URI> <Request Status> <TimeTaken> - <Actual text log message>

Here is an example log from jasperserver.log when Data source fails:

2023-07-14T03:22:13,469 ERROR SecureExceptionHandlerImpl,http-nio-8080-exec-6:135 -

{REQUEST_STATUS=200, REQUEST_TYPE=POST, SESSION_ ID=7F1DCDC110417960C651152A80E1877E, TIME_TAKEN=13, USER_ID=superuser}

- FATAL: no PostgreSQL user name specified in startup packet org.postgresql.util.PSQLException: FATAL: no PostgreSQL user name specified in startup packet

To configure the log format, use the log4j.properties file located in the WEB-INF folder. You can log all available parameters, or log a specific parameter.

Notes:

- Any delimeter can be used between the parameters.

- If any parameter is not applicable for given thread request, then that parameter will not be logged.

- By default, the log format used with version JasperReports Server 8.2 or earlier is still supported, but you can configure/add new parameters as per your requirements.

To log **all** parameters of Log4j's ThreadContext, use **%X** in layout pattern.

For example, use **%X** in log4j2.properties:

> appender.fileout.layout.pattern=%d{ISO8601}%5p %c{1},%t:%L - %X - %m%ex%n

To log **a specific parameter**, use:

> **%X{parameter_name}**

For example:

> **%X{USER_ID}**

To log **more than 1 parameter**, use:

> **%X{parameter_name1}, %X{parameter_name2}, %X{parameter_name3}**

For example:

> appender.fileout.layout.pattern=%d {ISO8601} %5p %c {1},%t:%L - **%X{USER_ID}**, **%X{SESSION_ID}**- %m%ex%n

Setting Log Levels in the UI

JasperReports Server provides a simple way to set log levels through the UI, and these settings take effect immediately on the server. Log settings in the UI are also stored persistently in the repository. This is useful for debugging any issues by setting the log level temporarily to avoid too much log output. However, you must remember to set it back or remove the setting when done.

To set the current logging levels

Procedure

1. Log in as a system administrator (superuser by default).

2. Select Manage > Server Settings and choose Log Settings in the left panel.
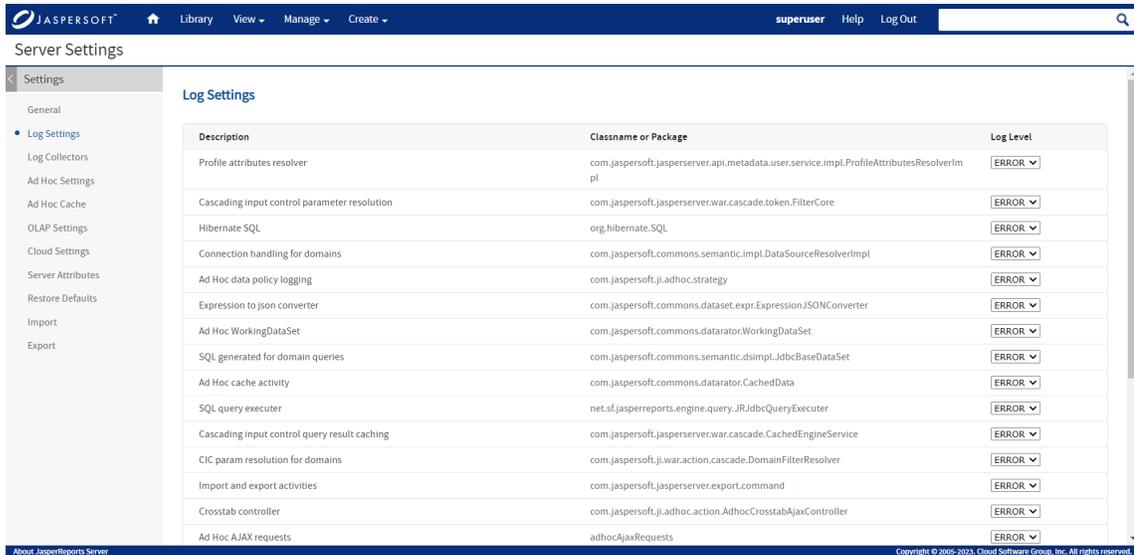
Figure 110: System Log Settings

3. In the list of loggers, use the dropdown selectors to change the log level for a given logger. Any change to a logging level on this page takes effect immediately, without restarting JasperReports Server.

4. If you want to set a logger that is not listed on the page, scroll to the bottom of the page and enter its Java classname in the text field, for example: com.jaspersoft.ji.adhoc.action.AdhocCrosstabAjaxController

5. Use the associated dropdown to set its logging level. When set, the new logger takes effect immediately.

    After creating a logger in this manner, the logger appears on the page and you can change its level again when necessary. The new logger is also stored in the repository and is persistent when the server is restarted, but it will no longer appear in the Log Settings Page. You can however enter its classname again with the desired log level.

    When viewing and setting loggers on the Log Settings page, keep the following in mind:

    • The list of loggers on this page is a pre-determined set that may be useful for debugging. This list of loggers is not related to those defined or set in the log4j2.properties file.

    • Setting a logger or log level on this page does not write or change any configuration in the log4j2.properties file.

- In the list of loggers, the logging levels reflect the current run-time level. If a logger level is set in the configuration file and never modified through this page, its level appears here. Otherwise, these loggers have the inherited root level of ERROR.

- As explained in Log Levels, a log level may be defined differently in the configuration file and on this page, in which case the level on this page takes precedence.

- Once you set a level or define a logger on the Log Settings page, its value is stored in the repository and becomes persistent when the server is restarted. Because the level stored in the repository has precedence, it will be the log level in effect after the server restarts.

- If you want to remove a logger or log level that was set on the Log Settings page and stored in the repository, select Restore Defaults in the left panel of the Server Settings UI. Click the delete icon beside the logger you want to remove.

The following table describes the loggers presented on the Log Settings page. To change which loggers appear permanently on the page, see Adding a Logger to the Log Settings Page.

Logger Name	Identifier in Log	Description
SQL query executer	JRJdbcQueryExecuter	Logs SQL text and parameter values for queries that are run by the SQL query executer.
Input control value queries	valueQueryLog	Logs SQL text and parameter values for queries associated with input controls.
Cascading input control parameter resolution	FilterCore	Logs activity associated with cascading input controls. Query-driven input controls can cascade when a query has a parameter whose value comes from another input control. When the parameter value is changed, the query is automatically rerun, possibly

Logger Name	Identifier in Log	Description
		changing the list of values for its input control.
Cascading input control query result caching	TokenControlLogic	Logs use of the cache for results of cascading input control queries.
Profile attributes resolver	ProfileAttributesResolverImpl	Determines the values of attributes when they are referenced.
Hibernate SQL	SQL	Logs SQL run by the Hibernate layer to access the JasperReports Server repository database. This logger generates a large volume of logging that could affect performance.
Ad Hoc data policy logging	CommonDomain-DataStrategy SubFilterInputControl-Generator Others	Logs the various activities of the Ad Hoc data policy implementations, which use SQL queries or in-memory operations to get datasets for Ad Hoc views.
SQL generated for Domain queries	JdbcBaseDataSet	Logs SQL queries generated from queries using a Domain.
Connection handling for Domains	DataSourceResolverImpl	Logs use of JDBC connections used by Domains to run SQL queries.
Expression to JSON converter	ExpressionJSON Converter	Logs information about the conversion between DomEL and JSON, which is used by Ad Hoc filters.
Domain-based	SemanticLayerSecurityResolver	Logs activity related to Domain

Logger Name	Identifier in Log	Description
security tests	Impl	column- and row-level security.
Cascading input control resolution for Domains	DomainFilterResolver	Logs the same activity as the FilterCore logger (Cascading input control parameter resolution) above, but adds information specific to Domain queries.
Ad Hoc cache activity	CachedData	Logs information about the life cycle of datasets that are cached in memory when Ad Hoc views are accessed.
Timing for SQL queries run for reports	JsControlledJdbcQueryExecuter	Logs the time that it takes a query run by the SQL query executer to return data to a report.
Ad Hoc WorkingDataSet	WorkingDataSet	Logs activity for the WorkingDataSet, used by the Ad Hoc Editor to perform in-memory dataset transformations of query results.
General controller	AdhocAjaxController	Logs activity of the Ad Hoc Editor.
Crosstab controller	AdhocCrosstabAjaxController	Logs additional activity of the Ad Hoc Editor specific to crosstab reports.
Groovy code generation for memory datasets	GroovyGenerator	Logs Groovy classes generated from DomEL expressions used by the Ad Hoc Editor for filters and calculated fields.
Ad Hoc AJAX requests	adhocAjaxRequests	Logs information about AJAX requests made by Ad Hoc Editor and dashboard designer, including

Logger Name	Identifier in Log	Description
		report parameters and response times. Enable this setting when you want to understand the Ad Hoc Editor and dashboard designer or if you have encountered an error or slow response times.
Ad Hoc cache activity	com.jaspersoft.commons. datarator.CachedData	Tracks the life cycle of datasets managed by the Ad Hoc cache as they transition between states. This log output includes information from the Ad Hoc cache in a format that lends itself to troubleshooting. Use this setting to understand how query response times contribute to the performance and responsiveness of the Ad Hoc Editor. Because it does not log the queries themselves, use it with the SQL Query Executer log setting.

Adding a Logger to the Log Settings Page

If you know of a Log4j 2 logger that JasperReports Server uses, you can edit a configuration file to add it to the Log Settings page available to the superuser.

To edit the list of loggers on the Log Settings page

Procedure

1. Edit the file .../WEB-INF/bundles/logger_descriptions_pro.properties.

2. Add a new line and specify the logger's classname and a brief description of it.

   Entries should be in the form <Java-classname> = <description>.

   See the other properties in the file for guidance, for example:

   com.jaspersoft.ji.adhoc.action.AdhocCrosstabAjaxController = Crosstab controller

3. Restart the server for your changes to take effect.

The logger_descriptions_pro.properties file does not set any logger levels, it only determines which loggers are listed in the Log Settings UI. Loggers are shown in the UI with their currently active level, as described in Log Levels.

> The logger_descriptions_pro.properties file contains descriptions for the English locale. You can specify text for other locales by editing the logger description property files for those locales. For example, to add the description in French, add the same Java classname to the logger_ descriptions_pro_fr.properties file. For more information on supporting other languages, refer to Localization.

Using the Monitoring Tool

JasperReports Server Professional Edition includes a monitoring tool that you can configure and launch to allow you to view diagnostic JMX and hardware details in a file generated by the tool.

Prerequisites

- Ensure that JMX is enabled on the JRS server.

- Java version 1.8 or later.

Configuration

Procedure

1. Configure the output file path with the environment variable TOOL_OUTPUT_FILE_ PATH, or with the system property as *tool.output.file.path*

    Otherwise, the output file path is configured, and then default output file name is *jrs-monitoring-report.txt*.

    Example of Output file path configuration

    a. Configuring the environment variable:
       export TOOL_OUTPUT_FILE_PATH=/tmp/myreport.txt

    b. While executing the tool jar, provide the system property option:
       -Dtool.output.file.path=/tmp/myreport.txt

2. The JMX URL can be configured with the environment variable TOOL_JMX_URL, or with the system property as *tool.jmx.url*

3. The JMX username can be configured with the environment variable TOOL_JMX_ USER_NAME, or with the system property as *tool.jmx.username*

4. The JMX user password can be configured with the environment variable TOOL_JMX_ USER_PASSWORD, or with the system property as tool.jmx.password

Launching the executable tool jar

Configuration

To launch the executable tool jar **using environment variable configuration**:

Once all the configuration are done by setting appropriate environment variables, launch the tool using:

> *java -jar monitoring-tool.jar*

To launch **using System properties configuration java**, run:

> *Dtool.jmx.url=service:jmx:rmi://localhost/jndi/rmi://localhost:10990/jasperserver -Dtool.jmx.username=superuser -Dtool.jmx.password=superuser -jar monitoring-tool.jar*

Using Log Collectors

Logs are useful for finding problems when the server is not behaving as expected. However, log files can be huge and difficult to search for the log messages associated with your issue.

Log collectors help you diagnose server errors by recording logs from a specific user or report for a specific duration, and make this excerpt of the log files available through the server's UI for administrators.

The output of a log collector includes excerpts from the log files and an export catalog of the resources being logged, including useful information such as the dataset being used for a report or Ad Hoc view. The export catalog and its dataset can be used to replay the report or view on another server.

Because the output may contain data or other sensitive information, it is always encrypted for security. After downloading this output, you can move the file to a test server or send

to Jaspersoft support. The first time you share a diagnostic log between two servers, you must also export and transfer the encryption key so the files can be decrypted.

To create and start a log collector

Procedure

1. Log in as system administrator (superuser by default).

2. Set your logging levels and your loggers for the classes you wish to capture, as described in Configuring System Logs.

3. Select Manage > Server Settings and choose Log Collectors in the left panel. The page displays any current log collectors, either running or stopped and ready to be downloaded.
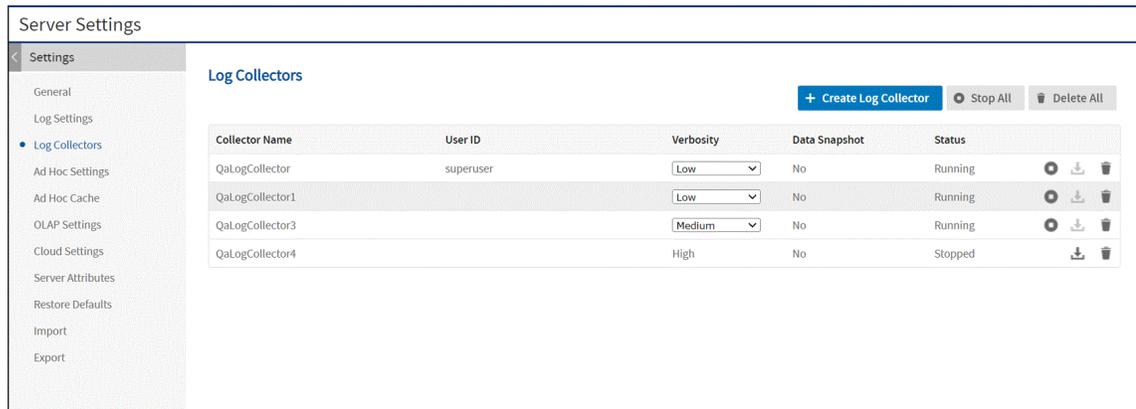
Figure 111: Viewing Log Collectors

4. Click Create Log Collector.

Server Settings

Figure 112: Creating a New Log Collector

5. On the New Log Collector page, enter a name for the collector and the following optional information:

- A user ID, in the format username|organizationID for commercial editions. When a user ID is specified, only logs related to that user are collected. When empty, logs for all user are included.

- The resource URI of a JasperReport or an Ad Hoc view. You can use the Browse button to select the report or Ad Hoc view in the repository. When specified, only the logs for that report or view will be collected. When empty, all logs are included.

- If you specified a report or view, and you have data snapshots enabled, you can select the check box to include the snapshot data from the report. The snapshot data is a separate file that can be imported to another server for verification.

- Set a verbosity level of low, medium or high. Low is the default.

6. Click Save to save your log collector and begin collecting the specified logs.

7. Perform the actions on the server for which you want to collect logs. If you specified a user ID, the user should log in or an administrator can log in as the user from the Manage > Users page. If you specified a report or view, run it and interact with it to collect logs about it.

To stop a log collector and download its output

Procedure

1. Log in as system administrator (superuser by default).

2. Select Manage > Server Settings and choose Log Collectors in the left panel.

3. On the Log Collectors page, locate your log collector and click the stop icon ⏹. The status for the collector shows "Stopping..."

4. After the log collector has stopped, the logs are ready to download in a compressed zip file. Click the download icon ⬇ to save the collected logs on your computer.

5. After downloading the logs, the log collector can be kept in case you want to download the files again. When you no longer need the log collector, click the delete icon ✖ and confirm. If you want to run a collector again, you must delete it first and create it again.

The zip archive contains two encpted files: a log file and an export catalog of the resource being logged, including any associated data such as a dataset or snapshot for a report or Ad Hoc view. To view the log file or import the catalog on another server, you need to export the corresponding encryption key.

To view the contents of a log collector output

Procedure

1. On the server that generated the log file, log into the host system as the user who installed JasperReports Server, and run the following commands:

```
cd <js-install>/buildomatic
js-export.sh --keyalias diagnosticDataEncSecret --destkeystore myks
--deststorepass mykspw --destkeypass logkeypw
```

You only need to export the key once, and then import it once to each of the servers with which you share the logs.

2. Securely transfer your log output and the keystore file created by the command above to the destination server.

3. On the destination server, log into the host system as the user who installed JasperReports Server, and run the following commands:

```
cd <js-install>/buildomatic
js-import.sh --input-key --keystore <path>/myks --storepass mykspw
--destkeypass logkeypw --keyalias diagnosticDataEncSecret
```

Now the destination server shares the same encryption key for decrypting the log and importing the catalog.

4. While still logged in as the system user, run the following commands:

```
js-import.sh --input-zip <logCollectorCatalog>
js-ant decrypt-diagnostics-data -DinFile=diagnostic.log.jsEncrypted
-Dkeyalias=diagnosticDataEncSecret -Dkeypass=logkeypw
```

It is also possible specify a hexadecimal key for the decryption. This can be used if you have previously replaced the source server's diagnosticDataEncSecret key with the same custom key.

```
js-ant decrypt-diagnostics-data -DinFile=diagnostic.log.jsEncrypted
-Dsecret-key="0x81 0x8d 0x3e 0x9b 0x1b 0xe0 0x9b 0x57 0x5a 0x93 0x4f 0xe6 0x43 0x90 0xb5
0x5e"
```

The js-ant output displays the content of the encrypted log file.

For more information about buildomatic and js-ant, see the JasperReports Server Installation Guide.

For more information about the diagnostic encryption key and how to export and import keys, see the JasperReports Server Security Guide.

To configure the properties of the log collector

Procedure

1. Open the file .../WEB-INF/applicationContext-diagnosticCollectors-pro.xml for editing and locate the diagnosticCollectorService bean.

2. If you want to collect more than 50MB of log data per file, modify the

appenderMaxFileSize property.

3. If you want your log files to be compressed inside the downloaded archive, specify a filename pattern with the .gz extension, for example:
    <property name="logFilePattern" value="diagnostic.%i.log.gz"/>

Auditing and Monitoring Events

Whereas logs capture messages from the internal code of JasperReports Server, the auditing and monitoring systems capture user events. This gives an operational picture of how users interact with the server and what resources they use. Auditing and monitoring can help you see who is using the server, and what resources are the most in demand. This can help you locate bottlenecks and optimize your reports.

In broad terms, an audit event is any atomic operation that can be recorded by the audit system. Event properties and attributes are features of the event; they can be defined internally or in custom code. Auditing and monitoring rely on the same record of events, so the audit events are also available in monitoring data sources, Domains, and reports.

The following table lists the defined audit events and the information collected about them. For every recorded event, JasperReports Server logs the time it occurred and the user who initiated it. See the configuration file applicationContext-audit.xml for complete specification of the events.

Event	Information Collected
Log in or log out	Time and user ID (recorded for every event)
Log in as	User logged in as
Run report or run a subreport within any other report	- Report referenced - Data source referenced - Report parameters and values - Report queries (such as SQL, Domain, HQL, generated SQL) - Execution start, end

Event	Information Collected
	• Query execution time (in milliseconds) • Report rendering time (in milliseconds) • Caching parameters • Errors that occurred
Report schedule created, deleted, or updated	• Report referenced • Scheduling parameters and values
Scheduled report run	• Report output • Report delivery parameters (such as email) • Same parameters as when report is run
Creating report in Ad Hoc Editor	• Field added as a column • Field added as a group
Resource accessed for any reason (such as view, used in report, etc.)	• Resource referenced • Resource type
Resource added or updated	• Resource referenced • Resource type
Resource or folder deleted	Resource or folder referenced
Resource or folder moved	Resource or folder referenced **Note**: When a resource or folder is moved, the logged URI is no longer valid. Also, when resources are moved, access events are updated in the queue, and the Home Page list of frequently accessed resources will be updated with new URIs after some delay.
Permissions added, updated, or deleted	• Resource or folder referenced • Previous permissions (before update)

Event	Information Collected
User added, updated, or deleted, also user password change	• User ID • User name • Email • Enabled flag • External flag • User attributes
Role added, updated, or deleted	• Role ID • Role name • Role organization
Organization added, updated, or deleted	• Organization ID • Organization description

Configuring Auditing and Monitoring

The auditing and monitoring features both rely on the same subsystem in JasperReports Server. The settings are located in the .../WEB-INF/applicationContext-audit.xml file, but their values are set in the .../WEB-INF/js.config.properties file. The beans and properties of these files are used to enable auditing and monitoring, set the archive options, and select the events to be logged.

> You must enable CLOB fields on your JasperReports Server before enabling audits if you want to use audit Domains for your data. See Enabling CLOB Fields for instructions on enabling CLOB fields.

Enabling Auditing and Monitoring

Because auditing and monitoring share the same subsystem, there is a master configuration setting to turn on the subsystem, and individual settings for auditing and monitoring. This allows you to turn on each feature independently, or turn off the whole subsystem to avoid any overhead from an unused feature.

By default, the auditing and monitoring subsystem is off.

Enabling Auditing and Monitoring Options	
Configuration File	
.../WEB-INF/js.config.properties	
Property	Description
feature.audit_monitoring.enabled	Set this property to true to enable the auditing and monitoring subsystem. You must also turn on audit and/or monitoring records independently with the following two properties. If you do not need auditing or monitoring, set this property to false to avoid any performance overhead.
audit.records.enabled	Once feature.audit_monitoring.enabled=true, set this property to true to enable auditing. This setting controls the logging of events in the tables used by the audit reports.
monitoring.records.enabled	Once feature.audit_monitoring.enabled=true, set this property to true to enable monitoring. This setting controls the logging of events in the tables used by the monitoring Domain.

Auditing Archive Options

Archiving automatically moves audit data to separate database tables, called the audit archive, after a certain time. For example if you create reports for weekly server usage based on the audit Domain, you'll never see audit data older than a month by default. Because audit data can be quite large, the old data can increase the time needed to write new data and query data for your report. If you want to create reports on audit archive tables, there is a separate Domain to access the archived audit data. You can also configure the audit feature to delete old archive data if you no longer need it.

You should set the archiving interval to a level that balances your need to record and access audit data with your server's performance on large audit tables. Use the following configuration settings to change the archiving behavior.

Archiving and deleting (or purging) of audit data happens once per day, by default in the early morning hours. You can also set the time and frequency of these operations.

Auditing Archive Options

Configuration File

…/WEB-INF/applicationContext-audit.xml

Property	Bean	Description
maxAudit EventAge ToArchive	auditService	The number of days to keep audit data in the active audit tables. The default is 30. Older data is moved to the archive.
maxAudit EventAge	auditService	The total number of days to keep any audit data. Older data is deleted from either the active or the archive tables. If this value is non-zero but less than the maxAuditEventAgeToArchive, the audit data will be deleted before it is ever archived (this may be useful if you don't want to use the archive). The default is 0 (zero), meaning that old data is never deleted.
cronExpression	auditEvent ArchiverTrigger	Defines the frequency of the archiving job as a cron expression. The default 0 0 5* * ? is every day at 5 A.M.
cronExpression	auditEvent PurgerTrigger	Defines the frequency of the audit delete (purge) job as a cron expression. The default 0 0 3 * * ? is every day at 3 A.M.

The cronExpression properties use a Quartz scheduler cron expression that specifies the repeating trigger as seconds, minutes, hour, day of month, month, day of week, respectively.

Disabling Events and Properties

By default, all events and properties are logged. To enable or disable logging of a given event or property, use the applicationContext-audit.xml configuration file. Events that are logged are used by both the audit and monitoring subsystems, therefore disabling an event makes it unavailable for both auditing and monitoring.

In the file, event types and their properties are listed under <util:map id="enabledEventsMapping">. The map has three parts:

- WEB_SERVICES – Event types related to accessing JasperReports Server through a web service.

- GUI – Event types for access through the user interface.

- INTERNAL – Event types used by the server itself, such as when running a scheduled report.

To disable an event, comment it out. For example:

```
<!-- <entry key="createFolder" value="folderName,folderLabel,folderDescription" /> -->
```

To disable a property, use any of these measures:

- Delete the property. For example, remove folderDescription, resulting in:
  <entry key="createFolder" value="folderName,folderLabel,exception" />

- Disable it with the | syntax. For example:
  <entry key="createFolder"
  value="folderName,folderLabel,|folderDescription,exception" />

- Use the "all except" *| syntax to specify only the disabled property. All others are recorded. For example:
  <entry key="createFolder" value="*|folderDescription" />

Using the Audit Data

JasperReports Server makes the audit data available to system admins through Domains and several prepared views and reports. These are located in the /Public/Audit folder of the repository.

Figure 113: Audit Reports in the Repository

There are two Domains and two sets of reports created for accessing audit data:

- Audit Domain and Audit Reports - Use these to view the current audit data; they run against the active audit database.

- Audit Archive Domain and Archived Audit Reports - Use these to run reports on archived data; they run against the archive database.

The contents of both Domains and reports are identical. They differ only in the database tables accessed in each case.

To create an Ad Hoc View based on the audit Domains:

1. Select Create > Ad Hoc View

2. Select the Domains tab in the Data Chooser, and expand the folders to select one of the audit Domains.
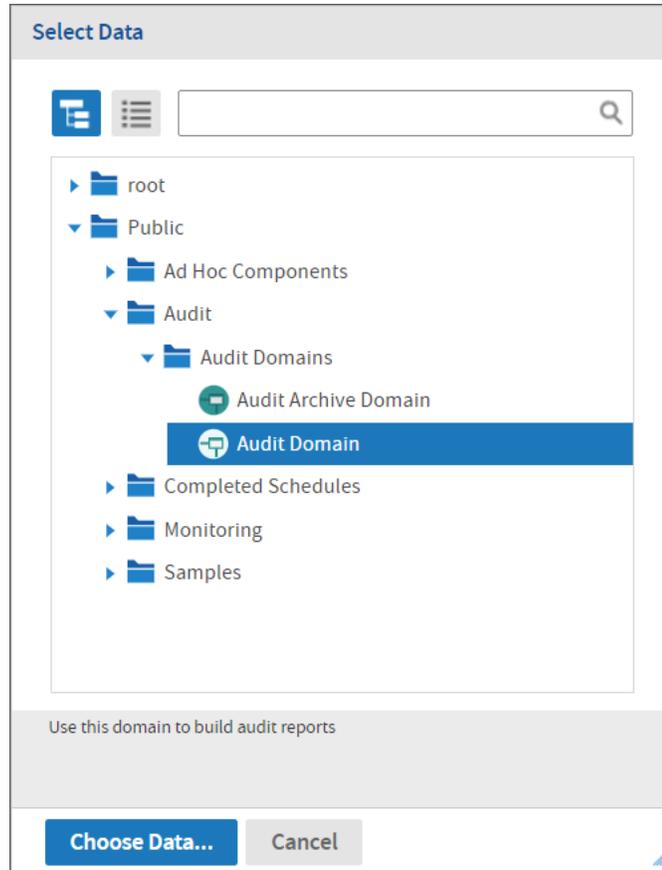
Figure 114: Selecting an Audit Domain to Create an Ad Hoc View

For instructions on using Domains in reports, see the Ad Hoc chapter in JasperReports Server User Guide. For documentation of Domains in general, see the Domains chapter in the same manual.

The following sections explain the contents of the Domains and the reports provided.

Audit Domain Items

The audit Domains expose the fields of the audit logging tables stored in the server's internal database. As with all Domains, the database tables are joined, and the fields are presented as items that can be used in Ad Hoc views.

The following tables describe the items in both audit Domains (Audit Domain and Audit Archive Domain). These items correspond to the information recorded for each event. When creating a view based on either Domain, choose the items that correspond to the type of event you want to report on.

These domain items are used in both general events and repository events:

Domain Item	Description
Date	Date event occurred.
Prop Long Value	clob value of event property, such as query.
Prop Type	Type of event property, such as destination folder, as per event map in configuration file.
Prop Value	string value of event property, such as folder name.
Time	The time event occurred.
Event Type	Type of event, such as save resource, as per the event map in the configuration file.
Request Type	Repository request type of event.
Resource Type	Repository type of resource accessed in an event.
Resource URI	URI of repository resource.

Domain items in the following table are recorded for user events:

Domain Item	Description
E-mail	E-mail address of event user (user at time of event).
Enabled	Whether the event user is currently a user.
External	Whether the event user was an external user.
Full Name	Full name of event user.
Password Changed	Whether the event was a change of password

Domain Item	Description
Organization	Organization of event user.
User Name	UserID of event user.

Domain items in the following table are recorded for role events:

Domain Item	Description
External	Whether a role was defined in an external system.
Role Name	Name of the role in the event.
Organization	Organization of the role in the event.

Domain items in the following table are recorded when a report is generated:

Domain Item	Description
Date	The date the report is generated.
Time	The time at which the report is generated.
Resource URI	URI of repository resource accessed for report.
Resource Type	Repository type of resource accessed for report.
Datasource URI	URI of data source accessed for report.
Query Execution Time	Time to execute the query in the database.
Report Rendering Time	Time to prepare query results for display.
Report Execution Time	Total time to execute the report (query execution + report rendering + overhead). Overhead includes tasks such as loading repository resources (report unit, data source, etc.) and obtaining a DB connection from the data source.

Domain Item	Description
Query	Specification of the report query.
User Name	UserID of event user.
Organization	Organization of event user.
Crosstab Group Field	Field name used in crosstab

Audit Reports and Ad Hoc Views

> Auditing is disabled by default, so the audit reports and their views are initially blank. To view audit reports, enable auditing as described in Configuring Auditing and Monitoring, then wait for user activity to generate events.

A number of Ad Hoc views and reports based on the audit Domains are provided in the /Public/Audit/Audit Reports folder. The same views and reports are also provided in the Archived Audit Reports subfolder. These reports are identical, except they use the Audit Archive Domain and run against the archived audit data. These reports are visible only to administrators.

The reports are designed to cover common audit needs and can be used as-is. When auditing is enabled, audit events are recorded, the reports contain an up-to-the-minute record of events on your server. You can run the reports or schedule them as needed. When running the audit reports, you can also change the input controls to filter different sets of events or date ranges. By default, the input control for event types contains no selection, which indicates that all are to be displayed. You can change this selection to display only the events you wish to observe.

The Ad Hoc views used to create each report are also included. You can open them in the Ad Hoc Editor to explore the audit data in real-time. You can also modify these views in the Ad Hoc Editor to generate new reports to suit your auditing requirements.

The following views and reports are provided:

- Audit Report - Generic example of an audit report showing commonly audited events.

- Performance Crosstab Report - A crosstab that shows average performance of reports.

- Performance Report - Generates a list of reports, sorted by run-time to identify slow reports.

- Repository Resources Report - Shows repository resources and their associated events.

- Resource Execution Report - Generates a list of executed reports.

- User Activity Report - Generates a list of reports run by a specified user.

Using the Monitoring Data

JasperReports Server makes the monitoring data available to administrators through a Domain and several prepared views and reports. These are in the /Public/Monitoring folder of the repository.

Figure 115: Monitoring Reports in the Repository

To create an Ad Hoc View based on the audit Domains, select Create > Ad Hoc View, select the Domains tab in the Data Chooser, and select the monitoring Domain. For instructions on using Domains in reports, see the Ad Hoc chapter in JasperReports Server User Guide. For documentation of Domains in general, see the JasperReports Server Data Management Using Domains.

The following sections explain the contents of the Domains and the reports provided.

Monitoring Domain Items

The monitoring Domain exposes the fields of the monitoring tables stored in the server's internal database. As with all Domains, the database tables are joined, and the fields are presented as items that can be used in Ad Hoc views.

In this release of JasperReports Server, the monitoring fields are limited to those that record report execution events:

Domain Item	Description
Day	Day of the month the report finished running.
Editing Action	The Ad Hoc editing step the user just performed (null for report execution): • insertDimensionInAxisWithChild - A dimension was added. • addMeasure - A measure was added. • setProperty - A property was set. • moveDimension - A dimension was moved.
Event Context	The context that triggered the report execution. Possible values are: • ui - The report ran interactively from the user interface. • web services - The report was run via web services. • internal - The report ran from an internal process, usually the scheduler.
Event Type	The type of report that was executed. Possible values are: • report execution - A report that ran from the repository. • ad hoc editing - A report that ran from an Ad Hoc view being edited.
Hour	The hour the report finished running.
id	ID number of the monitoring event.

Domain Item	Description
Minute	The minute the report finished running.
Month	The month report finished running.
Query Execution Time	The time spent executing the SQL query in the database.
Report Rendering Time	The time spent rendering the report after receiving the query results (dataset).
Report URI	Repository path of the report that was run.
Time Stamp	Full time and date when the report was finished running, including milliseconds.
Total Report Execution Time	The total time spent running the report. Typically this is a little more than the sum of the query execution and report rendering times. This is due to overhead such as loading repository resources (report unit, data source, etc.) and obtaining a DB connection from the data source.
User Name	The user who ran the report.
User Organization	The organization of the user who ran the report.
Year	The year the report finished running.

Monitoring Reports and Ad Hoc Views

The monitoring reports and their views are blank by default. This is because the audit subsystem that monitoring depends on is disabled by default and no audit data exists. To view these reports, first enable auditing as described in the section Configuring Auditing and Monitoring, then wait for user activity to generate events.

A number of Ad Hoc views and reports based on the monitoring Domain are provided in the /Public/Monitoring/Monitoring Reports folder.

The reports are designed to cover common monitoring needs and can be used as-is. When monitoring is enabled, audit events are recorded, and the reports contain an up-to-the-

minute record of events on your server. You can run the reports or schedule them as needed.

The Ad Hoc views used to create each report are also included. You can open them in the Ad Hoc Editor to explore the monitoring data in real-time. You can also modify these views in the Ad Hoc Editor to generate new reports to suit your monitoring requirements.

The following views and reports are provided:

- Report Monitoring Resources Report - Gives a list of all reports and shows their average and high-low execution times.

- Report Monitoring Details Report - A crosstab that shows report execution times on one axis and many dimensions on the other axis such as:

    - A time hierarchy

    - User and organization

    - Event type
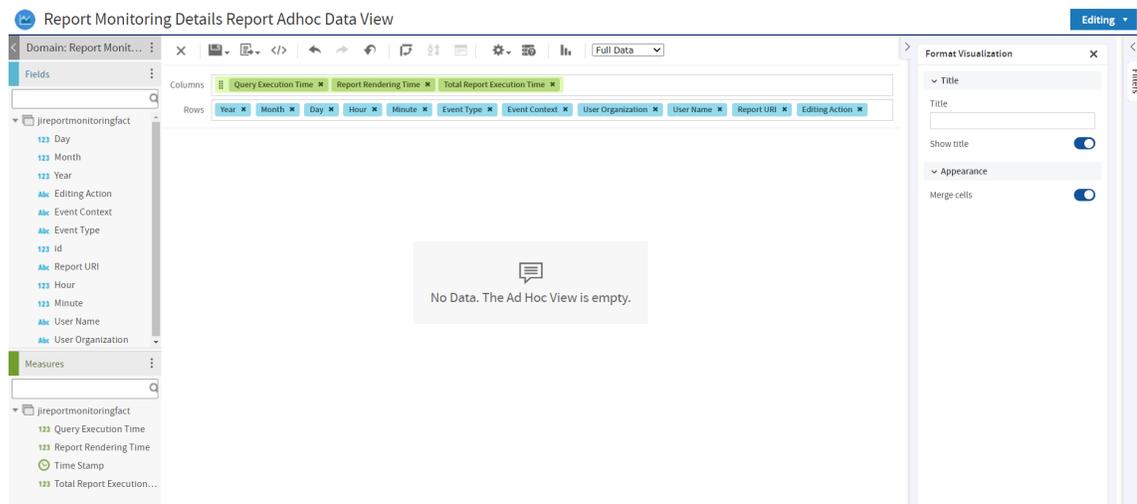
Figure 116: Monitoring Ad Hoc View with Multidimensional Analysis

Importing and Exporting Event Data

Audit and monitoring data can be imported and exported with the utilities described in the chapter Import and Export.

- To export audit data, use the --include-audit-events option in the export command.

- To export monitoring data use the --include-monitoring-events option in the export command.

- To import audit data, import the catalog containing audit data with the --include-audit-events option.

- To import monitoring data, import the catalog containing audit data with the --include-monitoring-events option.

> Data in temp folders is not exported.

Real-Time Diagnostics

JasperReports Server's full instrumentation provides real-time diagnostic information such as memory use, concurrently running reports, cache statistics, and much more. Whereas auditing and monitoring provide a historical record of events on the server, diagnostics display the current status of the server and its Java Virtual Machine (JVM) environment.

For example, you can use the diagnostic information to adjust your JVM settings. This helps to optimize memory based on the size of typical datasets in your Ad Hoc cache. As diagnostic data is available through a Topic, you can create Ad Hoc views. This Ad Hoc view helps you to analyze your server performance and identify performance bottlenecks before they happen.

The diagnostic feature relies on the Java Management Extensions (JMX) industry standard to gather run-time data and expose it to other management interfaces. JasperReports Server implements several JMX elements:

- New JMX-managed beans used to gather and hold diagnostic data such as the number of concurrent reports.

- Its own JMX agent, like a server that responds to JMX requests and exposes the information from the managed beans.

- Connections to existing JMX-managed beans in other components such as Ehcache to include their diagnostic data.

- Connections to other JMX agents such as JVM agent to gather system diagnostics.

For more information about JMX, see http://docs.oracle.com/javase/tutorial/jmx/.

There are two ways to use diagnostic data from JasperReports Server:

- If you have an external management console that can connect to a JMX agent, then you can use it to connect to the server's JMX agent. You will probably also want to connect to the JVM's JMX agent. You can then view the real-time diagnostic data from JasperReports Server integrated with the data from your other servers in your management console. Management consoles often offer further functionality like thresholds and alarms you can set on data from any JMX-managed bean.

- We provide a custom data source within the server that collects the same information from the managed beans in the server and in the JVM. You can then create JasperReports that access this data source and present this diagnostic data in a report that suits your needs. We also provide a report and an Ad Hoc Topic to demonstrate this.

Exposing Diagnostics Through Jaspersoft's JMX Agent

By default, the diagnostic feature is configured to allow access to the diagnostic data through a remote management console that supports JMX. This allows you to integrate JasperReports Server diagnostics with your overall IT management strategy. Alternatively, the Java Development Kit provides the jConsole tool that uses JMX and can be used to monitor your server remotely. All of these management consoles access the JMX agent provided with JasperReports Server, and access is secured by the same users, passwords, and roles used by the server's web interface.

Connecting to the JMX Agent

The JMX server component is now disabled by default, and only the Internal Diagnostic Report and Diagnostic DataSource are available. If you still require real-time diagnostics provided by the JMX Server, you can re-enable the server through the following steps:

Procedure

1. Edit **applicationContext-diagnostic.xml**.

2. Find string "BEGIN Embedded JasperServer JMX Server".

3. Uncomment beans up to the word END (beans jMXAuthenticator, jMXregistry,

jasperJMXServerConnector, jasperJMXServer).

4. Comment out the bean that goes after END. The updated XML snippet should look like this:

```xml
<!-- BEGIN Embedded JasperServer JMX Server -->
```

```xml
<bean id="jMXAuthenticator"
class="com.jaspersoft.jasperserver.api.logging.diagnostic.jmx.JmxSe
curityAuthenticator">
```

```xml
<property name="authenticationManager"
ref="authenticationManager"/>
```

```xml
<property name="allowedRoles"
ref="${bean.diagnosticAllowedRoles}"/>
```

```xml
</bean>
```

```xml
<bean id="jMXregistry"
class="org.springframework.remoting.rmi.RmiRegistryFactoryBean">
```

```xml
<property name="port" value="${diagnostic.port}"/>
```

```xml
</bean>
```

```xml
<bean id="jasperJMXServerConnector"
class="org.springframework.jmx.support.ConnectorServerFactoryBean"
lazy-init="false">
```

```
<property name="server" ref="jasperJMXServer"/>

<property name="objectName" value="connector:name=rmi"/>

<property name="serviceUrl"
value="service:jmx:rmi:///jndi/rmi://${diagnostic.rmiHost}:${diagno
stic.port}/${diagnostic.name}"/>

<property name="environmentMap">

<map>

<entry key="jmx.remote.authenticator" value-
ref="jMXAuthenticator"/>

</map>

</property>

</bean>


<bean id="jasperJMXServer"
class="org.springframework.jmx.support.MBeanServerFactoryBean"
depends-on="jMXregistry">

<property name="locateExistingServerIfPossible"
value="${diagnostic.usePlatformJMXServer}"/>

</bean>
```

```
<!-- END -->



<!--Comment this bean declaration out if Embedded JasperServer JMX
Server is enabled-->


<!--<bean id="jasperJMXServer"
class="org.springframework.jmx.support.MBeanServerFactoryBean" >


<property name="locateExistingServerIfPossible"
value="${diagnostic.usePlatformJMXServer}"/>


</bean>-->
```

5. Start the server.

To connect your management console to the JasperReports Server's JMX agent, initiate a new connection from your console and specify the following connection string:

```
service:jmx:rmi://localhost/jndi/rmi://<host>:<port>/<connectionName>
```

where:

- <host> is the computer where JasperReports Server is running

- <port> is the JMX port, by default 10990

- <connectionName> is the name of the JMX agent, by default jasperserver

Therefore, the default connection string is:

```
service:jmx:rmi://localhost/jndi/rmi://<host>:10992/jasperserver
```

> If you have a firewall implemented on the computer hosting JasperReports Server, you'll need to open the JMX port (10992 by default) before connecting.

When prompted to enter a username and password, give a user with the following roles:

Commercial editions:	ROLE_SUPERUSER, thus by default the superuser user
Community project:	ROLE_ADMINISTRATOR, thus by default the jasperadmin user

The following sections explain how to modify the default connection values for:

- The connection name

- The JMX port

- The required roles

Configuring the Port and Connection Name

The connection name is the name registered for the JMX agent with the remote method invocation (RMI) service. If you have more than one JasperReports Server instance in a single app server, you need to change the registered connection name for all but the first one.

There are times you may want to change the port from default (10992) to another port, for example, if another service is configured to use the default port. For example, change the port if you have JasperReports Server running in a second app server on the same computer.

To change the port or connection name, edit one of following files:

- In the WAR file distribution before installation: <js-install>/buildomatic/default_master.properties

- In the installed WAR file before starting the server: …/WEB-INF/js.diagnostic.properties

Find the following lines and edit the values to the port number and connection name you want:

```
diagnostic.jmx.port = 10992
diagnostic.jmx.name = jasperserver
```

Configuring Roles for JMX Connections

If you want to allow other users to establish the JMX connection from a remote management console, configure JasperReports Server to allow other roles:

- For commercial editions, edit the .../WEB-INF/applicationContext-diagnostic-pro.xml file and modify the following setting:

```
<util:list id="diagnosticAllowedRolesPro">
    <value>ROLE_SUPERUSER</value>
</util:list>
```

- For community projects, edit the .../WEB-INF/applicationContext-diagnostic.xml file and modify the following setting:

```
<util:list id="diagnosticAllowedRolesCe" value-type="java.lang.String">
    <value>ROLE_ADMINISTRATOR</value>
</util:list>
```

In both cases, you can change the existing role or add additional lines containing alternate <value>ROLE_name</value>.

Disabling Remote Connections to the JMX Agent

By default, remote connections to the JMX Agent are enabled and configured as described in Connecting to the JMX Agent.

To disable remote connections, edit the .../WEB-INF/applicationContext-diagnostic.xml file and make the following changes:

- Comment out the entire jMXAuthenticator bean.

- Comment out the entire jMXregistry bean.

- Comment out the entire jasperJMXServerConnector bean.

- Comment out or remove the property depends-on="jMXregistry" in the jasperJMXServer bean.

To comment out a section of this XML file, surround it with <!-- and --> tags.

Alternative Connection Through App Server JMX Service

Most app servers also have a JMX agent, and if configured properly, they can discover the JasperReports Server JMX agent and expose the diagnostic information. However, this connection has some significant security implications:

- When the app server discovers and connects to the JMX agent, it has full access to the diagnostic information.

- If the app server's JMX agent is configured for remote access, the remote manager also has access to the JasperReports Server JMX agent without needing to provide the JasperReports Server username and password. In other words, by trusting the app server to access the diagnostic information, you're also allowing anyone the app server trusts to connect.

- The connection between the two JMX agents is reciprocal. If remote access is still enabled on the JasperReports Server JMX agent, a remote manager who connects to it also sees the contents of the app server JMX agent. In other words, the app server JMX agent is also trusting anyone who we trust to connect to our JMX agent (with our username and password).

To connect automatically to the app server's JMX agent, assuming one is available, edit the default_master.properties file before you deploy the JasperReports Server web app, and add the following line:

```
diagnostic.jmx.usePlatformServer = true
```

If the app server is Apache Tomcat for example, a local JMX connection named Catalina appears to anyone accessing the JasperReports Server JMX agent.

Using the Diagnostic Data in Reports

In addition to remote JMX connections, JasperReports Server makes the diagnostic data available to administrators for internal reports. The following resources are visible to the system admin (superuser by default) in the repository:

- /Public/Diagnostic/Diagnostic Data Source – A custom data source that returns all of the diagnostic fields.

- /Public/Diagnostic/Diagnostic Report – A report that displays nearly every single diagnostic field.

- /Public/Ad Hoc Components/Topics/Diagnostic Topic – A JRXML resource that can be used as a Topic to create an Ad Hoc view using diagnostic fields.
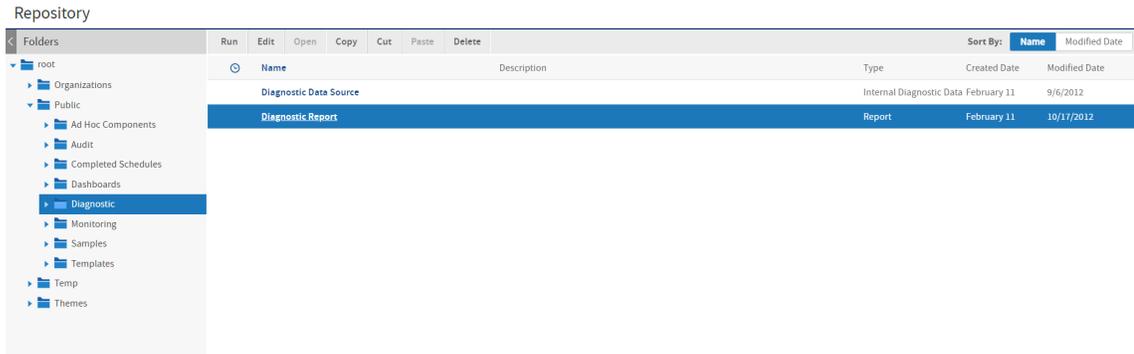
Figure 117: Diagnostic Report in the Repository

The diagnostic topic and report are based on the diagnostic data source. This is a custom data source that encapsulates the JMX information and makes it available as a data source for reports. In addition, there is an Internal Diagnostic Data Source type that can be used to create a new data source. If you accidentally delete the diagnostic data source, select Create > Data Source, set the type to Internal Diagnostic Data Source, and set the name and location to re-create it.

The following figure shows the beginning of the diagnostic report. This report displays nearly every field available through JMX and the diagnostic subsystem. This report serves as a reference for the diagnostic data. The report is also very useful when configuring your server, because it shows many server configuration settings as well as system information such as memory use. You can use this information to fine tune the performance of your JasperReports Server instance.

Figure 118: Contents of the Diagnostic Report

Excluding Diagnostic Attributes

The JasperReports Server diagnostic feature exposes a set of managed beans (MBeans), each with a number of JMX attributes. By default, all attributes of all MBeans are available. If you would like to limit what is exposed, you can exclude any of the attributes or entire beans through the following configuration file:

.../WEB-INF/applicationContext-diagnostic.xml

- To exclude an attribute, locate the MBean where it's defined and uncomment it from the excludedDiagnosticAttributes property. For example, if you want to hide sensitive information about your internal database, modify the diagnosticRepositoryDatabaseInfoCe MBean as follows:

```xml
<bean id="diagnosticRepositoryDatabaseInfoCe" class=
    "com.jaspersoft.jasperserver.api.logging.diagnostic.jmx.DiagnosticDynamicMBean">
  <property name="diagnosticServices">
    <set>
      <ref bean="repositoryDatabaseInfo"/>
    </set>
  </property>
  <property name="excludedDiagnosticAttributes">
    <set>
      <value>DatabaseProductName</value>
      <value>DatabaseProductVersion</value>
      <value>DriverName</value>
      <value>SQLKeywords</value>
      <value>URL</value>
      <value>UserName</value>
      <value>JDBCMajorVersion</value>
      <value>JDBCMinorVersion</value>
      <!--
      <value>MaxRowSize</value>
      <value>MaxStatementLength</value>
      <value>MaxConnections</value>
      <value>MaxCharLiteralLength</value>
      <value>MaxColumnsInTable</value>
      <value>MaxColumnsInSelect</value>
      <value>MaxColumnsInGroupBy</value>
      <value>MaxColumnNameLength</value>
      -->
    </set>
  </property>
</bean>
```

- To exclude an entire bean, comment it out or remove it from the list of beans in the diagnosticExportingMBeansMap. For example, instead of excluding selected attributes, you could remove the entire repository database MBean as follows:

```
<util:map id="diagnosticExportingMBeansMap" >
    <entry key="jasperserver:name=Users" value-ref="${bean.diagnosticUsers}"/>
    <entry key="jasperserver:name=Reports" value-ref="${bean.diagnosticReports}"/>
    <entry key="jasperserver:name=Scheduler" value-ref="${bean.diagnosticScheduler}"/>
    <entry key="jasperserver:name=Settings" value-ref="${bean.diagnosticSettings}"/>
    <entry key="jasperserver:name=Repository" value-ref="${bean.diagnosticRepository}"/>
    <entry key="jasperserver:name=About" value-ref="${bean.diagnosticJSAbout}"/>
    <entry key="jasperserver:name=HibernateStatics"
           value-ref="${bean.diagnosticHibernate}"/>
    <entry key="jasperserver:name=EhCache"
            value-ref="${bean.diagnosticEhCache}"/>
    <entry key="jasperserver:name=ExternalInfo"
            value-ref="${bean.diagnosticExternalInfo}"/>
    <!--
    <entry key="jasperserver:name=RepositoryDatabase"
            value-ref="${bean.diagnosticRepositoryDatabaseInfo}"/>
    -->
</util:map>
```

Disabling Real-Time Diagnostics

By default the JMX diagnostic subsystem is always enabled, but external access is password-protected and requires opening the diagnostic port in your firewall as described in Exposing Diagnostics Through Jaspersoft's JMX Agent. If you wish to disable all external access, see Disabling Remote Connections to the JMX Agent.

Internally, the diagnostics subsystem is passive and has no performance impact until it is accessed in a report through the diagnostic data source. However, if you wish to disable real-time diagnostics entirely, rename or remove the following files:

- …/WEB-INF/applicationContext-diagnostic.xml

- …/WEB-INF/applicationContext-diagnostic-pro.xml

In that case, the diagnostic data source, the sample report, and the sample Topic described in Using the Diagnostic Data in Reports will not function either. They can be deleted from the repository.

Server Monitoring

Administrative users have access to a set of Ad Hoc views with details of the completed scheduled reports via the server monitoring feature. Administrators can monitor the completed scheduled reports using these Ad Hoc views. Administrators can only monitor the data that they have access to using these Ad Hoc views. System admins (superusers) have access to all the data and organization admins (jasperadmin) have access to the data of their organization. This feature is available to users with access to the Audit database. Auditing and Monitoring options should be enabled in the .../WEBINF/js.config.properties file. For instructions on enabling these options, see Enabling Auditing and Monitoring.

Administrators also have access to Domains with the data that is used to create Ad Hoc views. You can change column headers, create calculated fields, and so on, using the data provided in domains.

The Ad Hoc views and domains are located in the /Public/Completed Schedules folder of the repository.

Monitoring Ad Hoc Views

The Ad Hoc views are provided in the /Public/Completed Schedules/Ad Hoc Views folder. The following Ad Hoc views are available:

- Successful Reports Ad Hoc View
- Failed Reports Ad Hoc View
- Output Formats Ad Hoc View
- Query Details Ad Hoc View
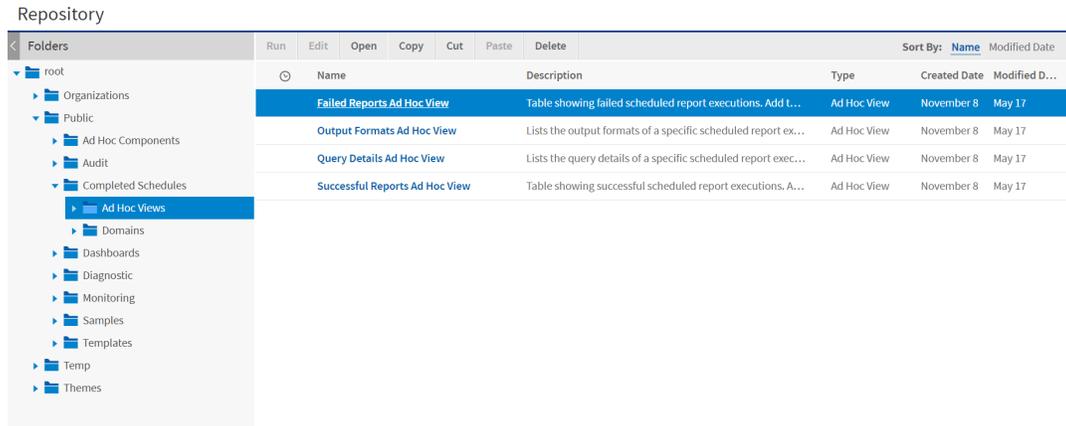
Figure 119: Ad Hoc Views in Completed Schedules

You can click the Ad Hoc view name to open it. You can add filters to the views. For Query Details Ad Hoc View and Output Formats Ad Hoc View, you need to use a specific Execution ID as an input control to view the data.
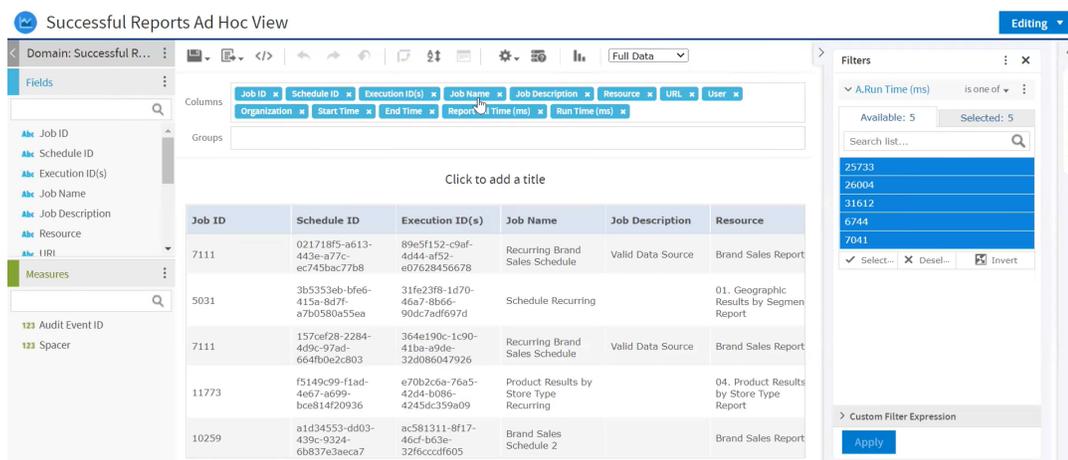
Figure 120: Successful Reports Ad Hoc View

Ad Hoc views use cached data for better performance, so the recommendation is to create a dashboard using these Ad Hoc views to view real-time data.

We have two levels of data for the completed scheduled reports:

- Level 1:
    - Successful Reports Ad Hoc View: Table showing successful scheduled report executions with data like Job ID, Schedule ID, Execution ID, User, Start Time,

End Time, Run Time, and more.

○ Failed Reports Ad Hoc View: Table showing failed scheduled report executions with data like Job ID, Schedule ID, Execution ID, User, Start Time, End Time, Run Time, Execution Error Category, Error Messages, and more.

You can add scheduled report executions to a dashboard to view the latest data every time the dashboard runs.

• Level 2:

○ Output Formats Ad Hoc View: Lists the output formats of a specific scheduled report execution.

○ Query Details Ad Hoc View: Lists the query details of a specific scheduled report execution.

You can add these views to the dashboard with the Successful Reports Ad Hoc View/Failed Reports Ad Hoc View and use Execution ID as input control for these views. Using the Execution ID as an input control to drill down from level 1 to level 2, you can view detailed data of the specific scheduled reports.

Viewing Real-Time Ad Hoc View Data

For performance reasons, Ad Hoc views use cached data. To see the real-time data do any of the following:

Clearing Ad Hoc Cache

To clear Ad Hoc cache

Procedure

1. Log in as an administrator (jasperadmin in the user's organization or any parent organization, or superuser).

2. Select Manage > Server Setting, and then click **Ad Hoc Cache** in the left panel.

3. Click **Clear All**.

Create Reports using Ad Hoc Views

To view the real-time Ad Hoc View data using reports

Procedure

1. Create reports using Ad Hoc views.

2. For the updated data, click the **Refresh** button on the report.

> By default, on refresh, the cache is deleted and new data is retrieved. (If you do not want to delete the cache on report refresh, the following property needs to be added to the js.config.properties file: flushAdHocCacheOnRefresh = false. By default, this property is not added and is true.)

> The report refresh works fine for Successful Reports Ad Hoc View and Failed Reports Ad Hoc View reports. However, it does not work for Query Details Ad Hoc View and Output Formats Ad Hoc View reports, because these views have input controls.

Adding Ad Hoc Views to Dashboards (Recommended)

To view the real-time Ad Hoc View data using dashboards

Procedure

1. Create a dashboard.

2. Add Successful Reports Ad Hoc View/Failed Reports Ad Hoc View to the dashboard.

3. Add Output Formats Ad Hoc View and Query Details Ad Hoc View to the dashboard.

4. For the Successful Reports Ad Hoc View/Failed Reports Ad Hoc View dashlet, in the Dashlet Settings, turn **Enable hyperlinks** on.

5. In the Dashlet Settings, click **Map Parameters** to open the **Parameter Mapping** dialog box. Map the **Execution ID** parameter of Successful Reports Ad Hoc View/Failed Reports Ad Hoc View dashlet with Output Formats Ad Hoc View and Query Details Ad Hoc View dashlets. For the Filter/Parameter Affected field, select **Schedule ID** for both the dashlets.
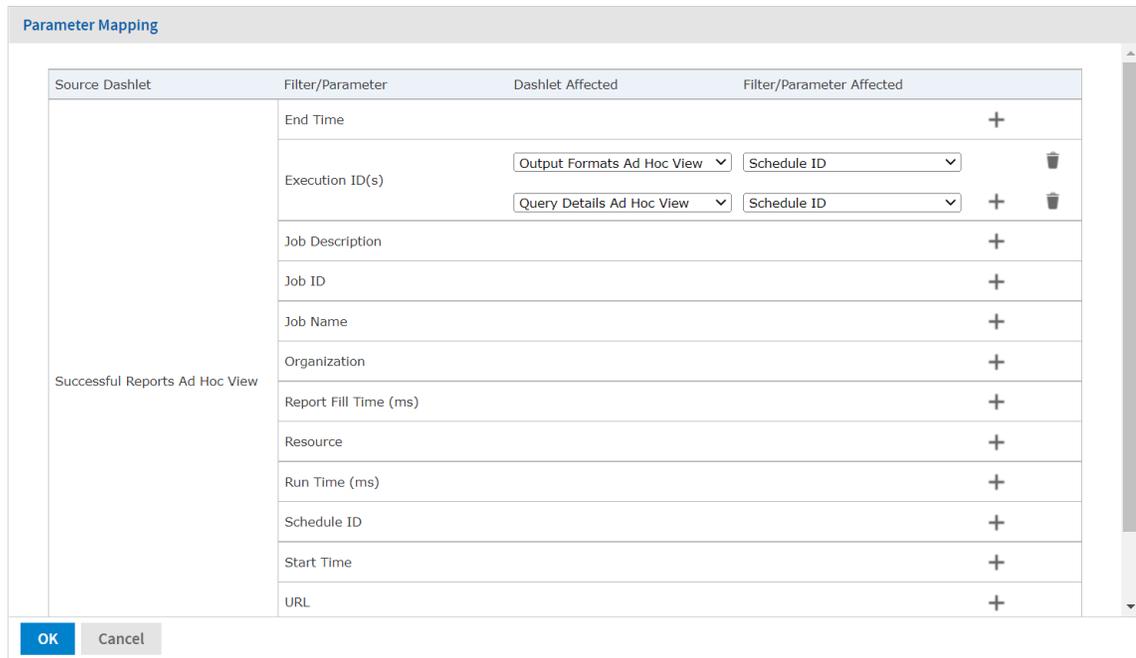
Figure 121: Parameter Mapping Dialog Box

For more information on parameter mapping, see the JasperReports Server User Guide.

6. For all dashlets, in Dashlet Settings, enable **Auto-refresh**.
   Successful Reports Ad Hoc View/Failed Reports Ad Hoc View Dashlet shows the updated data.

7. Click on the Execution ID of any scheduled report in the Successful Reports Ad Hoc View/Failed Reports Ad Hoc View dashlet, to view updated data of that report in the Query Details Ad Hoc View and Output Formats Ad Hoc View dashlets.

Figure 122: Successful Scheduled Reports Dashboard

Domains

The Domains are provided in the /Public/Completed Schedules/Domains folder. Administrators have access to Domains with the data that is used to create Ad Hoc views. You can change column headers, create calculated fields, and so on, using the data provided in domains. The following Domains are available:

- Successful Reports Domain

- Failed Reports Domain

- Query Details Domain

- Output Formats Domain

Figure 123: Domains in Completed Schedules

Editing the Domains

To edit the data of a domain

Procedure

1. From the list of domains, select the domain and click **Edit**.

2. On the domains page that displays, change the data as required for the Ad Hoc views.

Figure 124: Successful Reports Domain

For more information on editing domains, see *JasperReports Server Data Management Using Domains*.

Troubleshooting

🛠 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're licensed to use, or to upgrade your license, contact Jaspersoft.

This appendix contains the following sections:

- Number of Users Exceeded
- Running Out of Database Connections
- Data Chooser is Slow
- Fields Not Listed in Ad Hoc Editor
- Field Names Disappear in Ad Hoc Editor
- Ad Hoc Filter With All Values Causing Error
- Ad Hoc Dimensions Too Large
- Custom URLs Not Loading in Dashboards
- Print View Not Displaying in Dashboards
- Scheduler Sending Multiple Emails
- Scheduler Not Sending STARTTLS Emails
- Scheduler Running Deleted Jobs
- Scheduler Timezones in Excel Output
- Incorrect Time Stamp Pattern Appearing in Excel Export
- Charts Not Appearing in Excel Export
- Working With Data Sources
- Special Characters in Database Schemas
- Hadoop-Hive Reports Not Running
- Cassandra Reports Not Running
- Extra Comma Appearing in Time Stamp

- Error Exporting Reports, Ad Hoc Views, and Dashboards

- Frequent Issues Encountered with Alerts

Number of Users Exceeded

When you have more users defined in the server than your license allows, the login page displays a warning; users can still log in. After a 24-hour grace period, an email is sent to the administrator and users can no longer log in. Most server functionality is disabled. To re-enable server functionality:

- Contact Jaspersoft sales to purchase additional user licenses. When you install the new license, the server becomes fully functional for all users.

- Delete user accounts until the number of user accounts on the server is in accord with your license. When server functionality is disabled, administrators can still log on and select Manage > Users to delete user accounts. For more information see Managing Users.

Running Out of Database Connections

JasperReports Server manages a pool of connections for each JDBC data source. The default number of connections is 20, but if you run many reports concurrently against the same data source you may reach the connection limit and see degraded performance. In particular, using the web service APIs, REST clients can easily launch many report executions at the same time and reach the limit.

The connection pool size is limited to avoid having too much memory permanently allocated to connections. But if you need more concurrent connections on a regular basis, you can increase the limit with the following configuration:

Reducing the Size Limit for Ad Hoc Dimensions
Configuration File
.../WEB-INF/applicationContext.xml

Reducing the Size Limit for Ad Hoc Dimensions		
Property	Bean	Description
<constructor-arg type="int" value="20"/>	dataSource ObjectPool Factory	Change the default value to match your concurrent connections. Make sure you have enough memory to handle the connections and the concurrent report executions.

If you are using JNDI data sources, you can configure the number of connections in your application server. For more information, see the sections on JNDI in Working With Data Sources.

Data Chooser is Slow

When building a list of available data sources for Ad Hoc Views, the Data Chooser runs sub-queries to filter out ad hoc reports and views that are stored at `/adhoc/topics`. When a large number of ad hoc reports and views are stored in the repository under that location, the sub-query can be slow. To improve performance, you can disable the sub-uery by settingskip.adhoc.datasource.check to true in `jasperserver-pro/WEB-INF/js.config.properties`.

If skip.adhoc.datasource.check is enabled, avoid using ad hoc reports as a data source.

Configuration File
.../WEB-INF/js.config.properties

Property	Description
skip.adhoc.datasource.check	To improve performance, disable the sub-queries by setting skip.adhoc.datasource.check to true.

Fields Not Listed in Ad Hoc Editor

The Ad Hoc Editor supports only certain datatypes. If a Topic contains a field with an unsupported type, the field does not appear when you open the Topic in the Ad Hoc Editor. These are the datatypes supported in the Ad Hoc Editor:

- java.lang.String
- java.lang.Byte
- java.lang.Short
- java.lang.Integer
- java.lang.Long
- java.lang.Float
- java.lang.Double
- java.lang.Number

- java.util.Date
- java.sql.Date
- java.sql.Time
- java.sql.Timestamp
- java.math.BigDecimal
- java.math.BigInteger
- java.lang.Boolean
- java.lang.Object

Unsupported datatypes may occur when editing Topics manually, and sometimes with data sources for big data, in particular MongoDB. The connector for MongoDB uses the datatype of a given value in the last document containing that value, and errors in input files may cause unexpected types. For example, omitting the single quotes in the JSON format causes a string type to be interpreted as a numeric type.

If your Topic or Domain fields do not appear in the Ad Hoc Editor, you can enable logging on the following class to see details of fields with unsupported datatypes:

```
com.jaspersoft.ji.adhoc.metadata.AdhocTopicMetadata
```

For information about enabling logging, see Configuring System Logs.

Field Names Disappear in Ad Hoc Editor

Some fields with international characters in their display names disappear when the field is dragged into the canvas of the Ad Hoc Editor. This is caused by non-Unicode characters used in the field name in the JRXML underlying the selected Topic.

To make the international characters appear in Ad Hoc view field labels, use the resource bundle mechanism:

Procedure

1. Create a resource bundle (*.properties) and associate each of your field labels with a unique key. Use Unicode escape sequences such as \u0153 for the œ character to insert international characters in your label values.

2. Use the $R syntax in the Topic to specify the appropriate key for the label of each desired field.

3. Upload the resource bundle as a resource of the Topic.

When you open the Topic in the Ad Hoc Editor, the labels are displayed correctly from the resource bundle.

This method has the advantage that you can create a resource bundle for each language the Topic needs to support, and users see the labels for the locale they set in their browser.

For more information about localizing JasperReports Server, see Localization.

Ad Hoc Filter With All Values Causing Error

When using filters in the Ad Hoc Editor, your browser sends lists of values to the server with a POST operation. If you filter a field with tens or hundreds of thousands of distinct values, and then select all values, your browser sends megabytes of data in the POST operation. Some application servers are configured to reject such large inputs by default.

For example, if you select 100,000 values in an Ad Hoc filter on a default installation on Tomcat, Tomcat logs an error and redirect the user to the JasperReports Server home page. The Tomcat error log may contain the following entry:

```
2013-09-30 15:12:33,847 ERROR errorPage_jsp,http-8080-6:559 – stack trace of
exception that redirected to errorPage.jsp
java.lang.NullPointerException
```

If you apply filters to fields with large numbers of distinct values, make sure that your app server is configured to accept large input. The following table shows how to configure the Apache Tomcat. For other app servers, refer to your app server's documentation about POST operations.

Configure Apache Tomcat to Accept Large Filter Values	
Configuration File	
<tomcat>/conf/server.xml	
Property	Description
```<Connector port="8080" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443" URIEncoding="UTF-8" maxPostSize="0" /> or maxPostSize="-1" />```	Add the maxPostSize parameter to set the number of bytes accepted by the app server. For Tomcat 8 and 9, "-1" indicates that there is no limit (Tomcat 9 documentation).

Ad Hoc Dimensions Too Large

A column or row dimension in an Ad Hoc crosstab is the equivalent of a row group in an Ad Hoc table. The members of a dimension or group are the unique values used to aggregate the data. Some dimensions and groups can have hundreds or thousands of members. Even if the dimensions are collapsed, the internal engine must calculate and store the values for all dimension members and all cells.

Some datasets contain fields that make huge dimensions on the order of tens of thousands or hundreds of thousands of members. These fields are usually meant to be filtered. But if they are inserted into the crosstab before filtering or by accident, they trigger a large database operation and a large processing load in the Ad Hoc Editor. To limit this impact, you can configure a limit to the number of members in any dimension.

The internal process that calculates groups, dimensions, and members are called the categorizer. Change the following settings to reduce this limit and avoid performance issues with large cardinality dimensions.

Reducing the Size Limit for Ad Hoc Dimensions		
Configuration File		
.../WEB-INF/applicationContext-catFactory.xml		
Property	Bean	Description
maxMembers	baseCategorizer	By default, the limit is set to 100,000 members per dimension. When the limit is reached, all other members are grouped in a member named "Other". Set this property to a lower value based on your typical data and reporting needs.

In addition, dimensions and groups may be nested on several levels, for example Country, Province, and City. If your row data has 100 countries, and each country has 10 provinces, and each province has 10 cities or towns, there will be 100 x 10 x 10 = 10,000 rows in your full crosstab. If you also have two column dimension, each with 10 members, there will be 100 columns in your crosstab and one million cells when all dimensions are fully expanded. This scenario has several implications:

- Nested dimensions and high cardinality dimensions quickly create huge crosstabs.

- Huge crosstabs have a performance impact and take a long time to display and update.

- Consider whether it is possible for users to actually read and interpret such a large crosstab.

- Avoid dimensions with more than 10 members and avoid nesting many levels on each axis.

- Use filters as input controls instead of hiding and expanding dimensions in the crosstab.

For example, it is unlikely that a user can read the expanded data for more than one country at a time. The large report in this example can be replaced with two reports, one that has only the country dimension and allows the user to compare aggregate values from all countries, and another that displays all provinces and cities for a single country selected by a dropdown filter list. Both reports will run much faster than the single large report, and the user will not be blocked waiting for the report to refresh.

Custom URLs Not Loading in Dashboards

Dashboards allow you to specify frames that contain web pages loaded from custom URLs specified when designing the dashboard. These URLs can even include parameters from input controls. If the URL takes too long to load, JasperReports Server will display an error message instead of the content.

If you expect the custom URL to take longer than 10 seconds, you can change the default timeout as follows:

Configuring the Dashboard URL Loading Timeout	
Configuration File	
.../scripts/dashboard.designer.js	
Property	Description
CUSTOM_URL_IFRAME_TIMEOUT	Time in milliseconds that the server will allow a custom URL to load in a dashboard before displaying an error. The setting takes effect immediately when the file is saved, there is no need to restart the server.

Print View Not Displaying in Dashboards

If you cannot display the Print view for a dashboard, there might be an issue with the size of the input control values. Input control values are passed as URL parameters on this page, and the application server can limit the length of the URL that includes the parameters.

To avoid this limit and allow large numbers of input control values in dashboard print view, edit the following configuration file or the equivalent in your application server.

Configure Apache Tomcat to Accept Large Filter Values	
Configuration File	
<tomcat>/conf/server.xml	
Property	**Description**

``
<Connector port="8080"
protocol="HTTP/1.1"
 	
connectionTimeout="20000"
        redirectPort="8443"
        URIEncoding="UTF-8"
        maxPostSize="0"
 	
maxHttpHeaderSize=65535 />
``
	Add the maxHttpHeaderSize parameter to set the number of bytes accepted in the URL by the app server; "65535" is equivalent to 64 KB. For more information, see the Tomcat documentation.

Scheduler Sending Multiple Emails

In cases where you have a cluster of JasperReports Server instances accessing the same repository, the schedulers in each instance can sometimes conflict and send multiple emails. The behavior depends on the run-time of the scheduled reports, because a long report may cause the scheduler on another node to attempt to run the same report before the first node finishes.

To change this behavior, set the following parameter in .../WEB-INF/js.quartz.base.properties:

```
org.quartz.jobStore.clusterCheckinInterval = 900000
```

In case a job fails on the first node, the check-in interval is meant to ensure that the job runs on a second node after this delay. Because the schedulers do not communicate directly, the second scheduler cannot distinguish between a node that had a failure and a node that is still running a job. The default value corresponds to 15 minutes.

This parameter can be adjusted as follows:

- If you have scheduled reports that take a long time to run, longer than 15 minutes, you may see multiple emails. Increase this parameter to an interval longer than your longest report's expected run-time.

- On the other hand, if you have small reports that finish quickly, the default value means that any scheduler or node problem isn't detected by the other scheduler before 15 minutes. If you have time-critical reports scheduled, you can lower this parameter, but the value should still exceed your longest expected report run-time.

Restart all of your server instances after changing this parameter.

Scheduler Not Sending STARTTLS Emails

If you use OpenJDK 7 as your JVM, the scheduler may not be able to send emails if you have configured it to use secure email with STARTTLS. The reports run as scheduled, but the emails are not sent, and an error is recorded in the server log.

This error occurs with the following configuration in the file .../WEB-INF/applicationContext-report-scheduling.xml:

```
<props>
    <prop key="mail.smtp.auth">true</prop>
    <prop key="mail.smtp.starttls.enable">true</prop>
    <prop key="mail.smtp.sendpartial">true</prop>
    ...
</props>
```

To fix this behavior, do one of the following:

- Upgrade to OpenJDK 8.

- Or upgrade to the latest nss library. The error no longer occurs with the following version:

```
Name        : nss
Arch        : x86_64
Version     : 3.19.1
Release     : 3.el6_6
```

Scheduler Running Deleted Jobs

In some cases, old versions of JasperReports Server did not delete the scheduled jobs when deleting a report. These jobs cause errors when the scheduler tries to run them, but you can't remove the jobs through the user interface. The server no longer creates such orphan jobs, but they may appear again when you upgrade or import a catalog that contains them.

If you accidentally imported orphan jobs, make the configuration change shown below and restart your server.

Automatically Deleting Orphan Jobs		
Configuration File		
.../WEB-INF/applicationContext-report-scheduling.xml		
Entry	Bean	Description
autoDeleteBroken UriReportJob	quartz Scheduler	Change this property from false (the default) to true. Orphan jobs are detected and deleted just before they run, so all orphan jobs will be deleted gradually over time.

Scheduler Timezones in Excel Output

When scheduling a report, you can specify the timezone that should be applied to the output instead of using the data source's timezone. This works for all outputs except Excel (XLS), both paginated and non-paginated. To overcome this limitation, you must explicitly enable timezone usage in XLS output.

Setting Timezones in Scheduled Excel (XLS) Reports
Configuration File

Setting Timezones in Scheduled Excel (XLS) Reports		
.../WEB-INF/classes/jasperreports.properties		
Property	**Value**	**Description**
net.sf.jasperreports.export. xls.use.timezone	true	Define this property as true to apply timezones in scheduled Excel (XLS) reports.

This property can also be defined on individual elements in the report's JRXML if only certain fields should apply the timezone.

Incorrect Time Stamp Pattern Appearing in Excel Export

When exporting reports to Excel, the time stamp appears in the following pattern "mmm d, yyyy h:mm:ss \a" instead of "MMM d, yyyy h:mm:ss AM/PM".

To resolve this, you need to add mapping to the map in applicationContext.xml. It ensures date patterns in Java 11 will convert to corresponding Excel date patterns. And Excel export will have the correct time stamp, that is, "MMM d, yyyy h:mm:ss AM/PM".

To add the mapping

Procedure

1. Open the file .../WEB-INF/applicationContext-webapp.xml for editing.

2. Locate the `formatPatternsMap` bean and add the following new entry.

   ```
   <entry key="d MMM, yyyy, h:mm:ss a" value="d MMM, yyyy h:mm:ss AM/PM"/>
   ```

3. Restart the server.

Charts Not Appearing in Excel Export

When exporting a report to Excel, JasperReports Server usually removes images that decorate the report and that do not fit in the Excel data-centric layout. However, JasperReports Server also converts any charts to images and uses the special property net.sf.jasperreports.export.xls.ignore.graphics set to false to make the image appear. If your report does not set this property explicitly, the chart images do no appear in your reports when exported to Excel.

If you have a lot of reports with this issue, you can set the property on the server:

Charts Images in Excel Export	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	Description
net.sf.jasperreports. export.xls.ignore.graphics	By default, this property is set to true; in this case, images and chart images from the report do not appear when exported to Excel. Set this property to false to make chart images appear in Excel exports.

Working With Data Sources

When adding a data source to JasperReports Server, several things can cause errors. Start by looking at the following general connectivity issues:

- Check that your database server is available and accepting TCP/IP connections from the host where JasperReports Server is installed.

- Check in your RDBMS that the username and password you are using are correct and have access to the selected database.

- Check for firewalls or network connectivity errors.

Many databases, including MySQL, also require the user grants to include the specific host from which connections are allowed. Otherwise, when testing the JDBC connection, a

connection may not be allowed even though the username and password are correct. For more information, refer to the MySQL documentation for setting up users.

An easy way to test connectivity from the server to the database with a particular user is to use a tool such as SQuirreL or another DB query tool to connect to the database from the host of your JasperReports Server instance.

Logging JDBC Operations

You can enable additional logging to help you find the cause of the error. Set any or all of the following loggers in the server settings interface or in the .../WEB-INF/log4j.properties file:

- log4j.logger.com.jaspersoft.jasperserver.api.engine.jasperreports.service.impl.JdbcDataSourceService
- log4j.logger.com.jaspersoft.jasperserver.api.engine.jasperreports.service.impl. JndiJdbcDataSourceService
- log4j.logger.com.jaspersoft.jasperserver.war.action.ReportDataSourceAction
- log4j.logger.com.jaspersoft.commons.datarator.JdbcDataSet
- log4j.logger.com.jaspersoft.jasperserver.war.common.JasperServerUtil
- log4j.logger.com.jaspersoft.commons.semantic.dsimpl.JdbcDataSetFactory
- log4j.logger.com.jaspersoft.commons.semantic.metaapi.impl.jdbc.BaseJdbcMetaDataFactoryImpl
- log4j.logger.com.jaspersoft.jasperserver.war.validation.ReportDataSourceValidator

JDBC Drivers

JasperReports Server ships with drivers for several databases, as listed in the dialog for creating data sources. If the JDBC driver for your database is not included, the system administrator can easily upload the driver and use it immediately to access a data source.

Database Permissions

When creating database users, you must ensure that they have the appropriate privileges to access data, as well as permission to connect from the server that JasperReports Server is running on.

- The database user that you specify in your data source definition needs the privileges to run SELECT queries on the tables used in your reports. The server blocks any DROP, INSERT, UPDATE, and DELETE SQL commands through its SQL injection protection. In some cases, additional permissions may be required to execute stored procedures, depending on your configuration and needs.

- If you accept the defaults during installation of JasperReports Server on Linux from an RPM using apt-get, rpm, or yum, the bundled PostgreSQL allows only the user who owns PostgreSQL to connect. Enter the following commands to connect:

  ```
  su - postgres
  psql -U postgres
  ```

- Many databases, including MySQL, also require that the user permission name the specific host from which connections are allowed. Otherwise, when testing the JDBC connection, a connection may not be allowed even though the username and password are correct. For example, see the MySQL documentation for setting up users.

A fairly easy way to test permissions and connectivity is to use a tool such as SquirrelSQL or another DB query tool to connect to the database from the same host as JasperReports Server and to run typical queries against your database.

Unique JDBC Data Source Fields

When you choose a JDBC driver, the data source creation wizard displays the fields that are required for your database. In some cases, you may need to enter information that is unique to this type of JDBC data source. The following tables explain the field for the most common databases.

Autonomous REST

Field	Description
Authentication Method	The authentication method for accessing the data source. Changing the authentication method requires you to configure the URL. The authentication method can be one of the following: • none - Does not use an authentication method for accessing the data source. • basic - Uses a user ID and password for authentication Requires user and password properties in the URL. • HttpHeader - Uses a security token for authentication. Requires user and SecurityToken properties in the URL. • UrlParameter - Uses a security token and URL parameter for authentication. The URL requires user, SecurityToken, and AuthParam properties. • OAuth2 - Uses OAuth 2.0 for authentication. There are multiple ways to use OAuth 2.0. Jaspersoft recommends using the official driver documentation for more information. • Custom - Uses the custom token-based authentication defined in the config file. Jaspersoft recommends using the official driver documentation for more information. By default, the authentication method is set to none.
Path to Config File	The full path to the configuration file for the data source.

Cassandra

Field	Description
Key Space	The name of the key space that acts as the data store for the Cassandra data source.

Google BigQuery

Field	Description
Project ID	The unique identifier for your Google Cloud project.
Dataset ID	The unique BigQuery dataset name.
Google Service Account Email Address	The email address associated with the BigQuery service account.
Private Key Path	The full path to the .json key file used to authenticate the service account email address.
	The .p12 files are not supported.

Impala

Field	Description
Schema Name	Name of the Impala Schema object.

Oracle

Field	Description
Service	Name of the Oracle database service.

Salesforce

Field	Description
Security Token	A case-sensitive alphanumeric key is used in combination with a password to access the Salesforce data source.
Maximum Number of Web Service Calls	The maximum number of times JasperReports Server calls the Salesforce web service in a 24-hour period. Set to 0 for unlimited calls.

JDBC Database URLs

When you choose a JDBC driver, the data source creation wizard prompts you for the elements of the URL that are required for your database. In some cases, you may need to add certain arguments to the JDBC URL. Ensure that the database URL you entered when defining your JDBC data source is consistent with what is required for your specific database and database driver. The following table gives the default URLs and port numbers and examples of optional arguments supported by the most common databases:

Database	Default JDBC Database URL
PostgreSQL	jdbc:postgresql://<host>:5432/<db-name>
MySQL and	jdbc:mysql://<host>:3306/<db-name>?tinyInt1isBit=false

Database	Default JDBC Database URL
MariaDB	
Oracle	jdbc:oracle:thin:@<host>:1521:ORCL
SQL Server	jdbc:sqlserver://<host>:1433;databaseName=<db-name>;sendTimestampEscapeAsString=false;encrypt=true;trustServerCertificate=true;sslProtocol=TLSv1.2
IBM DB2	jdbc:db2://<host>:50000/<db-name>:driverType=4;currentSchema=<DB-SCHEMA>;
Cassandra	jdbc:cassandra://<host>:9042;DefaultKeyspace=

Enabling the JDBC Driver for ElasticSearch Data Sources

The ElasticSearch JDBC driver is not enabled by default due to certain limitations. If you want to use the driver, you can modify the .../WEB-INF/applicationContext-webapp.xml file to make it active.

> Be aware of the following issues when using an ElasticSearch data source:
> - It is not supported for JBoss or Wildfly app servers.
> - Table joins are not supported.
> - There may be issues with calculated fields, including more complex aggregation functions, using* as a special character, and the use of calculated fields in crosstabs and charts.
> - When an ElasticSearch data source is used in a virtual data source, the virtual data source only displays the Base tables of the ElasticSearch data source, not the views, when used in a domain.

To enable the Jaspersoft JDBC drivers for ElasticSearch data sources

1. Open the file .../WEB-INF/applicationContext-webapp.xml for editing.

2. Locate the jdbcTibcoConnectionMap bean and find the following lines for the inactive

ElasticSearch JDBC driver:

```
<!--  entry key="elastic_search">
   ...
</entry -->
```

3. Modify the lines as follows to enable the JDBC driver in JasperReports Server:

```
<entry key="elastic_search">
   ...
</entry>
```

4. Save the file and restart JasperReports Server.

JNDI Services on Apache Tomcat

If you have trouble with a JNDI connection, you need to look at the JNDI definition for your database on your application server. This section gives common issues with JNDI definitions on Apache Tomcat connecting to MySQL. If you use a different application server or database server, refer to its documentation.

A JNDI connection on Tomcat is defined in two different files. Make sure both have the following information:

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

```
<Resource name="jdbc/<db-name>" auth="Container" type="javax.sql.DataSource"
maxActive="100" maxIdle="30" maxWait="10000"
username="<db-user>" password="<db-user-password>"
driverClassName="org.postgresql.Driver"
validationQuery="SELECT 1" testOnBorrow="true"
url="jdbc:mysql://<host>:3306/<database>?autoReconnect=true&amp;autoReconnect
ForPools=true"/>
```

- <tomcat>/webapps/jasperserver/WEB-INF/web.xml

```
<resource-ref>
<description>JNDI Example</description>
<res-ref-name>jdbc/<db-name></res-ref-name>
<res-type>javax.sql.DataSource</res-type>
<res-auth>Container</res-auth>
</resource-ref>
```

Also check the following points:

- Ensure that the driver for your database connection is in the <tomcat>/lib folder.

- Ensure the database user has the privileges to run SELECT queries on the tables used in your reports. In some cases, additional permissions may be required to execute stored procedures, depending on your configuration and needs. For more information, see Database Permissions.

- If you installed JasperReports Server from a WAR file, Tomcat may have created a separate copy of context.xml in <tomcat>/conf/Catalina/Localhost/jasperserver.xml. See the corresponding section in the troubleshooting appendix of the JasperReports Server Installation Guide.

- See the Apache Tomcat documentation for JNDI data sources.

- For Oracle databases, you may need to specify additional parameters in the context.xml file. For example, to support in Oracle, add the following line:

```
driverClassName="oracle.jdbc.OracleDriver"
validationQuery="SELECT 1 FROM dual"
accessToUnderlyingConnectionAllowed="true"
```

JNDI Services on JBoss

Follow these steps to configure JasperReports Server to use JNDI data sources with JBoss:

1. Add a new JNDI service user to <jboss>/standalone/deployments/jasperserver.war/WEB-INF/js-jboss7-ds.xml. For example:

```
<datasource jta="false" jndi-name="java:/jdbc/sample_db" pool-name="sample_db" enabled="true" use-
ccm="false">
        <connection-url>jdbc:postgresql://localhost:5432/sample_db</connection-url>
        <driver>postgresql-9.4-1210.jdbc41.jar</driver>
        <security>
            <user-name>postgres</user-name>
            <password>postgres</password>
        </security>
        <pool>
            <min-pool-size>5</min-pool-size>
            <max-pool-size>50</max-pool-size>
            <prefill>true</prefill>
        </pool>
        <validation>
            <validate-on-match>false</validate-on-match>
```

```
            <background-validation>false</background-validation>
            <check-valid-connection-sql>SELECT 1</check-valid-connection-sql>
        </validation>
        <statement>
            <share-prepared-statements>false</share-prepared-statements>
        </statement>
    </datasource>
```

JNDI Services on WebLogic

Follow these steps to configure JasperReports Server to use JNDI data sources with WebLogic:

1. Append the following definition to the <reference-descriptor> node of .../WEB-INF/weblogic.xml:

   ```
   <resource-description>
       <res-ref-name>TestDatabase</res-ref-name>
       <jndi-name>jdbc/testDatabase</jndi-name>
   </resource-description>
   ```

2. Append the following definition to .../WEB-INF/web.xml:

   ```
   <resource-ref>
       <description>TestDatabase database</description>
       <res-ref-name>TestDatabase</res-ref-name>
       <res-type>javax.sql.DataSource</res-type>
       <res-auth>Container</res-auth>
   </resource-ref>
   ```

3. In the WebLogic Admin Console, create a JNDI data source, in this example its JNDI name would be TestDatabase.
   Ensure that the database user in your JNDI definition has the privileges to run SELECT queries on the tables used in your reports. In some cases, additional permissions may be required to execute stored procedures, depending on your configuration and needs. For more information, see Database Permissions.

4. Restart the jasperserver instance using the WebLogic Admin Console.

Creating a Data Source on SQL Server Using Windows Authentication

If your database is Microsoft SQL Server and you use Windows Authentication (also called Integrated Security), use the following procedure to create a data source.

1. Download the latest JDBC driver for your version of Microsoft SQL Server. For example download Microsoft SQL Server JDBC Driver 6.4.

   > To find the latest version of your Microsoft SQL Server driver, see the Microsoft JDBC Driver for SQL Server Support Matrix.
   >
   > For information about downloading and installing Microsoft SQL Server drivers, see the Microsoft JDBC Driver for SQL Server page.

2. Download and run the self-extracting executable: sqljdbc_6.4.0.0_enu.exe

3. Open the extracted folder sqljdbc_6.4\enu\auth\x64 and copy the file sqljdbc_auth.dll to the folder your app server automatically searches for DLLs.
   For Tomcat, this is the <tomcat>\bin folder.

4. Open the extracted folder sqljdbc_6.4\enu and copy mssql-jdbc-6.4.0.jre8.jar to the folder to let your app server automatically searches for jars. For Tomcat, this is the <tomcat>\lib folder.

5. Restart your app server.

6. Log into JasperReports Server as an administrator.

7. Select Create > Data Source from the main menu.

8. In the Type field, select JDBC Data Source. The page refreshes to show the fields necessary for a JDBC data source.

9. Enter a name and optional description for your data source.

10. From the dropdown field, select com.microsoft.sqlserver.jdbc.SQLServerDriver.

11. Enter the database hostname and database name of your SQL Server instance.

12. In the URL field, add the following string to the end of the generated URL:
    ;integratedSecurity=true

13. In the username field, enter any non-blank string you want, for example none.

14. In the Password field, enter any non-blank string you want, for example none.

15. Set the Time Zone and Save Location fields if necessary.

16. Click Test Connection and verify that the connection works.

17. Click Save to save the data source in the repository.

Upgrading Bean Data Sources

There was a change in the Spring configuration for JasperReports Server 6.0 that changes how some of the existing Spring beans are made accessible for use by other beans. This can break existing custom data sources. This change specifically affects beans that implement the interface ReportDataSourceServiceFactory.

Prior to release 6.0, if the code in JasperReports Server accessed a bean of this type, it would get the actual instance of the Spring bean as configured in the Spring XML file, and it could be cast to the concrete class. In 6.0 and later, these beans were intercepted to implement the profile attributes feature, and instead of seeing the actual instance, other code would see a dynamic proxy instead of the actual bean.

Dynamic proxies are a Java feature, which allows classes to be generated at run time that implement any interface that can be loaded on the classpath. The resulting object can be cast to any of those interfaces, but doesn't correspond to any concrete class. For more information:

> http://www.javaworld.com/article/2076233/java-se/explore-the-dynamic-proxy-api.html

Since proxies can only represent interfaces, existing code that tries to cast the bean to a concrete class will break. Casting is usually done to get access to methods on a more specific class or interface. As long as the code is not casting the bean to a concrete class, it will work, so there are two ways to get around this problem:

- If the code needs to access methods on an existing interface, do a cast to that interface, or inject the property using the existing interface, so no cast is needed.

- If the code needs to access methods that are not on an existing interface, simply create an interface with the methods needed, and have the target object implement that interface.

For example, let us say you have a bean with id myBean that needs to access the jdbcDataSourceServiceFactory, configured as follows:

```
<bean id="jdbcDataSourceServiceFactory" class="com.jaspersoft.jasperserver.
api.engine.jasperreports.service.impl.JdbcReportDataSourceServiceFactory">
...
</bean>
```

Where myBean has the following Spring configuration:

```
<bean id="myBean" class="example.MyBean">
    <property name="jdbcDSSF" ref="jdbcDataSourceServiceFactory"/>
</bean>
```

The following code worked before but will now cause an error:

```
public class MyBean {
  private ReportDataSourceServiceFactory jdbcDSSF;

  public ReportDataSourceServiceFactory getJdbcDSSF() {
    return jdbcDSSF;
  }

  // before 6.0, was called by Spring with the actual bean
  // 6.0 and after, is called with a dynamic proxy
  public void setJdbcDSSF(ReportDataSourceServiceFactory jdbcDSSF) {
    this.jdbcDSSF = jdbcDSSF;
  }

  public void doSomething() {
    // this code used to work, but now it will break
    ((JdbcReportDataSourceServiceFactory) jdbcDSSF).createService();
    ((JdbcReportDataSourceServiceFactory) jdbcDSSF).doSomethingElse();
  }
}
```

Because the first call is a method that is part of the ReportDataSourceServiceFactory, the cast is unnecessary; to fix it, just leave it out:

```
        jdbcDSSF.createService();
```

In this example, JdbcReportDataSourceServiceFactory has a method called doSomethingElse(). This method is not part of any interface, but you can create an interface that includes it:

```
public interface MyDSSF extends ReportDataSourceServiceFactory {
    public void doSomethingElse();
}
```

JdbcReportDataSourceServiceFactory would need modification so that it implements this new interface:

```
public JdbcReportDataSourceServiceFactory implements MyDSSF {
....
}
```

You do not have to change the declaration in MyBean because Spring generates a dynamic proxy implementing MyDSSF, but if you change the declaration, the code will be easier to understand because no casts will be necessary:

```
public class MyBean {
  private MyDSSF jdbcDSSF;

  public MyDSSF getJdbcDSSF() {
    return jdbcDSSF;
  }

  // called with a dynamic proxy which implements all needed interfaces
  public void setJdbcDSSF(MyDSSF jdbcDSSF) {
    this.jdbcDSSF = jdbcDSSF;
  }

  public void doSomething() {
    // no need to cast
    jdbcDSSF.createService();
    jdbcDSSF.doSomethingElse();
  }
}
```

Special Characters in Database Schemas

Some databases allow you to use spaces and special characters in their database schema. These spaces and symbols can thus appear in table names and column names. However, in order to define Domains, JasperReports Server relies on a restricted set of symbols. Databases having table names and column names with forbidden symbols will cause Domain errors and be unusable with Domains.

As of JasperReports Server 6.0, Domains allow the following symbols:

Symbols Allowed in Table Names	Symbols Allowed in Column Names
<space> @ # $ % & _ /	<space> @ # $ % & ( ) _ [ ] : , + - /

These symbols have been tested with the following databases:

- SQL Server

- DB2

- PostgreSQL

Hadoop-Hive Reports Not Running

If you created a Hadoop-Hive data source in some older JasperReports Server versions before Version 6, it may not run when imported to a more recent version of the server. This problem is caused by the changes to the Hadoop-Hive connectors.

To update Hadoop-Hive data sources

Procedure

1. Create a new Hive data source using the same URL as the old Hadoop-Hive data source used by your reports. If you're using Cloudera 5, create a CDH 5 Hive-Impala data source. For all other hive servers, create a JDBC data source that uses the Hive JDBC driver. For instructions, see Hadoop-Hive Data Sources.

2. Update your Hadoop-Hive-based reports to use the new data source.

3. Delete the old, imported data source.

Cassandra Reports Not Running

The Cassandra architecture introduces some restrictions that other databases do no have. Certain filters such as is-one-of (IN), <, and > cannot be used with all fields. For example, you may see the following error:

```
Caused by: com.datastax.driver.core.exceptions.InvalidQueryException: Cannot use IN operator
on column not part of the partition key
```

Cassandra clusters use partition keys to split data between nodes, but this imposes a certain segmentation of the data. This segmentation is not compatible with all filtering, because it would be very inefficient. Therefore, the definition of the partition keys in your Cassandra schema limits you to certain comparison operations on certain fields.

In general you should avoid comparison operators other than strict equality in filters on Cassandra data sources. The Cassandra reference documentation recommends not using the IN comparison "under most conditions."

For more details, see the Cassandra documentation of the SELECT WHERE clause in CQL (Cassandra Query Language).

Maximum Parameter Size in Wildfly

On Wildfly, if a report sends more than 1,000 parameters, you may see a 500 error, for example:

```
"The number of parameters exceeded the maximum of 1000"
```

To allow more than 1,000 parameters, increase max-parameters. To do this, open the file <wildfly-home>/wildfly/standalone/configuration/standalone.xml and add or change the max-parameters attribute of the http-listener property. For example:

```
<http-listener name="default" socket-binding="http" max-header-size="974247881"
    max-post-size="974247881" max-parameters="5000"/>
```

Extra Comma Appearing in Time Stamp

Java 11 changed the default date format to conform to CLDR 33, which results in an extra comma appearing in the time stamps for Domains, Ad Hoc views, and reports. If you are running on Java 11 and want backwards compatibility with Java 8 date formatting, you need to set the following parameter on all nodes:

```
-Djava.locale.providers=JRE,COMPAT,CLDR
```

For JasperReports Server to show reports correctly in Java 11, we recommend running with these parameters in the given order.

For more information on setting JVM options, see the JasperReports Server Installation Guide.

Error Exporting Reports, Ad Hoc Views, and Dashboards

The export of Reports, Ad Hoc Views, and Dashboards fails when Tomcat is run as root in the JasperReports Server installation on Linux. The Tomcat log file displays an error, for example:

```
2020-06-11T17:32:19,031 ERROR SecureExceptionHandlerImpl,http-nio-8080-
exec-8:116 -
com.github.kklisura.cdt.launch.exceptions.ChromeProcessTimeoutException:
Failed while waiting for chrome to start: Timeout expired! Chrome
output: [0611/173218.897370:ERROR:zygote_host_impl_linux.cc(89)] Running
as root without --no-sandbox is not supported. See
https://crbug.com/638180.
```

To resolve this, you need to set the following property in the jasperreports.properties file:

- net.sf.jasperreports.chrome.argument.no-sandbox=true

After setting this property, restart JasperReports Server to enable it.

For information about configuring this property, see Configuring a JavaScript Engine for Graphical Report Rendering.

Frequent Issues Encountered with Alerts

This section includes the list of common troubleshooting scenarios that might occur when alerts are set on the report.

Alert fails to trigger when the data source is modified

Alert is not triggered when the data source is modified for report on which alert is set, resulting in the user not receiving any email notification. To receive an alert notification, it is necessary to fix the data source connection back to correct values.

For more information, see Data Sources section in JasperReports Server Administrator Guide.

Alert fails to trigger when measure is edited from domain designer

Alerts is not triggered when the report measure are modified on which alert is set from domain designer, resulting in the user not receiving any email notification. To receive an alert notification, the user must edit domain and change back the report measure value to its original value and re-save it.

For more information, see Working with the Domain Designer section in JasperReports Server Data Management Using Domains Guide.

Alert fails to trigger when the visualization type is changed

Alert is not triggered when the visualization type is changed to a different type (e.g., changing from Table report to Crosstabs report) for which the user has set an alert, resulting in the user not receiving any email notification. To receive an alert notification, the visualization type should be changed back to its original setting, that is the visualization type on which the alert was initially set.

For more information, see The Visualization Selector section in JasperReports Server User Guide.

Alert fails to trigger when data mode is changed

Alert is not triggered when data mode is changed to a different mode. For example, if an alert is set for the Totals value of a column in a table report, and the user changes the Data Detail mode to Details, resulting in the user not receiving any email notification. To receive alert notifications, the Data Detail mode must be changed back to its original value i.e. Totals value on which the alert was initially set or edit the same alert to new threshold value.

For more information on users, see Controlling the Data Set section in JasperReports Server User Guide.

Alert fails to trigger when the summary function is removed

Alerts is not triggered when summary data is removed. For example, if an alert is set up based on a summary function to obtain the Totals value of a column in the report, and the user removes the summary function, resulting in the user not receiving any email notification. To receive alert notifications, the summary function should be changed back to its original setting, the type on which the alert was initially set, or edit the same alert to new threshold value.

To add or remove data summaries from all columns, see the Summaries section in JasperReports Server User Guide.

Alert fails to trigger when a user is disabled

Alert is not triggered when a user who had set an alert is disabled, resulting in the user not receiving any email notification.

To re-enable the user, click **Manage > Users** on the Admin Home page.

For more information on users, see Managing Users section in JasperReports Server Administrator Guide.

Localization

By default, JasperReports Server is presented in the English language (US version), but it supports other languages with translations that include data formats and resource bundles. Supported languages are Brazilian Portuguese, Chinese (Simplified), French, German, Italian, Japanese, and Spanish. The translations are included in your JasperReports Server instance by default; to view the application in a specific locale, select it on the login page.

If you need to support a language other than the supported ones, you can localize JasperReports Server, including translating it into a different language by providing labels and messages in the preferred language. For other locales, you may also need to change the default locale and time zone. This chapter describes the procedures and gives a few examples.

For information about localizing Topics, Domains, and reports, refer to the JasperReports Server User Guide and JasperReports Server Data Management Using Domains.

This chapter contains the following sections:

- Configuring JasperReports Server for Multibyte Fonts
- UTF-8 Configuration
- Changing Character Encoding
- Creating a Locale
- Configuring JasperReports Server to Offer a Locale

Configuring JasperReports Server for Multibyte Fonts

Although translation packs for Chinese and Japanese ship with JasperReports Server, the fonts that it uses by default do not support those languages. Therefore, if your organization requires those fonts, you need to configure JasperReports Server for them.

While the fonts JasperReports Server uses are generally dictated by the JRXML files that define your reports, some font configuration is required for special circumstances. For

example, you can configure Jaspersoft OLAP to offer different options in the Chart Default Font field in the Chart Options dialog. But note that, in order to use a font, the font must be available to the host's operating system. This section describes steps you may need to take, depending on the functionality you use and the locales you support.

The tasks in this section require you to edit these files:

File Name	Location	Purpose of Edits
jpivot_internal_messages.properties	.../WEB-INF/internal	Specifying chart fonts for Jaspersoft OLAP Community
Ja_pro_internal_messages.properties	.../WEB-INF/internal	Specifying chart fonts for Jaspersoft OLAP Professional and Enterprise
userConfig.xml	.../WEB-INF/jpivot/print	Embedding fonts in PDF

Enabling East Asian Fonts

The default configuration of the Java Runtime Environment (JRE) does not support East Asian fonts. If your locale requires such a font, you need to configure your users' computers for the fonts and update their JRE.

To configure a Microsoft Windows computer (XP and later) for East Asian fonts

Details of this procedure vary, depending on your version of Windows.

Procedure

1. In the Control Panel, click Region and Language.

2. In the Region and Language dialog, select the Keyboards and Languages tab.

3. On the tab, install the language(s) that you need.

4. If necessary, install the related keyboard modifications.

5. Close the control panel.

6. Locate the fontconfig.properties.src file in the C:\Program Files\<JRE_directory>\lib folder.

7. In the file, locate the following line:
    sequence.allfonts=alphabetic/default,dingbats,symbol

8. Change the line to include the East Asian fonts that you need, such as the following:
    sequence.allfonts=alphabetic/default,dingbats,symbol,korean,japanese,chinese-ms936,chinese-ms950

9. At the end of the file, check to be sure that the fonts you selected are listed, as in the following:
    filename.Gulim=gulim.TTC
    If the fonts are not listed, add them.

10. Save and close the file.

11. Rename the file to fontconfig.properties in the file system.

Configuring Ad Hoc Charts for Asian Fonts

The default font for the legend of an Ad Hoc chart does not support some Asian characters, such as Japanese kana glyphs. Default fonts for Ad Hoc reports are defined in the following file:

Changing Ad Hoc Chart Legend Fonts

Configuration File

.../WEB-INF/adhoc/themes/actual_size.<ver>.jrxml

Property	Value	Description
style name= "ChartLegend" fontName=	"fontName=" <default>	The default value, DejaVu Sans, does not include kana glyphs, and thus the legend contains blank symbols.
	"SansSerif"	If the server machine has system Japanese fonts available, setting the value to SansSerif should work in most Java font configurations.

After making changes to the actual_size.<ver>.jrxml file and importing the modified file into JasperReports Server, you must create new Ad Hoc reports for charts that contain Asian characters using the modified report template.

If Japanese fonts are not installed in the server machine, you can install them and create a custom report template to support them in the following manner:

Procedure

1. Package a Japanese font as a font extension JAR.

2. Add the new JAR to the .../WEB-INF/lib folder of the JasperReports Server WAR file.

3. Log into JasperReports Server as an administrator and export the Actual Size template from the Public > Templates folder in the repository. The procedures for exporting and importing a template are described in the *JasperReports Server Ultimate Guide*.

4. In the exported file folder, find resources/public/templates/actual_

size.<ver>.jrxml.data and rename the file to remove the .data extension.

5. Open actual_size.<ver>.jrxml to edit it.

6. Locate the following line:
   style name="ChartLegend" ... fontName="DejaVu Sans"
   Modify the line as follows:
   fontName="SansSerif"

7. In JasperReports Server, browse to the Public > Templates folder in the repository.

8. Right-click the Templates folder and select Add Resource > File > JRXML from the context menu to import the edited actual_size.<ver>.jrxml file as a new custom report template.

You can use the new custom report template for charts that contain Asian characters.

Configuring OLAP Options for Chart Default Fonts

If you implement Jaspersoft OLAP and support a locale with special font requirements, you can configure Jaspersoft OLAP to offer different options in the Chart Default Font field in the Chart Options dialog of the OLAP view. This may be necessary if you implement locales that Latin 1 doesn't support.

An OLAP view's Chart Options dialog includes the Chart Default Font field, which allows users to select the font to use in charts. The default options are SansSerif, Serif, and MonoSpaced. JasperReports Server reads these values from a properties file and attempts to map them to fonts available in the server host's operating system. You can configure the server to offer different fonts if these fonts don't support the locales you implement.

To change the Chart Default Font field's options

Procedure

1. Save the jpivot_internal_messages.properties file with a new name that reflects the new locale. For example, for Japanese, the new file would be called jpivot_internal_messages_ja.properties.

2. Open the new file and locate the following keys:

```
JAJ_000_jsp.jpivot.chartpropertiesform.sansSerif=SansSerif
```

```
JAJ_000_jsp.jpivot.chartpropertiesform.serif=Serif
JAJ_000_jsp.jpivot.chartpropertiesform.monospaced=Monospaced
```

📝 If you are using Jaspersoft OLAP Community Edition, the name of the file and the keys that you edit are different. For the Community Edition, open the jpivot_internal_message.properties file and edit these keys:

```
jsp.wcf.chart.sansserif=SansSerif
jsp.wcf.chart.serif=Serif
jsp.wcf.chart.monospaced=Monospaced
```

3. Change one or more of the strings to the name of a font available in the host's operating system. For example, if you wanted to change the SansSerif font to the SimHei font, edit the value specified by jsp.wcf.chart.sansserif. For example:

```
jsp.wcf.chart.sansserif=SimHei
```

4. Save the file.

5. Restart JasperReports Server.

Embedding Fonts in PDF Output for OLAP Views

📝 By default, JasperReports Server can create PDF (Portable Document Format) files with many different fonts. However, if you experience font problems in the PDF output, you may need to take the steps described in this section to make the fonts available to JasperReports Server's XSL Formatting Object (XSL-FO) processor.

You must have distribution rights to a font in order to embed it in a PDF file.

When users export OLAP views in PDF format, JasperReports Server generates the PDF output using Apache FOP (Formatting Objects Processor). In order for FOP to render fonts properly, you must install the font itself (for example, a TTF file) on the server host, create a font metrics file (using Apache's org.apache.fop.fonts.apps.TTFReader utility), and update the userConfig.xml file to associate the font with its metrics. For more information, refer to the Apache FOP documentation.

You can embed any Unicode font using this procedure, though larger font files may have significantly larger memory footprints. To keep memory requirements small, we recommend you use the smallest font file you can, such as SimHei to support Chinese, Japanese, and Korean.

UTF-8 Configuration

JasperReports Server uses UTF-8 (8-bit Unicode Transformation Format) character encoding. If your database server or application server uses a different character encoding form, you may have to configure them to support UTF-8. This section provides information for configuring the character encoding for several application servers and database servers. If you use a different application server or database, and its default character encoding isn't UTF-8, you may need to make similar updates to support certain locales. For more information, refer to the documentation for your application server or database.

Java Options

Make sure that your Java environment is set to support UTF-8. You can set this option as follows:

Linux	export JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8
Windows	set JAVA_OPTS=%JAVA_OPTS% -Dfile.encoding=UTF-8

You may see an error in a report that uses Unicode encoding if the -Dfile.encoding option is not set to UTF-8.

Tomcat

By default, Tomcat uses ISO-8859-1 (ISO Latin 1) character encoding for URIs, which is sufficient for Western European locales, but does not support many locales in other parts of the world.

If you plan to support locales that Latin 1 does not support, you must change Tomcat's URI encoding format.

> If you chose the instance of Tomcat that is bundled with the installer, you don't need to make this change. The bundled Tomcat is preconfigured to support UTF-8. If you installed the WAR file distribution with your own instance of Tomcat and want to support UTF-8, follow this procedure.

To configure Tomcat to support UTF-8

Procedure

1. Open the conf/server.xml file and locate the following code:

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<Connector>
    port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true"
</Connector>
```

2. At the end of this section, insert the following line before the closing tag:

```
URIEncoding="UTF-8"
```

3. For example, after your changes, the section might read:

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
    <Connector>
    port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true"
    URIEncoding="UTF-8"
</Connector>
```

4. Save the file.

5. Restart Tomcat.

PostgreSQL

JasperReports Server requires PostgreSQL to use UTF-8 character encoding for the database that stores its repository as well as for data sources. A simple way to meet the requirement is to create the database with a UTF-8 character set. For example, enter the following command:

```
create database jasperserver encoding='utf8';
```

MySQL

By default, MySQL uses ISO-8859-1 (ISO Latin 1) character encoding. However, JasperReports Server requires MySQL to use UTF-8 character encoding for the database that stores its repository as well as for data sources. The simplest way to meet the requirement is to create the database with a UTF-8 character set. For example, enter the following command:

```
create database jasperserver character set utf8;
```

To support UTF-8, the MySQL JDBC driver also requires that the useUnicode and characterEncoding parameters be set as in this startup URL:

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

If the MySQL database is a JNDI data source managed by Tomcat, such as the JasperReports Server repository database, the parameters can be added to the JDBC URL in .../META-INF/context.xml. The following is a sample resource definition from that file:

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

JBoss

JBoss ignores the context.xml file. Instead define JNDI data sources in <jboss>/standalone/deployments/jasperserver.war/WEB-INF/js-jboss7-ds.xml. For an example configuration, see JNDI Services on JBoss.

If the database is a JDBC data source configured in the repository, change the JDBC URL by editing the data source in the JasperReports Server repository. The following is an example of the JDBC URL (note that the ampersand isn't escaped):

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

Oracle

Oracle databases have both a default character set and a national character set that supports Unicode characters. Text types beginning with N (NCHAR, NVARCHAR2, and NCLOB) use the national character set. All the text data used by the JasperReports Server repository (when stored in Oracle) is stored in NVARCHAR2 columns, so that JasperReports Server metadata can use the full Unicode character set. For more information about Unicode text support, refer to the Oracle white paper (PDF).

To work properly with Unicode data, the Oracle JDBC driver requires you to set a Java system property by passing the following argument to the JVM:

```
-Doracle.jdbc.defaultNChar=true
```

In Tomcat, add the variable to JAVA_OPTS in <jboss>/bin/standalone.conf (Linux) or <jboss>\bin\standalone.conf.bat (Windows):

Procedure

1. Locate the following line in the script:

Linux	# Set the default -Djava.endorsed.dirs argument
Windows	rem Set the default -Djava.endorsed.dirs argument

2. Add the following line before it:

Linux	JAVA_OPTS="$JAVA_OPTS "-Doracle.jdbc.defaultNChar=true
Windows	set JAVA_OPTS=%JAVA_OPTS% -Doracle.jdbc.defaultNChar=true

Because JBoss also uses JAVA_OPTS to pass options to the JVM, you can add the same line to bin/run.sh (Linux) and bin/run.bat (Windows). Add it before this line:

Linux	# Setup the java endorsed dirs
Windows	rem Setup the java endorsed dirs

Changing Character Encoding

Depending on the third-party software used and supported locales, you may also have to configure JasperReports Server and its host.

The steps described in this section are necessary only under certain circumstances, such as if you plan to use a character encoding form that UTF-8 cannot handle.

To use a character encoding form other than UTF-8, you must configure JasperReports Server, your application server, and your database server.

Configuring JasperReports Server

To specify a different character encoding

Procedure

1. Open the .../WEB-INF/applicationContext.xml file and locate the following bean. It is configured for UTF-8:

```
<bean id="encodingProvider"
        class="com.jaspersoft.jasperserver.api.common.util.
        StaticCharacterEncodingProvider">
    <constructor-arg value="UTF-8" />
</bean>
```

2. Change "UTF-8" to the encoding type your database server and application server use. For example:

```
<bean id="encodingProvider"
        class="com.jaspersoft.jasperserver.api.common.util.
        StaticCharacterEncodingProvider">
    <constructor-arg value="UTF-16" />
</bean>
```

3. Save the file.

4. Restart JasperReports Server.

Configuring the Application Server and Database Server

If you want to use a character encoding other than UTF-8, you may need to configure the third-party software that JasperReports Server relies on. For more information, refer to the

documentation for your application server and database server. For Tomcat, you can specify a different character encoding by following steps similar to those described in Tomcat and UTF-8 Configuration.

Configuration for Localized Analysis Schemas

If you plan to use localized OLAP views, you must take additional steps to configure JasperReports Server.

To configure JasperReports Server for localized OLAP views

Procedure

1. Every Unicode database that JasperReports Server interacts with (whether it is the repository database or a database accessed through a data source defined in JasperReports Server) must be created to support UTF-8. For example, to create the Foodmart database in PostgreSQL, you might give a command similar to the following:

```
create database foodmart_ja encoding='utf8';
```

2. The URL of any OLAP data source that JasperReports Server accesses must be properly configured in the .../META-INF/context.xml file. For example, the URL definition for the Foodmart sample database might be similar to the following:

```
<Resource name="jdbc/MondrianFoodMart_ja"
    auth="Container" type="javax.sql.DataSource"
    maxActive="100" maxIdle="30" maxWait="10000"
    username="postgres" password="postgres" driverClassName="org.postgresql.Driver"
    url="jdbc:postgresql://localhost:5432/foodmart_ja" />
```

3. Encoding options must be added to the JDBC connection string for any data source that points to an OLAP database. For example, when creating a data source in JasperReports Server that points to an OLAP database, use the following connection string:

```
jdbc:postgresql://localhost:5432/foodmart_ja
```

Creating a Locale

When you want to create other locales for JasperReports Server, translation is only one aspect of localization. Creating a locale includes these tasks:

- Translating labels and messages

- Creating a Resource Bundle

- Setting Date and Datetime Formats

- Setting Data Format Masks

The tasks in this section require you to edit these files:

File	Location	Purpose of Edits
*.properties files	.../WEB-INF/bundles <js-install>/buildomatic/conf_ source/iePro/bundles	Translating labels and messages
jasperserver_ config.properties	.../WEB-INF/bundles	Changing date formats
adhoc_masks	.../WEB-INF/bundles	Changing format masks

About Properties Files

Resource bundles for JasperReports Server and Jaspersoft OLAP are Java properties files found in the .../WEB-INF/bundles directory. The properties files contain all the labels and messages used in JasperReports Server and Jaspersoft OLAP.

A bundle includes a default locale (for example, jasperserver_messages.properties), which is written in U.S. English. Then it consists of all the properties files with the same base name, but different locale (such as jasperserver_messages_fr.properties). Each file translates all of the strings of the default file into the language given by the locale. The Java programming language has rules for specifying locales and alternate locales and determining which locale in the bundle to use.

Default Resource Bundles in JasperReports Server	
File in …/WEB-INF/bundles	**Description**
accessibility_ messages.properties	Text spoken in accessibility software such as screen readers.
AdHocFiltersBundle.properties	Labels and messages for the Ad Hoc Filters panel.
adhoc_masks.properties	Masks (data formats) for values appearing in the Ad Hoc Editor.
adhoc_messages.properties	Labels and messages for the Ad Hoc Editor.
AttributeBundle.properties	Messages for attribute validation.
AttributeBundles.properties	Messages for attribute dialogs.
calendar.properties	Labels and messages used by the pop-up calendar dialog.
.properties	.
CommonBundle.properties	Bundle for tests.
createsldatasource_messages .properties	Labels and messages for the Create Domain dialog.
domain_designer_messages .properties	Labels and messages for the Domain Designer UI.
HomeBundle.properties	Labels for the home page.
image_descriptions_messages .properties	Labels and messages for the AWS (Amazon Web Services) machine images.
jasperreports_highcharts_ messages.properties	Messages for charts in the report viewer.
jasperreports_messages	Labels and messages for the report viewer.

Default Resource Bundles in JasperReports Server	
File in .../WEB-INF/bundles	**Description**
.properties	
jasperserver_config.properties	Configuration properties for date formats and email templates.
jasperserver_ messages.properties	Labels and messages used in the main JasperReports Server user interface.
jsexceptions_ messages.properties	Messages used in errors and exceptions, both in the UI and in the log messages.
LicenseMessages.properties	Labels and messages used when validating licenses.
logger_descriptions.properties logger_descriptions_ pro.properties	Internal log messages.
pro_nav_messages.properties	Labels and messages for the menu bar and Home page.
querybuilder_ messages.properties	Labels and messages for the Choose Data dialog (for creating a Domain Topic before using Ad Hoc Editor).
report_option_messages .properties	Labels and messages for the report options dialog.
ScalableInputControlsBundle .properties	Labels for lists of values in input controls.
scheduling_ws.properties	Validation messages for the report scheduler.
security.properties	Messages used by input validation.
semanticLayer.properties	Labels and messages for the Domain designer and Ad Hoc data policies.
ValidationBundle.properties	Messages for dashboard dialogs.

Default Resource Bundles in Jaspersoft OLAP	
File in .../WEB-INF/bundles	**Description**
ja_mondrian.properties	Labels and messages in the OLAP settings UI.
ja-pro_messages.properties	Labels and messages in the commercial edition OLAP viewer.
jpivot_messages.properties	Labels and messages in the community edition OLAP viewer.
mondrian_exception_ messages .properties	MDX validation error messages specific to the internal analysis engine.

The standalone import and export tools have their own bundles located outside of the web app. The following bundles are located in the WAR file distribution package, under the buildomatic folder.

Default Resource Bundles for js-export.sh and js-import.sh	
File in buildomatic/conf_ source/iePro/bundles	**Description**
ji-export-messages.properties	Labels and messages for js-export --help.
multi-tenancy-export- messages.properties	Labels and messages for export in commercial editions.
multi-tenancy-import- messages.properties	Labels and messages export in commercial editions.
jsexceptions_ messages.properties	Messages used in errors and exceptions, both in the UI and in the log messages.

Creating a Resource Bundle

Create a resource bundle by making a copy of each *.properties file, using the following syntax for the copy's file name:

```
<default_file_name>_<locale>.properties
```

where

<default_file_name> is the name of the default version of the properties file, and

<locale> is a Java-compliant locale identifier.

For example, consider the core JasperReports Server resource bundle. For various locales, it might be named as follows:

Language	File Name
Default (English)	jasperserver_messages.properties
French	jasperserver_messages_fr.properties
French in Switzerland	jasperserver_messages_fr_CH.properties

For a list of Java-compliant locales, please refer to the Java documentation.

> The resource bundles described in this document consist of locale-specific Java properties files. Java properties files use the ISO-8859-1 (Latin-1) encoding that is the same as ASCII for all English non-accented characters. For international characters that are not in ISO-8859-1, use Unicode escape sequences (for example \u00e9 is é).

To create a new JasperReports Server resource bundle

Procedure

1. Copy each of the properties files (keeping them in the same directory as the originals) and rename them according to your locale.

2. Translate each *.properties file's labels and messages into the new language. Some of the strings in the properties files are date formats and format masks that may need to be edited for the new locale. For more information, refer to Setting Date and Datetime Formats.

3. Save the files.

4. If the new locale requires specific character encoding or fonts, ensure that JasperReports Server and the third party software it relies on are configured to

support them. For more information, refer to Configuring JasperReports Server for Multibyte Fonts.

> The new locale is not available in JasperReports Server until you follow the steps described in Specifying Additional Locales.

Setting Date and Datetime Formats

Each locale may have its own rules for displaying dates and datetime values. System date and datetime formatting is controlled by four patterns that are specified in the jasperserver_config_<locale>.properties file associated with a particular locale.

For example in the English resource bundle, the four entries are:

```
date.format=dd-MM-yyyy
datetime.format=dd-MM-yyyy HH:mm
calendar.date.format=%d-%m-%Y
calendar.datetime.format=%d-%m-%Y %H:%M
```

The first two keys are used to parse and format dates and datetime values using an internal java.util.DateFormat object across the whole application. These patterns should be non-localized date patterns, in accordance with the Java Development Kit (JDK) syntax.

The other two keys are used by the calendar control, which formats the user-selected date and datetime values in accordance with its own pattern syntax.

To change the system date and datetime formatting for a new locale, edit the strings specified by these keys.

For some locales, you can use a different (non-default) Date/Time format. However, this format is not applied for Input Controls, because the date format for Input Controls is a constant and is set to ISO date/time format (your date and your calendar settings are ignored for Input Controls).

If your existing configurations for Input Controls and your locales are working fine, then no changes are needed and JasperReports Server will proceed sending or getting date/datetime in ISO format as before.

If you prefer to not have Input Controls operating only in ISO date format (for example, when Input Controls are used in Visualize.js implementation), you can enable the new provider added in this release and re-save ad-hoc views/reports if needed.

To use the date/format from jasperserver_config.properties for Input Controls, apart from setting formats found in the config.properties file, perform the following steps:

Procedure

1. Update the "jasperserver_config.properties" file.

2. Update the **/WEB-INF/js.spring.properties** file:

   a. Comment out
      `bean.calendarFormatProvider=isoCalendarFormatProvider`
      line.

   b. Uncomment
      `bean.calendarFormatProvider=messagesCalendarFormatProvider` line.

Setting Data Format Masks

Each locale may have its own format masks that determine how numbers appear in the Ad Hoc Editor. To make the data format masks vary by locale, you must create an adhoc_masks file for the new locale. To do so, copy the file adhoc_masks.properties to a new name that specifies the new locale and change the masks defined in the new file. For example, the French file would be named adhoc_masks_fr.properties.

> The data format masks described in this section are used in the Domains and in the Ad Hoc Editor; they appear in Ad Hoc views as well as JRXML reports based on Domains. They're not applicable to Jaspersoft OLAP.

Customize the available data format masks for dates, integers, and decimals by editing the existing masking entries or adding new ones. The default entries are given in the following table:

Data Format Mask Properties	Appearance in en_US Locale
| ```
ADH_100_MASK_date_0 = short,hide
ADH_100_MASK_date_1 = long,hide
ADH_100_MASK_date_2 = short,medium
ADH_100_MASK_date_3 = medium,medium
``` | 3/31/09<br>Mar 31, 2009<br>March 31, 2009<br>Mar 31, 2009 23:59:59 |
| ```
ADH_100_MASK_int_0 = #,##0
ADH_100_MASK_int_1 = 0
ADH_100_MASK_int_2 = $#,##0;($#,##0)
ADH_100_MASK_int_3 = #,##0;(#,##0)
``` | -1,234<br>-1234<br>($1,234)<br>(1234) |
| ```
ADH_100_MASK_dec_0 = #,##0.00
ADH_100_MASK_dec_1 = 0
ADH_100_MASK_dec_2 = $#,##0.00;($#,##0.00)
ADH_100_MASK_dec_3 = $#,##0;($#,##0)
``` | -1,234.56<br>-1234<br>($1,234.56)<br>($1,234) |

The data format masks for each type are numbered consecutively from zero; create new masks by adding new entries. The keys of the new entries must follow the convention established in the default entries. For example, a new decimal data format mask might have this ID:

```
ADH_100_MASK_dec_4
```

Date format masks are implemented using java.text.SimpleDateFormat and JasperReports extensions that provide access to predefined localized data format masks. New datetime masks must be specified in one of the following formats:

- A style for the date part of the value and a style for the time part (separated by comma) or a single style for both parts. A style is one of Short, Medium, Long, Full, Default (which correspond to java.text.DateFormat styles) and Hide.

- A pattern that can be supplied to java.text.SimpleDateFormat. In this case, internationalization support is limited.

Both integer and decimal data format masks are implemented with java.text.DecimalFormat, which localizes characters in the format specification. For

example, consider the case of the digit grouping symbol (thousands separator): in French, it is a space; in U.S. English, it is a comma. DecimalFormat handles both cases: if the number pattern #,##0 is used, the number 6000 appears as 6 000 in the French locale and as 6,000 in the U.S. English locale.

For more information about Java's handling of decimal and date format masks, see:

- http://download.oracle.com/javase/6/docs/api/java/text/DecimalFormat.html

- http://download.oracle.com/javase/6/docs/api/java/text/DateFormat.html

> By default, monetary values in Ad Hoc views are masked as USD (United States Dollars). Depending on your data, you may need to support a different currency, support more than one currency, or support currency conversion. These are three very different cases:
>
> - Supporting a different currency than USD involves changing the monetary masks to use the correct symbol for your currency (for example, replace the $ symbol in the ADH_100_ MASK_dec_2 and ADH_100_MASK_dec_3 masks). However, changing this symbol does not actually convert currencies in your reports.
>
> - Supporting other currencies in addition to USD involves adding new masks. However, adding data formats does not actually convert currencies in your reports.
>
> - Supporting currency conversion is more complicated; you must consider such issues as fluctuations in conversion rates. Oftentimes, a third-party service can be used to perform currency conversion

Configuring JasperReports Server to Offer a Locale

After creating a locale, you must configure JasperReports Server to offer it to your users, along with any new time zones.

The tasks in this section require you to edit these files:

File Name	Location	Purpose of Edits
applicationContext-security.xml	WEB-INF	Specifying additional locales
jasperserver-servlet.xml	WEB-INF	Specifying additional time zones

Specifying Additional Locales

By default, JasperReports Server appears in the locale selected in the end user's browser. The Login page allows users to specify the locale they want to use. The list of available locales is defined in applicationContext-security.xml. Edit this file to add a new locale.

To add a new locale

Procedure

1. Edit the applicationContext-security.xml file and locate the bean named userLocalesList. For example:

```
<bean id="userLocalesList"
      class="com.jaspersoft.jasperserver.war.common.LocalesListImpl">
    <property name="locales">
        <list>
            <value type="java.util.Locale">en</value>
            <value type="java.util.Locale">fr</value>
            <value type="java.util.Locale">it</value>
            <value type="java.util.Locale">de</value>
            <value type="java.util.Locale">ro</value>
            <value type="java.util.Locale">ja</value>
            <value type="java.util.Locale">zh_TW</value>
        </list>
    </property>
</bean>
```

2. Add the new locale to the end of the list. For example, add the following line for Dutch (Java's nl_NL locale):

```
<value type="java.util.Locale">nl_NL/value>
```

3. Save the file.

4. Restart JasperReports Server, and log into the web application to test your translation. Reviewing the translated strings in context can help you improve your word choices.

For a list of Java-compliant locales, please refer to the Java documentation.

Specifying Additional Time Zones

By default, JasperReports Server assumes the user's time zone is that of the JasperReports Server host. The Login page allows users to choose a different time zone. The available list is defined in applicationContext.xml file.

To add a time zone

Procedure

1. Open the applicationContext.xml file and locate the userTimeZonesList bean. For example:

```
<bean id="userTimeZonesList"
        class="com.jaspersoft.jasperserver.war.common.JdkTimeZonesList">
    <property name="timeZonesIds">
        <list>
            <value>America/Los_Angeles</value>
            <value>America/Denver</value>
            <value>America/Chicago</value>
            <value>America/New_York</value>
            <value>Europe/London</value>
            <value>Europe/Berlin</value>
            <value>Europe/Bucharest</value>
        </list>
    </property>
</bean>
```

2. Add the new time zone to the bottom of the list. Specify each time zone as the standard Java time zone values so that JasperReports Server adjusts for daylight savings time when appropriate. For example, add the following line for Tokyo:

```
            <value>Asia/Tokyo</value>
```

3. Save the file.

4. Restart JasperReports Server.

For more information about Java-complaint time zones, please refer to the Java documentation.

Setting a Default Time Zone

If you want JasperReports Server to use a time zone other than the host computer's, you can set a specific time zone in Java. It becomes the default time zone for all users, but they can still select a different time zone when they log in.

To set a default time zone, set the user.timezone property in the JVM as shown in the tables below. Locate the file containing JVM settings for your platform and application server. The value for the property must be a Java-compliant time zone, for example, Europe/Bucharest.

You must restart your application server for this setting to take effect. The time zone is set for all applications in your application server, including JasperReports Server.

JVM Settings for Default Time Zone			
Operating System	**App Server**	**File**	**Setting**
Windows	Tomcat	<apache-tomcat>\bin\setenv.bat	Add this line of code: set JAVA_OPTS=%JAVA_OPTS% - Duser.timezone=<timezone>
	JBoss	<jboss>\bin\standalone.conf.bat	
Linux	Tomcat	<apache-tomcat>/bin/setenv.sh	Add this line of code: export JAVA_OPTS="$JAVA_OPTS - Duser.timezone=<timezone>"
	JBoss	<jboss>/bin/standalone.conf	

Glossary

Ad Hoc Editor

The interactive data explorer in JasperReports Server Professional and Enterprise editions. Starting from a predefined collection of fields, the Ad Hoc Editor lets you drag and drop fields, dimensions, and measures to explore data and create tables, charts, and crosstabs. These Ad Hoc views can be saved as reports.

Ad Hoc Report

In previous versions of JasperReports Server, a report created through the Ad Hoc Editor. Such reports could be added to dashboards and be scheduled, but when edited in Jaspersoft Studio, lost their grouping and sorting. In the current version, the Ad Hoc Editor is used to explore views which in turn can be saved as reports. Such reports can be edited in Jaspersoft Studio without loss, and can be scheduled and added to dashboards.

Ad Hoc View

A view of data that is based on a Domain, Topic, or OLAP client connection. An Ad Hoc view can be a table, chart, or crosstab and is the entry point to analysis operations such as slice and dice, drill down, and drill through. OLAP View You can save an Ad Hoc view as a report in order to edit it in the interactive viewer, schedule it, or add it to a dashboard.

Aggregate Function

An aggregate function is one that is computed using a group of values; for example, Sum or Average. Aggregate functions can be used to create calculated fields in Ad Hoc views. Calculated fields containing aggregate functions cannot be used as fields or added to groups in an Ad Hoc view and should not be used as filters. Aggregate functions allow you to set a level, which specifies the scope of the calculation; level values include Current (not available for PercentOf), ColumnGroup, ColumnTotal, RowGroup, RowTotal, Total.

Amazon Web Services (AWS)

Cloud platform, used to provide and host a family of services, such as RDS, S3, and EC2.

Analysis View

OLAP View.

Audit Archiving

To prevent audit logs from growing too large to be easily accessed, the installer configures JasperReports Server to move current audit logs to an archive after a certain number of days, and to delete logs in the archive after a certain age. The archive is another table in the JasperReports Server's repository database.

Audit Domains

A Domain that accesses audit data in the repository and lets administrators create Ad Hoc reports of server activity. There is one Domain for current audit logs and one for archived logs.

Audit Logging

When auditing is enabled, audit logging is the active recording of who used JasperReports Server to do what when. The system installer can configure what activities to log, the amount of detail gathered, and when to archive the data. Audit logs are stored in the same private database that JasperReports Server uses to store the repository, but the data is only accessible through the audit Domains.

Auditing

A feature of JasperReports Server Enterprise edition that records all server activity and allows administrators to view the data.

Calculated Field

In an Ad Hoc view or a Domain, a field whose value is calculated from a user-defined formula that may include any number of fields, operators, and constants. For Domains, a calculated field becomes one of the items to which the Domain's security file and locale bundles can apply. There are more functions available for Ad Hoc view calculations than for Domains.

CloudFormation (CF)

Amazon Web Services CloudFormation gives developers and systems administrators an easy way to create and manage a collection of related AWS resources, provisioning, and updating them in an orderly and predictable fashion.

CRM

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

CrossJoin

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

457 | Glossary

Cube

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

Custom Field

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

Dashboard

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

Dashlet

An element in a dashboard. Dashlets are defined by editable properties that vary depending on the dashlet type. Types of dashlet include reports, text elements, filters, and external web content.

Data Island

A single join tree or a table without joins in a Domain. A Domain may contain several data islands, but when creating an Ad Hoc view from a Domain, you can only select one of them to be available in the view.

Data Policy

In JasperReports Server, a setting that determines how the server processes and caches data used by Ad Hoc reports. Select your data policies by clicking Manage > Server > Settings Ad Hoc Settings. By default, this setting is only available to the superuser account.

Data Source

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

JasperReports® Server Administrator Guide

Dataset

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

Datatype

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

Denormalize

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

Derived Table

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

Dice

An OLAP operation to select columns.

Dimension

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

Domain

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

Domain Topic

A Topic that is created from a Domain by the Data Chooser. A Domain Topic is based on the data source and items in a Domain, but it allows further filtering, user input, and

selection of items. Unlike a JRXML-based Topic, a Domain Topic can be edited in JasperReports Server by users with the appropriate permissions.

Drill

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

- Drill down. An OLAP operation that exposes more detailed information down the hierarchy levels by delving deeper into the hierarchy and updating the contents of the navigation table.

- Drill through. An OLAP operation that displays detailed transactional data for a given aggregate measure. Click a fact to open a new table beneath the main navigation table; the new table displays the low-level data that constitutes the data that was clicked.

- Drill up. An OLAP operation for returning the parent hierarchy level to view to summary information.

Eclipse

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

ETL

Extract, Transform, Load. A process that retrieves data from transactional systems, and filters and aggregates the data to create a multidimensional database. Generally, ETL prepares the database that your reports will access. The Jaspersoft ETL product lets you define and schedule ETL processes.

Fact

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

Field

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

Frame

In Jaspersoft Studio, a frame is a rectangular element that can contain other elements and optionally draw a border around them. Elements inside a frame are positioned relative to the frame, not to the band, and when you move a frame, all the elements contained in the frame move together. A frame automatically stretches to fit its contents.

Group

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

- In a table, the value appears in a header and footer around the rows of the group, while the other fields appear as columns.

- In a chart, the field chosen to define the group becomes the independent variable on the X axis, while the other fields of each group are used to compute the dependent value on the Y axis.

Hierarchy Level

In an OLAP cube, a member of a dimension containing a group of members.

Input Control

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

Item

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

JasperReport

A combination of a report template and data that produces a complex document for viewing, printing, or archiving information. In the server, a JasperReport references other resources in the repository:

- The report template (in the form of a JRXML file)

- Information about the data source that supplies data for the report

- Any additional resources, such as images, fonts, and resource bundles referenced by the report template.

The collection of all the resources that are referenced in a JasperReport is sometimes called a report unit. End users usually see and interact with a JasperReport as a single resource in the repository, but report creators must define all of the components in the report unit.

JasperReports IO

An HTTP-based reporting service for JasperReports Library that provides a REST API for running, exporting, and interacting with reports and a JavaScript API for embedding reports and their input controls into your web pages and web applications.

JasperReports Library

An embeddable, open source, Java API for generating a report, filling it with current data, drawing charts and tables, and exporting to any standard format (HTML, PDF, Excel, CSV, and others). JasperReports processes reports defined in JRXML, an open XML format that allows the report to contain expressions and logic to control report output based on run-time data.

JasperReports Server

A commercial open source, server-based application that calls the JasperReports Library to generate and share reports securely. JasperReports Server authenticates users and lets them upload, run, view, schedule, and send reports from a web browser. Commercial versions provide metadata layers, interactive report and dashboard creation, and enterprise features such as organizations and auditing.

Jaspersoft Studio

A commercial open source tool for graphically designing reports that leverage all features of the JasperReports Library. Jaspersoft Studio lets you drag and drop fields, charts, and sub-reports onto a canvas, and also define parameters or expressions for each object to create pixel-perfect reports. You can generate the JRXML of the report directly in Jaspersoft Studio, or upload it to JasperReports Server. Jaspersoft Studio is implemented in Eclipse.

Jaspersoft ETL

A graphical tool for designing and implementing your data extraction, transforming, and loading (ETL) tasks. It provides hundreds of data source connectors to extract data from many relational and non-relational systems. Then, it schedules and performs data aggregation and integration into data marts or data warehouses that you use for reporting.

Jaspersoft OLAP

A relational OLAP server integrated into JasperReports Server that performs data analysis with MDX queries. The product includes query builders and visualization clients that help users explore and make sense of multidimensional data. Jaspersoft OLAP also supports XML/A connections to remote servers.

Jaspersoft Studio

An open source tool for graphically designing reports that leverage all features of the JasperReports Library. Jaspersoft Studio lets you drag and drop fields, charts, and sub-

reports onto a canvas, and also define parameters or expressions for each object to create pixel-perfect reports. You can generate the JRXML of the report directly in Jaspersoft Studio, or upload it to JasperReports Server. Jaspersoft Studio is implemented in Eclipse.

JavaBean

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

JDBC

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

JNDI

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

Join Tree

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

JPivot

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

JRXML

An XML file format for saving and sharing reports created for the JasperReports Library and the applications that use it, such as Jaspersoft Studio and JasperReports Server. JRXML is an open format that uses the XML standard to define precisely all the structure and configuration of a report.

Level

Specifies the scope of an aggregate function in an Ad Hoc view. Level values include Current (not available for PercentOf), ColumnGroup, ColumnTotal, RowGroup, RowTotal, Total.

MDX

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

Measure

Depending on the context:

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

- In an OLAP view, a formula that calculates the facts that constitute the quantitative data in a cube.

Mondrian

A Java-based, open source multidimensional database application.

Mondrian Connection

An OLAP client connection that consists of an OLAP schema and a data source. OLAP client connections populate OLAP views.

Mondrian Schema Editor

An open source Eclipse plug-in for creating Mondrian OLAP schemas.

Mondrian XML/A Source

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

MySQL

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

Navigation Table

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

ODBO Connect

Jaspersoft ODBO Connect enables Microsoft Excel 2003 and 2007 Pivot Tables to work with Jaspersoft OLAP and other OLAP servers that support the XML/A protocol. After setting up the Jaspersoft ODBO data source, business analysts can use Excel Pivot Tables as a front-end for OLAP analysis.

OLAP

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

OLAP Client Connection

A definition for retrieving data to populate an OLAP view. An OLAP client connection is either a direct Java connection (Mondrian connection) or an XML-based API connection (XML/A connection).

OLAP Schema

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

OLAP View

Also called an analysis view. A view of multidimensional data that is based on an OLAP client connection and an MDX query. Unlike Ad Hoc views, you can directly edit an OLAP view's MDX query to change the data and the way they are displayed. An OLAP view is the entry point for advanced analysis users who want to write their own queries. Ad Hoc View

Organization

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

Organization Admin

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

Outlier

A fact that seems incongruous when compared to other member's facts. For example, a very low sales figure or a very high number of help desk tickets. Such outliers may indicate a problem (or an important achievement) in your business. The analysis features of Jaspersoft OLAP excel at revealing outliers.

Parameter

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

Pivot

To rotate a crosstab such that its row groups become column groups and its column

groups become rows. In the Ad Hoc Editor, pivot a crosstab by clicking ⤵ .

Pivot Table

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

Properties

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

Report

In casual usage, report may refer to:

- A JasperReport. JasperReport

- The main JRXML in a JasperReport.

- The file generated when a JasperReport is scheduled. Such files are also called content resources or output files.

- The file generated when a JasperReport is run and then exported.

- In previous JasperReports Server versions, a report created in the Ad Hoc Editor. Ad Hoc Report

Report Run

An execution of a report, Ad Hoc view, or dashboard, or a view or dashboard designer session, it measures and limits usage of Freemium instances of JasperReports Server. The executions apply to resources no matter how they are run (either in the web interface or through the various APIs, such as REST web services). Users of our Community Project and our full-use commercial licenses are not affected by the limit. For more information, please contact sales@jaspersoft.com.

Repository

Depending on the context:

- In JasperReports Server, the repository is the tree structure of folders that contain all saved reports, dashboards, OLAP views, and resources. Users access the repository through the JasperReports Server web interface or through Jaspersoft Studio. Applications can access the repository through the web service API. Administrators use the import and export utilities to back up the repository contents.

- In JasperReports IO, the repository is where all the resources needed to create and run reports are stored. The repository can be stored in a directory on the host computer or in an S3 bucket hosted by Amazon Web Services. Users access the repository through a file browser on the host machine or through the AWS console.

Resource

In JasperReports Server, anything residing in the repository, such as an image, file, font, data source, Topic, Domain, report element, saved report, report output, dashboard, or OLAP view. Resources also include the folders in the repository. Administrators set user and role-based access permissions on repository resources to establish a security policy.

Role

A security feature of JasperReports Server. Administrators create named roles, assign them to user accounts, and then set access permissions to repository objects based on those roles. Certain roles also determine what functionality and menu options are displayed to users in the JasperReports Server interface.

S3 Bucket

Cloud storage system for Amazon Web Services. JasperReports IO can use an S3 bucket to store files for its repository.

Schema

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

Schema Workbench

A graphical tool for easily designing OLAP schemas, data security schemas, and MDX queries. The resulting cube and query definitions can then be used in Jaspersoft OLAP to perform simple but powerful analysis of large quantities of multi-dimensional data stored in standard RDBMS systems.

Set

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

Slice

An OLAP operation for filtering data rows.

SQL

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

Stack

A collection of Amazon Web Services resources you create and delete as a single unit.

System Admin

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

Topic

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

Transactional Data

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

User

Depending on the context:

- A person who interacts with JasperReports Server through the web interface. There are generally three categories of users: administrators who install and configure JasperReports Server, database experts or business analysts who create data sources and Domains, and business users who create and view reports and dashboards.

- A user account that has an ID and password to enforce authentication. Both people and API calls accessing the server must provide the ID and password of a valid user account. Roles are assigned to user accounts to determine access to objects in the repository.

View

Several meanings pertain to JasperReports Server:

- An Ad Hoc view. Ad Hoc View

- An OLAP view. OLAP View

- A database view. See http://en.wikipedia.org/wiki/View_%28database%29.

Virtual Data Source

A virtual data source allows you to combine data residing in multiple JDBC and/or JNDI data sources into a single data source that can query the combined data. Once you have created a virtual data source, you create Domains that join tables across the data sources to define the relationships between the data sources.

WCF

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

Web Services

A SOAP (Simple Object Access Protocol) API that enables applications to access certain features of JasperReports Server. The features include repository, scheduling and user administration tasks.

XML

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

XML/A

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

XML/A Connection

A type of OLAP client connection that consists of Simple Object Access Protocol (SOAP) definitions used to access data on a remote server. OLAP client connections populate OLAP views.

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