## Table of Contents

### Chapter 1 Overview of Server Administration

1. **Overview of Organizations** .................................................. 10
   1.1 **Single Default Organization** .......................................... 10
   1.2 **Multiple Organizations** ................................................ 10
   1.3 **Levels of Administrators** .............................................. 10

2. **Overview of the Repository** .................................................. 11
   2.1 **Folder Structure** .......................................................... 11
   2.2 **Resources** ..................................................................... 12
   2.3 **Sample Data** .................................................................. 12
   2.4 **Browsing and Searching** ............................................... 13

3. **Overview of Users and Roles** ................................................ 14
   3.1 **Administering Users and Roles** ....................................... 14
   3.2 **Delegated Administration** .............................................. 14

4. **Security Overview** ................................................................. 15
   4.1 **Authentication** ............................................................... 15
   4.2 **Authorization Overview** ................................................ 16

5. **Administrator Login** ............................................................... 17
   5.1 **JasperReports Server Heartbeat** ..................................... 17
   5.2 **Administrator Email** ..................................................... 17

### Chapter 2 Organization, User, and Role Management

6. **Managing Organizations** ....................................................... 19
   6.1 **Viewing Organization Properties** .................................... 20
   6.2 **Creating an Organization** ............................................... 21
   6.3 **Default Folders for Organizations** ................................... 21
   6.4 **Editing an Organization** .............................................. 22
   6.5 **Deleting an Organization** ............................................. 22

7. **Managing Users** ................................................................. 23
   7.1 **Viewing User Properties** ............................................... 23
<table>
<thead>
<tr>
<th>Chapter 5</th>
<th>Themes</th>
<th>75</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>75</td>
</tr>
<tr>
<td>5.2</td>
<td>How Themes Work</td>
<td>76</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Theme Files</td>
<td>78</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Inheritance Mechanism</td>
<td>79</td>
</tr>
<tr>
<td>5.2.3</td>
<td>CSS Priority Scheme and Custom Overrides</td>
<td>79</td>
</tr>
<tr>
<td>5.3</td>
<td>Administering Themes</td>
<td>80</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Setting the System Theme</td>
<td>80</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Setting an Organization Theme</td>
<td>81</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Restricting Access to Themes</td>
<td>82</td>
</tr>
<tr>
<td>5.3.4</td>
<td>Creating Theme Folders and Files</td>
<td>83</td>
</tr>
<tr>
<td>5.4</td>
<td>Working With CSS Files</td>
<td>85</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Theme Development Workflow</td>
<td>86</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Firebug Plug-in for Firefox</td>
<td>86</td>
</tr>
<tr>
<td>5.4.3</td>
<td>Test Platform</td>
<td>86</td>
</tr>
<tr>
<td>5.4.4</td>
<td>Modifying the Appearance of Jaspersoft OLAP</td>
<td>87</td>
</tr>
<tr>
<td>5.4.5</td>
<td>User Interface Samples</td>
<td>87</td>
</tr>
<tr>
<td>Chapter 6</td>
<td>Import/Export</td>
<td>89</td>
</tr>
<tr>
<td>6.1</td>
<td>Running Import and Export</td>
<td>89</td>
</tr>
<tr>
<td>6.2</td>
<td>Exporting Repository Resources</td>
<td>90</td>
</tr>
<tr>
<td>6.3</td>
<td>Importing Repository Resources</td>
<td>91</td>
</tr>
<tr>
<td>Chapter 7</td>
<td>System Configuration</td>
<td>93</td>
</tr>
<tr>
<td>7.1</td>
<td>Security Framework Configuration</td>
<td>94</td>
</tr>
<tr>
<td>7.2</td>
<td>Configuring Password Options</td>
<td>94</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Enabling Password Expiration</td>
<td>94</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Allowing Users to Change their Passwords</td>
<td>95</td>
</tr>
<tr>
<td>7.3</td>
<td>Configuring the User Session Timeout</td>
<td>95</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>7.4</td>
<td>Encrypting the Repository Database Password</td>
<td>96</td>
</tr>
<tr>
<td>7.5</td>
<td>Configuring Ad Hoc</td>
<td>96</td>
</tr>
<tr>
<td>7.5.1</td>
<td>Ad Hoc Settings</td>
<td>96</td>
</tr>
<tr>
<td>7.5.2</td>
<td>Ad Hoc Editor Configuration File Options</td>
<td>99</td>
</tr>
<tr>
<td>7.5.3</td>
<td>Managing the Ad Hoc Cache</td>
<td>99</td>
</tr>
<tr>
<td>7.5.4</td>
<td>Configuring Ad Hoc OLAP Filters</td>
<td>103</td>
</tr>
<tr>
<td>7.6</td>
<td>Configuring System Logs</td>
<td>103</td>
</tr>
<tr>
<td>7.6.1</td>
<td>Managing Log Settings</td>
<td>103</td>
</tr>
<tr>
<td>7.6.2</td>
<td>Log Configuration Files</td>
<td>106</td>
</tr>
<tr>
<td>7.7</td>
<td>Disabling the Domain Validation Check</td>
<td>107</td>
</tr>
<tr>
<td>7.8</td>
<td>Configuring JasperReports Library</td>
<td>107</td>
</tr>
<tr>
<td>7.8.1</td>
<td>Extending JasperReports Library</td>
<td>107</td>
</tr>
<tr>
<td>7.8.2</td>
<td>Changing the Crosstab Limit</td>
<td>108</td>
</tr>
<tr>
<td>7.8.3</td>
<td>Setting a Global Chart Theme</td>
<td>108</td>
</tr>
<tr>
<td>7.8.4</td>
<td>Enabling the XHTML or HTML Exporter</td>
<td>109</td>
</tr>
<tr>
<td>7.8.5</td>
<td>Enabling Flash or HTML5 for Pro Charts</td>
<td>109</td>
</tr>
<tr>
<td>7.9</td>
<td>Configuring the Heartbeat</td>
<td>110</td>
</tr>
<tr>
<td>7.10</td>
<td>Removing Report Scheduling Interval Options</td>
<td>110</td>
</tr>
<tr>
<td>7.11</td>
<td>Special Domain Support</td>
<td>111</td>
</tr>
<tr>
<td>7.11.1</td>
<td>Enabling Oracle Synonyms</td>
<td>111</td>
</tr>
<tr>
<td>7.11.2</td>
<td>Enabling CLOB Fields</td>
<td>112</td>
</tr>
<tr>
<td>7.11.3</td>
<td>Enabling Proprietary Types</td>
<td>113</td>
</tr>
<tr>
<td>7.12</td>
<td>Configuring the Online Help</td>
<td>113</td>
</tr>
<tr>
<td><strong>Chapter 8</strong></td>
<td><strong>Auditing</strong></td>
<td><strong>115</strong></td>
</tr>
<tr>
<td>8.1</td>
<td>Audited Events</td>
<td>115</td>
</tr>
<tr>
<td>8.2</td>
<td>Configuring Auditing</td>
<td>116</td>
</tr>
<tr>
<td>8.2.1</td>
<td>Enabling Auditing</td>
<td>116</td>
</tr>
<tr>
<td>8.2.2</td>
<td>Archive Options</td>
<td>117</td>
</tr>
<tr>
<td>8.2.3</td>
<td>Events and Properties</td>
<td>117</td>
</tr>
<tr>
<td>8.3</td>
<td>Using the Audit Domains</td>
<td>117</td>
</tr>
<tr>
<td>8.3.1</td>
<td>Domain Items</td>
<td>118</td>
</tr>
<tr>
<td>8.3.2</td>
<td>Sample Reports</td>
<td>119</td>
</tr>
<tr>
<td>8.4</td>
<td>Importing and Exporting Audit Data</td>
<td>119</td>
</tr>
<tr>
<td><strong>Appendix A</strong></td>
<td><strong>Troubleshooting</strong></td>
<td><strong>121</strong></td>
</tr>
<tr>
<td>A.1</td>
<td>Number of Users Exceeded</td>
<td>121</td>
</tr>
<tr>
<td>A.2</td>
<td>Field Names Disappear in Ad Hoc Editor</td>
<td>121</td>
</tr>
<tr>
<td>A.3</td>
<td>Scheduler Sending Multiple Emails</td>
<td>122</td>
</tr>
<tr>
<td>A.4</td>
<td>Adding Data Sources</td>
<td>122</td>
</tr>
<tr>
<td>A.4.1</td>
<td>JDBC Driver in Classpath</td>
<td>122</td>
</tr>
<tr>
<td>A.4.2</td>
<td>JDBC Driver Name</td>
<td>122</td>
</tr>
<tr>
<td>A.4.3</td>
<td>Database Permissions</td>
<td>123</td>
</tr>
<tr>
<td>A.4.4</td>
<td>JDBC Database URL</td>
<td>123</td>
</tr>
<tr>
<td>A.4.5</td>
<td>JNDI Services on Apache Tomcat</td>
<td>123</td>
</tr>
</tbody>
</table>
# A.4.6 JNDI Services on JBoss

- A.4.6 JNDI Services on JBoss ................................................................. 124

## Appendix B Integrating With Liferay Portal .................................................. 125

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.1 Changing Liferay's Port Numbers</td>
<td>125</td>
</tr>
<tr>
<td>B.2 Configuring JasperReports Server to Accept Web Services Calls</td>
<td>126</td>
</tr>
<tr>
<td>B.3 Configuring Liferay to Access JasperReports Server</td>
<td>126</td>
</tr>
<tr>
<td>B.4 Setting the Portal Organization</td>
<td>127</td>
</tr>
<tr>
<td>B.5 Testing Liferay</td>
<td>127</td>
</tr>
<tr>
<td>B.6 Deploying the JasperReports Server Portlet WAR File</td>
<td>127</td>
</tr>
<tr>
<td>B.7 Configuring a Default Report to Display</td>
<td>128</td>
</tr>
<tr>
<td>B.8 Testing the JasperReports Server Portlet</td>
<td>129</td>
</tr>
<tr>
<td>B.9 JasperReports Server Portlet Configuration Options</td>
<td>129</td>
</tr>
<tr>
<td>B.10 Setting up JasperReports Library Hyperlinks for Use in a Portlet Environment</td>
<td>131</td>
</tr>
</tbody>
</table>

## Appendix C Localization ................................................................. 133

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1 Configuring JasperReports Server for the Default Multi-byte Fonts</td>
<td>133</td>
</tr>
<tr>
<td>C.2 UTF-8 Configuration</td>
<td>133</td>
</tr>
<tr>
<td>C.2.1 Tomcat</td>
<td>134</td>
</tr>
<tr>
<td>C.2.2 JBoss</td>
<td>134</td>
</tr>
<tr>
<td>C.2.3 PostgreSQL</td>
<td>134</td>
</tr>
<tr>
<td>C.2.4 MySQL</td>
<td>135</td>
</tr>
<tr>
<td>C.2.5 Oracle</td>
<td>135</td>
</tr>
<tr>
<td>C.3 Creating a Locale</td>
<td>136</td>
</tr>
<tr>
<td>C.3.1 About Properties Files</td>
<td>136</td>
</tr>
<tr>
<td>C.3.2 Creating a Resource Bundle</td>
<td>138</td>
</tr>
<tr>
<td>C.3.3 Changing Format Masks and Date Formats</td>
<td>138</td>
</tr>
<tr>
<td>C.4 Configuring JasperReports Server to Offer a Locale</td>
<td>140</td>
</tr>
<tr>
<td>C.4.1 Specifying Additional Locales</td>
<td>140</td>
</tr>
<tr>
<td>C.4.2 Specifying Additional Time Zones</td>
<td>141</td>
</tr>
<tr>
<td>C.4.3 Setting a Default Time Zone</td>
<td>142</td>
</tr>
<tr>
<td>C.5 Character Encoding and Fonts</td>
<td>142</td>
</tr>
<tr>
<td>C.5.1 Changing Character Encoding</td>
<td>143</td>
</tr>
<tr>
<td>C.5.2 Working with Fonts</td>
<td>144</td>
</tr>
<tr>
<td>C.6 JasperBabylon</td>
<td>146</td>
</tr>
</tbody>
</table>

## Glossary ........................................................... 147

## Index ............................................................... 155
CHAPTER 1   OVERVIEW OF 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 providing shared services, such as security, a repository, and scheduling.

The heart of the Jaspersoft BI Suite is the server, which provides the ability to:
- Easily create new reports using an intuitive web-based drag-and-drop Ad Hoc reporting interface.
- Efficiently and securely manage many reports.
- Interact with reports, including entering parameters and drilling on data.
- Arrange reports and web content to create appealing, data-rich Jaspersoft Dashboards that quickly convey business trends.

Jaspersoft OLAP is an optional component of JasperReports Server, controlled by licence and described in its own user guide.

While the Ad Hoc Editor lets users create simple reports, more complex reports can be created outside of the server. You can use Jaspersoft iReport Designer (hereafter called “iReport”) or write your own JRXML code to create a report that can be run in the server. Jaspersoft recommends that you use iReport unless you have a thorough understanding of the JasperReports Library file structure. See Unified Reporter Server User Guide for more information.

Jaspersoft provides several other source of information to help extend your knowledge of JasperReports Server:
- Our Ultimate Guides document advanced features, best practices, and numerous examples. The guides are available as PDFs for purchase in the Jaspersoft online store. Commercial customers can download them freely in the support portal.
- Our free Business Intelligence Tutorials let you learn at your own pace, and cover topics for developers, administrators, business users, and data integrators. The tutorials are available online in the Professional Services section of our website.

Our free samples, which are installed with JasperReports Library, iReport, and JasperReports Server, are documented online. The samples documentation can be found on our community website.

This administrator guide describes features that are only available to users who have administrator roles. Many of the configuration procedures also assume you have unlimited access to the JasperReports Server host computer.

This chapter contains the following sections:
- Overview of Organizations
- Overview of the Repository
- Overview of Users and Roles
- Security Overview
- Administrator Login
- Administrator Pages
1.1 Overview of Organizations

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

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

1.1.1 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 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. Even in a single organization, there is a system admin and an organization admin that share administrative duties. If the needs of your business call for more organizations, you must 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.

1.1.2 Multiple Organizations

There are many scenarios 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 being offered to many customers. It integrates JasperReports Server in its application in order to offer dashboards, reports, and analysis. There are a number of common reports and data sources that 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 JasperReports Server maps the application’s authentication scheme to the correct organization. The organization mechanism 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 to consolidate the BI environment so that all departments are sharing a common BI infrastructure. Corporate IT only needs to deploy and maintain a single instance of JasperReports Server, and 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 user name. 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 like it. In all cases, administrators can configure secure environments for any number of organizations, and end-users experience a powerful BI platform that is tailored to their needs.

Each organization or hierarchy of organizations co-exists independently in the same instance of JasperReports Server, which isolates neighboring organizations from each other but allows parent organizations to have full control over their suborganizations. Users may access only the data and resources in their organization or a suborganization, and administrators may define roles and set permissions to further restrict access.

1.1.3 Levels of Administrators

Each organization has an administrator who can manage users, roles, and repository permissions in that organization. The administration of organizations is hierarchical, meaning that the administrator can also manage all users and roles in suborganizations of any level.

There are essentially three levels of administration:

- The system administrator – Also called system admin. The ID 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 of 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 due to the hierarchy of organizations, manages a limited set of user, roles, and repository objects and may be overridden by a top-level organization admin.

The most important distinction is between the system admin and organization admin. Even in the case of a single default organization, there is a system admin for server settings, and an organization admin for the single organization. The system admin can manage all users and the entire repository, but sometimes it is more convenient to use the organization admin to do this because the organization admin sees the repository in the same way as the organization users.

When there are suborganizations, the administrator of the parent organization can either manage their users and roles, or delegate those tasks to an administrator in each suborganization. The administrator of a suborganization is limited to accessing resources and managing users and roles in the suborganization, thereby maintaining the security of the parent organization and any of the parent’s other suborganizations.

1.2 Overview of the Repository

The repository is a hierarchical structure of folders where JasperReports Server, administrators, and users store resources for creating, running, and viewing 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.

1.2.1 Folder Structure

The root of the repository tree structure is accessible only to the system admin logged in as superuser (Figure 1-1). It contains the folders for each organization and folders for certain configuration settings.

![Figure 1-1 Root of the Repository Showing Default Folders](image)

In the repository, each organization has its own branch, contained in a folder named after the organization. Each organization’s top-level folder also contains a folder called Organizations so that suborganizations can be created.

Figure 1-1 shows the superuser’s view, starting at root. In the Organizations folder there is a folder for the default organization, which is named Organization. This figure also shows a second top-level organization.

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 only view and create folders in the organizations they administer. In general, Jaspersoft recommends that you avoid placing resources directly in the root or organization folder. Instead, use folders for various resource types, as in the sample data.
1.2.2 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 iReport, are uploaded from files. Others, such as data sources and Domains, are created in JasperReports Server itself. Of course, dashboards and Ad Hoc can also be saved in the repository to be run as often as needed, and output such as PDF or HTML can be saved in the repository as well.

All resources, including folders, have a unique ID, a name, and an optional description. The ID of a resource, along with the ID of its enclosing folders creates a path that can be used to reference resources. The name and description appear in the user interface when browsing or searching the repository.

Resources are stored in an internal format that is not accessible to users or administrators, although certain objects can be downloaded to your file system in an output format such as XML. Any resource may 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, user names, and roles. The system admin and organization admin can define permissions as explained in section 1.3, “Overview of Users and Roles,” on page 14.

1.2.3 Sample Data

When you install the sample data in JasperReports Server, the default organization (Organization) has sample content. Figure 1-2 shows the folders containing the sample resources, as seen by the system admin and default organization admin.

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.

The Public folder is a special system folder that appears at the root and in every organization folder. 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.
1.2.4 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. In many cases, such as when you are editing permissions, you can use either mode, but sometimes only one mode displays the features and commands you need, such as when you copy an object.

- **Browse** - On the Home page, click **View > Repository**.

  ![Figure 1-3 Browsing the Repository](image)

  In Browse mode, 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 you to perform actions such as **Copy**, **Cut**, **Paste**, and **Delete**; select several resources in a same folder to perform actions in bulk. Search mode does not have the tool bar.

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

  ![Figure 1-4 Searching the Repository](image)

  The filters in the left-hand panel allow users to refine their search. The **View > Reports** and **View > OLAP Views** also use the search feature with preset filters to find all reports and all OLAP views to which the user has access.

For more information on browsing and searching the repository, see the JasperReports Server User Guide.
1.3 Overview of Users and Roles

User accounts and role membership provide authentication and authorization mechanisms to implement access control in JasperReports Server. Users enter an organization name, a login name, and a password in order to access JasperReports Server. Administrators assign named roles to users and then create role-based permissions to further restrict access to objects in the repository and to data in Domains.

Both users and roles are associated with the organizations in which they are defined, and they follow the same hierarchical model. Users and roles defined in an organization may be granted or denied access to any repository folder or object in the organization or its suborganizations. However, the administrator of the suborganization has no visibility of 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 the organization must be given when logging in, JasperReports Server can distinguish between every user. In some cases such as web services, a user is identified by the unique string username|organization_ID.

Access to the repository is defined directly on the repository resources. Administrators may define a level of access, such as read-write, read-only or no access, and each permission may be based either on a user name or on a role name.

1.3.1 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 setting the password. However, no administrator can ever view a user’s existing password in clear text.
- Login as any user in the organization to test permissions.
- Create, modify, and delete roles.
- Assign roles to users.
- Set access permissions on repository folders and resources.

1.3.2 Delegated Administration

JasperReports Server enables three levels of delegated administration:

- The hierarchical structure of organizations means administrators in each organization are limited to actions within their organization. But this only applies to multiple organizations where it makes sense to have subordinate administrators.
- The Administer permissions allows a user to view and set permissions on a folder or resource. This allows a power-user to manage her own 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 superuser or jasperadmin has the same abilities.

In the case of true delegated administration, there are three factors that 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 in the repository, 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, who by definition has this role, can assign it 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; organization admins cannot see or assign this role.

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 \texttt{ROLE\_SUPERUSER}, ever view or modify any organization, user, role, or folder outside of the organization to which the user belongs.

Any administrator can grant \texttt{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 \texttt{ROLE\_ADMINISTRATOR} and \texttt{ROLE\_SUPERUSER} to grant them system admin privileges. For further information about these roles, see section 3.5, “Permissions,” on page 42.

1.4 Security Overview

JasperReports Server ensures that people can only access the data they are allowed to see. The mechanisms that define organizations, users, roles, and repository resources work together to provide complete access control that includes:

• Authentication – Restricts access to identified users and protects that access with passwords. Defines roles for grouping users and assigning permissions. Authentication is further explained in the next section.
• Authorization – Controls access to repository objects, pages, and menus based on users and roles. Authorization is further explained in a following section.
• Data level security – Defines row and column level permissions to access your data. Row and column level permissions can be defined and enforced in Domains. For more information, refer to the \textit{JasperReports Server User Guide}. If you implement Jaspersoft OLAP, you can use roles to secure your data at any level of an analysis schema’s hierarchy. For more information, refer to the \textit{Jaspersoft OLAP User Guide}.

Administrators must keep security in mind at all times when managing organizations, user, roles, and resources, because the security mechanism behind each of these rely on the others.

1.4.1 Authentication

The first part of security is to define user accounts and secure them with passwords. 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 section 2.2, “Managing Users,” on page 23.

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. The server stores role definition in its private database, and administrators create, modify, and delete roles through the administrator pages, as described in section 2.3, “Managing Roles,” on page 27.

JasperReports Server relies on the open source Spring security framework; it has many configurable options for:

• External authentication services such as LDAP (used by Microsoft Active Directory and Novell eDirectory)
• Single sign-on using JA-SIG's Central Authentication Service (CAS)
• Java Authentication and Authorization Service (JAAS)
• Container security (Tomcat, Jetty)
• SiteMinder
• Anonymous user access (disabled by default)

JasperReports Server also supports these encryption and authentication standards:

• HTTPS, including requiring HTTPS
• HTTP Basic
• HTTP Digest
• X509

The Spring framework is readily extensible to integrate with custom and commercial authentication services and transports. Authentication occurs by default through the web user interface, forcing login, and/or through HTTP Basic authentication for web services, such as Jaspersoft iReport Designer and for XML/A traffic. The server can automatically synchronize with an
external authentication service. The external users don’t need to be created manually in the server first. Both users and roles are created automatically in the server from their definitions in an external authentication service. For an overview of the authentication system and details about external authentication, see the JasperReports Server Authentication Cookbook.

1.4.2 Authorization Overview

With a user’s identity and roles established, JasperReports Server controls the user’s access in these ways:

**Menu options 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.

**Organization scope**
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 section 3.4, “Multiple Organizations in the Repository,” on page 40.

**Resource permissions**
Administrators can define access permissions on every folder and resource in the repository. Permissions can be defined for every role and every user, or they can be left undefined so they are inherited from the parent folder. For example, user may have read-write access to a folder where they create reports, but the administrator can also create shared reports in the same folder that are set to read-only. The possible permissions are: no access, execute only, read-only, read-delete, read-write-delete, and administer (see section 3.5, “Permissions,” on page 42).

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 are cumulative, meaning that a user has the most permissive access that is granted to any of the roles to which the user belongs.

**Administrator privileges**
JasperReports Server distinguishes between reading or writing a resource in the repository and viewing or editing the internal definition of a resource. For security purposes, granting a user read or write permission on a resource does not allow viewing or editing the resource definition. For example, users need execute or read permission on a data source to run reports that use it, but they cannot view the data source’s definition which includes a database password. Also, only administrators may interact with theme folders to upload, download, and activate CSS files that control the user interface.

**Data-level security**
Data-level security defines what data can be retrieved and viewed in a report, based on the username and roles of the user who runs the report. For example, a management report could allow any user to see the management hierarchy, managers would see the salary information for their direct employees, and only human resource managers would see all salary values.

Data-level security in Domains is explained in the JasperReports Server User Guide. Data-level security through analysis views is covered in the Jaspersoft OLAP User Guide.

**Profile attributes**
A profile attribute is a name-value pair inserted in the database record of a user object, such as a user account or role. Profile attributes may then be referenced in data-level security definitions to provide additional security. Profile attributes can be added only by database administrators. For information on defining profile attributes, see the JasperReports Server Ultimate Guide.
1.5 Administrator Login

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

- System admin: username superuser and password superuser
- Organization admin: username jasperadmin and password jasperadmin

For security reasons, always change the default administrator passwords immediately after installing JasperReports Server. For instructions, see section 2.2.3, “Editing a User,” on page 25.

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.

1.5.1 JasperReports Server Heartbeat

When you log in to JasperReports Server for the first time after installation, you 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, RDBMS (type and version), and JasperReports Server edition and version number. By tracking this information, Jaspersoft 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.

1.5.2 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 very 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 as well.

To set the email and passwords on the administrator accounts, edit the user account information as described in section 2.2.3, “Editing a User,” on page 25.

1.6 Administrator Pages

Administrators have access to special pages to manage the server. After logging in, click the large Manage Server button on the Getting Started page or select the Manage > Admin Home menu item on any page.
The administrator controls are different for system and organization admins, as shown in Figure 1-5. 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. In addition, only system admins have access to the server-wide settings that apply to Jaspersoft OLAP, Ad Hoc, and logs.

Figure Figure 1-6 show the Home page for system admins that includes the OLAP Settings button not available to organization admins.

Figure Figure 1-7 shows the dialog that appears when you click the About JasperReports Server link in the footer of all pages. The dialog displays the product version number. It also shows the software build, your license type, and its expiration. Please have this information if you need to contact Jaspersoft for support.
CHAPTER 2 ORGANIZATION, USER, AND ROLE MANAGEMENT

In a single-organization deployment, the administrator only needs to create the users and roles. In deployments with multiple organizations, administrators need to create organizations, populate them with users, and create the roles that they use afterwards to set access permissions.

In a deployment with multiple organizations, there can be administrators at every level of the hierarchy, as described in section 1.3.2, “Delegated Administration,” on page 14. Part of any large deployment is to designate the administrators who are 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 accommodates all levels of administrators and 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 insures that an organization administrator cannot see roles and users from parent organizations.

This chapter contains the following sections:
- Managing Organizations
- Managing Users
- Managing Roles

2.1 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, whereas organization admins are limited to suborganizations.

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

Figure 2-1 shows the organizations that the system admin/superuser can view, that is, all the organizations in the repository. As shown in the Organizations panel on the left, the system admin’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 repository. In this example, there are two top-level organizations, and one of them has several suborganizations.
2.1.1 Viewing Organization Properties

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

2. Select Manage > Organizations.
   The organization management page appears, as shown in Figure 2-1 or Figure 2-2.

3. To select an organization, click its parent in the left-hand Organizations panel, then select it in the center Organization panel. If there are many organizations, you can enter a search term to find a specific organization. However, the search term only searches the list of organization in the center Organizations panel.

4. Once an organization is selected, the Properties panel shows information about the organization:
   • Name – Display name of the organization that appears on the organization’s top folder.
   • ID – Unique and permanent ID of the organization that is used for logging into the organization.
   • Alias – Unique but editable short name for the organization that can also be used when logging in.
   • Description – Optional description that only appears 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 – Counts 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.

2.1.2 Creating an Organization
1. Log in as a user with administrative privileges for the parent of the new organization.
2. Click Manage > Organizations.
3. In the left-hand 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.

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 you needed. The Description is optional. Figure 2-3 shows 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 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 section 2.2, “Managing Users,” on page 23.

• The organization has no roles of its own. The default users have the system-wide roles ROLE_USER and ROLE_ADMINISTRATOR.
• There is a folder created in the repository, under the parent’s Organization folder. The new organization folder contains a copy of the parent’s Organization/Folder Template folder. To manage the Organization folders, select View > Repository.

2.1.3 Default Folders for Organizations
Every organization contains a special folder named Organizations where suborganizations are created. The Organizations folder always contains a folder named Folder Template. When a new organization is created, the entire contents of the Folder
Template is copied to create the new organization’s folders. Admins can add folders and resources in Folder Template, and these are also copied when additional organizations are created.

The default folders in the Folder Template are:

- **Ad Hoc Components\Topics** – The location where the Ad Hoc Designer looks for Topics to create new reports.
- **Temp** – A folder visible only administrators, used by the server to store temporary files.
- **Themes** – A special folder managed by the system to contain CSS files that define the user interface.

The Public folder visible in every organization is a special shared 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 respective Folder Template for use in creating suborganizations.

Finally, the Folder Template itself is copied into a new organization, so new suborganizations have the same default folders and resources as their parent.

### 2.1.4 Editing an Organization

1. Log in as a user with administrative privileges for the organization.
2. Click **Manage > Organizations**.
3. In the left-hand Organizations panels, select the organization’s parent. In the center Organizations panel, select the organization.
4. In the right-hand Properties panel, click **Edit**. The fields in the organization’s Properties panel become editable.

5. Change the organization properties as needed. Changing the organization name changes the name of the organization’s folder, as well, but no other data. The organization ID cannot be changed; it always has the value defined when the organization is created. The alias and description can be changed.
6. Click **Save** to keep your changes, or **Cancel** to quit without saving.

### 2.1.5 Deleting an Organization

1. Log in as a user with administrative privileges for the organization.
2. Click **Manage > Organizations**.
3. In the left-hand Organizations panels, select the organization’s parent. In the center Organizations panel, select the organization.
4. In center Organizations panel, click **Delete**.
   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.

### 2.2 Managing Users

As with organizations, system admins can manage all users in all organizations, as well as create users outside of organizations, as described in section **1.3.2, “Delegated Administration,” on page 14**. Organization admins can manage only the users in the organizations they administer.

The default installation of JasperReports Server contains the following users:

**Table 2-1 Default Users in JasperReports Server Installations**

<table>
<thead>
<tr>
<th>User Name</th>
<th>Default Password*</th>
<th>Organization</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>superuser</td>
<td>superuser</td>
<td>none</td>
<td>Default system admin.</td>
</tr>
<tr>
<td>anonymousUser</td>
<td>anonymoususer</td>
<td>none</td>
<td>Allows anonymous login; disabled by default.</td>
</tr>
<tr>
<td>jasperadmin</td>
<td>jasperadmin</td>
<td>Organization</td>
<td>Default organization admin in every organization.</td>
</tr>
<tr>
<td>joeuser</td>
<td>joeuser</td>
<td>Organization</td>
<td>Default end user in every organization.</td>
</tr>
<tr>
<td>demo</td>
<td>demo</td>
<td>Organization</td>
<td>Included for use with sample data.</td>
</tr>
<tr>
<td>CaliforniaUser</td>
<td>CaliforniaUser</td>
<td>Organization</td>
<td>Included for use with sample data.</td>
</tr>
</tbody>
</table>

* Passwords are case-sensitive.

You should advise your users to change their passwords regularly. To configure periodic expiration of their passwords, refer to section **7.2, “Configuring Password Options,” on page 94**.

#### 2.2.1 Viewing User Properties

1. Log in as a user with administrative privileges for the user’s organization.
2. Select **Manage > Users** or, on the Admin Home page, click **Users**.
   As shown in **Figure 2-5**, the Manage Users page displays the users in each organization and properties for the selected user.
The list of users includes everyone in the chosen organization and its suborganizations. The same user ID may appear more than once in the Users panel, indicating that users with the same ID were created in different organizations. The third column gives the organization name of a particular user.

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 that is created.

3. To locate a user:
   - To browse for users, expand the organization hierarchy in the left-hand panel, and select an organization. Scroll through the list of users, or choose a suborganization to reduce the list.
   - To search for a specific 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 users in the selected organization and suborganizations whose username contains the search string. If necessary, scroll through the results or refine your search.

   To stop the search, click 

4. Select the user in the Users panel. The user’s properties appear in the Properties panel. The properties include the user’s name, user ID, email address, assigned roles, user status, and profile attributes. User status can be Enabled or Disabled; disabled users are displayed in gray text in the list of users. Profile attributes are special user attributes that are added directly in the database, not through this Manage Users page (see JasperReports Server Ultimate Guide). For convenience, the role names link to the role management page for each role.

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 section 2.3, “Managing Roles,” on page 27.

2.2.2 Creating a User

1. Log in as a user with administrative privileges for the organization to which the user will belong.
2. Select Manage > Users or, on the Admin Home page, click Users.
3. In the Organizations panels, select the organization to which the user will belong, then click Add User.

   The Add User dialog appears.
4. Enter the following information:
   - User name – The full name of the person associated with the user account. The name is optional but recommended; it can be in any format or convention. JasperReports Server always displays the current user’s name in the top right-hand corner of the screen.
   - User ID – Generated automatically from the user name; 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. User IDs must be unique within an organization, but may exist in other organizations.
   - Email – The email address of the person. The email is optional but the address must be in a valid format.
   - Password and confirmation – Enter the same password in both fields.
   - User is enabled – Select the checkbox to enable the user right away.

   Users that are not enabled cannot log in. JasperReports Server automatically assigns every user the default role, ROLE_USER, but you might want to delay enabling the user until you assign more roles. For more information on roles, see section 2.3, “Managing Roles,” on page 27.

5. Click Add User to <organization> to create the user.

   The new user appears in the Users panel, unless you entered a search term that excludes it. If you want to assign roles to the user, click Edit in the Properties panel of the new user, as shown in the following section.

2.2.3 Editing a User

One way to assign roles is to add available roles to a given user. Alternatively, when you edit roles, you can assign any number of users to a given role.

1. Log in as a user with administrative privileges for the user’s organization.
2. Click Manage > Users or, on the Admin Home page, click Users.
3. In the Organizations panel, select the user’s organization, then in the Users panel, select the user.
   - The user is displayed in the Properties panel.
4. In the Properties panel, click Edit.
5. Modify the user name, email, password, and enabled status as needed.
   You cannot edit the user ID or the profile attributes. The user ID always has the value defined when the user is created originally; the profile attributes can only be modified in the database (see the JasperReports Server Ultimate Guide).

6. To assign or remove roles from the user, select the roles, and click 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 section 2.3, “Managing Roles,” on page 27.

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

8. In the Properties panel, click **Login as User** to test the user’s permissions, as explained in section 3.5.7, “Testing User Permissions,” on page 46.
   Logging in as another user is also necessary when you are maintaining resources that use absolute references in the repository. For more information, see section 3.4.3, “Referencing Resources in the Repository,” on page 41.

### 2.2.4 Enabling or Disabling Users in Bulk

Administrators sometimes need to prevent users from logging in by disabling the user accounts. For example, when performing configuration changes, you may want to lock out all users until the changes are finished. Bulk operations let administrators select any number of users, and **superuser** can select all users in the server, except himself.

1. Log in as a user with administrative privileges for the users’ organization.
2. Click **Manage > Users** or, on the Admin Home page, click **Users**.
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 list of users, select all the users to enable or disable. Use Control-click and Shift-click to make multiple selections. If the list of users 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.
2.2.5 Deleting One or More Users

1. Log in as a user with administrative privileges for the user’s organization.
2. Click Manage > Users or, on the Admin Home page, click Users.
3. In the Organizations panel, select the user’s organization; to delete multiple users in different organizations, select the common parent organization.
4. In the list of users, select the user to delete. 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.

2.3 Managing Roles

Roles define sets of users who are granted similar permissions. Administrators create roles, assigned them to users, and set permissions in the repository (see section 3.5, “Permissions,” on page 42). By default, JasperReports Server includes the following roles; some are needed for system operation, some are included as part of the sample data:

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROLE_SUPERUSER</td>
<td>This role determines system admin privileges, as explained in section 1.3.2, “Delegated Administration,” on page 14. It is a system-level role, however the system admin may assign it to organization admins in single-organization deployments.</td>
</tr>
<tr>
<td>ROLE_ADMINISTRATOR</td>
<td>This role determines organization admin privileges, as explained in section 1.3.2, “Delegated Administration,” on page 14. JasperReports Server automatically assigns this role to the default jasperadmin user in every new organization. It is a special system-level role that is visible in every organization and which organization admins may assign to other users.</td>
</tr>
<tr>
<td>ROLE_USER</td>
<td>Every user that logs into JasperReports Server must have this role. The server automatically assigns this role to every user that is created, and it cannot be removed. It is a special system-level role that is visible in every organization.</td>
</tr>
<tr>
<td>ROLE_ANONYMOUS</td>
<td>When anonymous access is enabled, JasperReports Server automatically assigns this role to any agent accessing the server without logging in. This role is also assigned to the default anonymous user. By default, anonymous access is disabled and this role isn’t used. It is a special system-level role that is visible in every organization.</td>
</tr>
<tr>
<td>ROLE_PORTLET</td>
<td>JasperReports Server assigns this role to users that are created automatically when a portal such as Liferay requests authentication for a connection. If the specified user name does not exist in the server, it is created, assigned the password of the user in the portal, and assigned the ROLE_PORTLET and ROLE_USER roles.</td>
</tr>
<tr>
<td>ROLE_DEMO</td>
<td>This role grants access to the SuperMart demo Home page, reports, and if you implement Jaspersoft OLAP, analysis 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 JasperReports Server. It is a special system-level role that is visible in every organization.</td>
</tr>
</tbody>
</table>
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 separate 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.

### 2.3.1 Viewing Role Properties

1. Log in as a user with administrative privileges for the role’s organization.
2. Select **Manage > Roles** or, on the Admin Home page, click **Roles**.

As shown in Figure 2-8, the Manage Roles page displays the roles in each organization and properties for each role.

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROLE_SUPERMART_MANAGER</td>
<td>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.</td>
</tr>
<tr>
<td>ROLE_ETL_ADMIN</td>
<td>This role no longer governs any JasperReports Server permissions or functionality, unless your server is integrated with Talend Integration Suite Enterprise Edition (TIS EE). Otherwise, it can be deleted safely.</td>
</tr>
</tbody>
</table>

The list of roles includes all roles in the chosen organization and its suborganizations. The list of roles also includes the five default system-level roles. The same role name may appear more than once, indicating that 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, then click the role in the Roles panel.

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 name contains the search string. If necessary, scroll through the new list or refine your search.
To stop the search, click ✗

4. 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 is defined, and the list of users to whom the role has been assigned. The list of users shows only their user IDs, but hovering over an ID displays a tooltip with the full name and organization, as shown in Figure 2-8.

   ![Figure 2-8](image)

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

2.3.2 Creating a Role

1. Log in as a user with administrative privileges for the organization in which the role will be used.
2. Select Manage > Roles or, on the Admin Home page, click Roles.
3. In the Organizations panels, select the organization to which the role will belong, then click Add Role.
   The Add Role dialog appears.

![Figure 2-9 Adding a Role](image)

4. Enter the name of the role. Roles have no other properties or settings.
5. Click Add Role to <organization> to create the role.
   The new role appears in the Roles panel, unless you entered a search term that excludes it. If you want to assign users to the role, click Edit in the Properties panel of the new role, as shown in the following section.

2.3.3 Assigning Users to a Role

The management interface for roles lets you assign multiple users to one role. To assign multiple roles to a single user, edit the user’s properties with the procedure in section 2.2.3, “Editing a User,” on page 25.

1. Log in as a user with administrative privileges for the organization in which the role is defined.
2. Select Manage > Roles or, on the Admin Home page, click Roles.
3. In the Organizations panels, select the role’s organization, then select the role in the Roles panel.

   ![Figure 2-8](image)

   Unless you are logged in as the system admin, you cannot edit or delete the five special system-level roles.

4. In the Properties panel, click Edit.
   The role’s properties become editable. You can change the role name and the users assigned to it.
5. 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 verify the files as well. For more information, see the JasperReports Server User Guide.

6. To assign or remove users from the role, select the users, and click the arrow buttons between the Users Available and Users Assigned lists.

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

2.3.4 Deleting One or More Roles

1. Log in as a user with administrative privileges for the organization in which the role is defined.

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

3. In the Organizations panels, select the role’s organization, then select the role in the Roles panel. Use Control-click and Shift-click to make multiple selections.

Unless you are logged in as the system admin, you cannot edit or delete the five special system-level roles.

4. In the tool bar of the Roles panel, click Delete and confirm the action.
CHAPTER 3 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 that 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 a theme.

The repository is structured as a hierarchy of folders that is 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 covered in the following sections:

- Section 1.2, “Overview of the Repository,” on page 11 explains the basic structure and navigation of the repository.
- Chapter 4, “Resources in the Repository,” on page 47 describes the resources that administrators create for reports.
- Chapter 5, “Themes,” on page 75 describes how special CSS and image files in the repository define the user interface.

You can also access the repository programmatically by using the web services and APIs. For more information on these features, refer to the JasperReports Server Web Services Guide and to the JasperReports Server Ultimate Guide, respectively.

This chapter contains the following sections:

- Resource Types
- JasperReport Structure
- Managing Folders and Resources
- Multiple Organizations in the Repository
- Permissions

3.1 Resource Types

Resources in the repository have a type that determines how users can interact with it. The various resource types are based on various reports and report elements that users and administrators store in the repository.
There are two fundamental types of resources: those that the end user creates, and those that the administrator must create. End users create the following resources. Procedures for end users to create these resources are described in the *JasperReports Server User Guide*:

### Table 3-1 Resources Created by End Users

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ad Hoc Report</td>
<td>Created interactively in the Ad Hoc Editor by dragging and dropping columns of data onto a table, chart, or crosstab. Users may then filter the data and save the report. An Ad Hoc report can either be run or re-opened for further data exploration.</td>
</tr>
<tr>
<td>Dashboard</td>
<td>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.</td>
</tr>
<tr>
<td>Content Resource</td>
<td>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.</td>
</tr>
<tr>
<td>JasperReport</td>
<td>A complex type that combines a JRXML file, a data source, and optional components such as input controls to define a report that users can run in the server. Depending on the usage scenario, administrators also create JasperReports in the server. For more information, see section 3.2, “JasperReport Structure,” on page 33.</td>
</tr>
<tr>
<td>Report Version</td>
<td>Reports with input controls allow you to save combinations of input data so that you can run a custom version of the report directly. In the repository, report versions are always listed under the original report.</td>
</tr>
</tbody>
</table>

The other types of resources are all created and managed in the repository by administrators. The following resources generally support the creation of reports:

### Table 3-2 Resources Created by Administrators

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
<td>A connection that points to a database or other data store. Data sources define where data is stored for running reports. There are several types of data sources, based on the type of connection or location of the data: JDBC, JNDI, and bean data sources. For more information, see section 4.1, “Data Sources,” on page 47.</td>
</tr>
<tr>
<td>Datatype</td>
<td>A basic type 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.</td>
</tr>
<tr>
<td>Domain</td>
<td>A metadata layer that selects, joins, and filters tables and fields from your data and lets you give them user-friendly labels. A Domain can be the basis for an Ad Hoc report. Domains also support row-level security and localization of labels. Domains are further documented in the <em>JasperReports Server User Guide</em>.</td>
</tr>
<tr>
<td>File</td>
<td>A resource that stores a file in the repository. <em>Table 4-2, “File Resource Types,” on page 71</em> gives the list of supported file formats and their purpose.</td>
</tr>
<tr>
<td>Input Control</td>
<td>A complex type that specifies what 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.</td>
</tr>
<tr>
<td>List of Values</td>
<td>A basic type 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 the values 1 to 12.</td>
</tr>
<tr>
<td>Query</td>
<td>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.</td>
</tr>
</tbody>
</table>
Administrators may 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.

Table 3-3  OLAP Resources Created by Administrators

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mondrian XML/A Source</td>
<td>A server-side XMLA source definition of a remote client-side XML/A connection.</td>
</tr>
<tr>
<td>OLAP Client Connection</td>
<td>Defines 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 (XMLA connection).</td>
</tr>
<tr>
<td>OLAP View</td>
<td>If you implement Jaspersoft OLAP, a view of multidimensional data that is based on an OLAP client connection and an MDX query. Like JasperReports, they are collections of individual resources that define how to access and present the data.</td>
</tr>
</tbody>
</table>

3.2  JasperReport Structure

The resource in the repository 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 open source library that the server uses to render reports.

A JasperReport is a complex resource that is composed of other resources:

- A JRXML file that defines the report, called the main JRXML.
- 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 any parameters that users may enter before running the report. Input controls are composed of either:
  - A datatype definition
  - A list of values
- Any additional file resources, such as images, fonts, and resource bundles referenced by the report template.
- If the report includes sub-reports, the JRXML files for the subreports.

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 component resources.

3.2.1  Referencing Resources in the Repository

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

In environments that rely directly on JasperReports, shared resources are usually stored on a network file system 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. In addition, storing the resources in the file system discourages their reuse: your developers may invest time in creating new versions of resources that already exist because they don’t know about them.

By storing resources in the repository, JasperReports Server makes it easy and reliable to share resources such as images, style templates, and subreports between reports. In order to share resources, reports must reference resources with a special syntax. To reference resources in the repository, use a URI with the repo: prefix in your JRXML. This is similar to the file: syntax for file-based resources and the http: syntax that is used for web-based resources, however it is recognized only inside of JasperReports Server.

When you use this syntax in your JRXML files, 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 iReport, through the JasperReports Server user interface, or through the server’s APIs. For example, when you update a logo image resource, all reports that reference the resource also display the new logo.

The following sections describe several ways of using the repo: syntax to create different kinds of references.
3.2.2 Absolute References

Absolute references are URIs in the JRXML of the report that specify a resource’s complete repository path. The following example references an image named myImage that is stored in the /images folder:

```
repo:/images/myImage
```

The path must start with a `/` to represent the root of the repository, and is composed of the resource ID of every parent folder, ending with the ID of the resource. iReport with the JasperReports Server plug-in 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 the user’s organization, as described in section 3.4, “Multiple Organizations in the Repository,” on page 40.

When uploading the JRXML with absolute resource references as part of a JasperReport in the server, you only need to ensure 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.

3.2.3 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 placeholder name for the resource, for example:

```
repo:myImageLink
```

When you upload a JRXML with this reference, the server prompts you to provide the resource. You have two choices:

- Creating 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.
- Selecting a resource from the repository, called an external reference because it is external to the JasperReport. This allows any number of reports to link to the same resource, yet allow that resource to 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 JasperReport, and the server doesn’t allow you to delete resources that are being referenced. No one can delete the resource without first removing the external reference, either by editing the JasperReport to change the external reference or by deleting the JasperReport that contains it.

In cases when you don’t want to reference existing resources, local resources allow reports to be highly customized and self-contained. A local resource that is 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.

Users who are not administrators may create JasperReports but not other resources in the repository. Therefore, if an administrator does not provide them resources for external references, their only option is to create local resources for all dependencies within the JasperReport.

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 of creating a new local resource or of using 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 query local 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. Indirect references are also used implicitly when administrators define the following resources in the repository:

- For input controls, you must specify a datatype or list of values, either as a local resource or an external reference to one in the repository.
- A query may optionally define a data source, either by creating a local data source resource, or by referencing an existing one in the repository.
- Domains may reference XML design files that are either created in the Domain Designer or uploaded from an external file resource. Domains also have XML security files and resource bundles, both of which can be either uploaded as local file resources or referenced from file resources in the repository.
- OLAP resources in Jaspersoft OLAP rely on other OLAP resources, for example views reference connections, and connections reference schemas. In each case, you can define local resources or external references. For more information, see the Jaspersoft OLAP User Guide.

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 use an external reference to its datatype. The server still manages the dependency between the local input control and the datatype resource in the repository.

### 3.3 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 that are available. For the definition of the permissions on folders and resources, see section 3.5, “Permissions,” on page 42.

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

Another scenario that administrators can set up involves the resources for users to create JasperReports. When given write permission, users have the ability to upload JRXML files and define resources inside the JasperReport. But for security reasons, only the administrator can create shared data sources and other resources in the repository. If many users are uploading JRXML files as JasperReports, the administrator should create shared data sources and resources ahead of time in the repository. This also has the benefit of simplifying maintenance, for example updating one shared logo file instead of having many users edit their reports.

#### 3.3.1 Resource IDs

All resources, including folders, have an ID, a name, and an optional description:

- The ID is used internally to reference resources. As with files, the ID must be unique within its folder, but may exist in several folders.
- The name is a label for the resource displayed in the repository UI.
- The optional description appears in the repository and in tooltips. You can give longer descriptions to resources to help users understand their contents or purpose.

As in a file system, the IDs of nested folders containing a resource define the path to the object. For example, the path to a report might be: /reports/samples/Freight. The path of every resource is shown under its name in the repository listing or search results.

To view the name and resource ID of a resource, right-click the folder or the resource in the repository or search results and select Properties... from the context menu.
If you have write or administer permission as shown in the figure, you can also edit the name and description of the resource.

### 3.3.2 Creating Folders

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

**To create a folder:**

1. Log on as a user who has write permission to the parent folder.
2. Select **View > Repository** and locate the parent folder in the left-hand Folders panel.
3. Right-click the parent folder and select **Add Folder** from the context menu.
4. The Add Folder dialog appears.
5. Enter a folder name and optional description, then click **Add**.

The folder is created in the repository. The name appears in the hierarchy of folders. The description is only visible when viewing the properties of the folder, as shown in Figure 3-1.

New folders and their future contents inherit the permissions of their parent folders. Administrators can change the permissions on the new folder, as described in section **3.5.6, “Setting Permissions,” on page 44**. End users cannot change permissions in the repository unless they have been granted the Administer permission by an administrator.
3.3.3 Adding Resources

Each resource has different requirements, for example some are created from uploaded files, others are created by defining values in a wizard. Specific procedures for adding each type of resource are given as follows:

- Interactive resources such as Ad Hoc reports 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 User Guide*.
- Mondrian and OLAP resources are covered in the *Jaspersoft OLAP User Guide*.
- Adding data sources, queries, input controls, and file resources is covered in Chapter 4, “Resources in the Repository,” on page 47.

Most resources are created through the Add Resource menu item on the context menu for folders in the repository. Figure 3-3 shows the full menu and submenu with all the resources administrators can create:

![Add Resource Context Menu Expanded](image)

For every resource you create, you must specify a name and resource ID that can used to reference the resource in the repository. In addition, each dialog has one or more pages for specifying the values and controls specific to the resource you are adding.

> 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 section 3.5.6, “Setting Permissions,” on page 44.

3.3.4 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 is copied from the name of the organization. Therefore, to change the name of the folder, you have to change the name of the organization, as described in section 2.1.4, “Editing an Organization,” on page 22.

**To rename a folder or resource:**

1. Log on 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.
You can change the folder or resource’s name and description, but not other information, such as resource type and ID; these data are created by the system and displayed here for information only.

4. Click **Submit** to save your changes.

### 3.3.5 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.

To copy or moving folders and resources:

1. Log on 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 object and select **Copy** or **Cut**. If the Cut command does not appear in the menu, you do not have delete permission required to move the object.
   
   You can select multiple resources with Control-click or Shift-click, but you can select only a single folder at a time.

4. Right-click the destination folder and select **Paste** in the context menu. If the **Paste** command does not appear in the menu, you do not have write permission there.

   Alternatively, you can drag the selected resource or folder to move it the destination folder. To perform a copy, you must press the Control key **before** clicking to drag. When dragging resources, the destination folder is highlighted in blue if you have permission to write there, and in gray otherwise. If you try to move a resource to a location where you don’t have permission, the server displays the contents of the target folder, but the resource is not moved.

The moved objects inherit their permissions from the destination folder in which they are placed; they do not keep the permissions they had before the move. If you want the objects to have other permissions, you must set the permissions again after the move (see section 3.5.6, “Setting Permissions,” on page 44).
3.3.6 Editing Resources

The various types of resources have different ways of being edited. For end users who work with interactive resources, editing mostly involves the **Open in Designer** action on the context menu for dashboards and Ad Hoc reports. The procedures in the following table are further described in the *JasperReports Server User Guide*.

### Table 3-4 Resources That End Users Can Edit

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>How to Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ad Hoc Report</td>
<td>Users select <strong>Open in Designer</strong> and modify the report interactively. After changing the content, users can overwrite the existing report or save as a new report. Ad Hoc reports 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.</td>
</tr>
<tr>
<td>Dashboard</td>
<td>Users select <strong>Open in Designer</strong> and modify the dashboard interactively. After changing the content, users can overwrite the existing dashboard or save it as a new dashboard.</td>
</tr>
<tr>
<td>Content Resource</td>
<td>Report output is a file stored in the repository. These files cannot be edited, only downloaded or deleted.</td>
</tr>
<tr>
<td>JasperReport</td>
<td>Users select <strong>Edit</strong> and change the data source, input controls, or file resources that are referenced in the JasperReport. Administrators may also edit JasperReports. For more information, see section 3.2, “JasperReport Structure,” on page 33.</td>
</tr>
<tr>
<td>Report Version</td>
<td>Users may select <strong>Edit</strong> to change the values stored as input parameters.</td>
</tr>
</tbody>
</table>

For the other resources in the following table, editing is accessible only to administrators. End users, even those with write permission on a resource, cannot edit these resources.

### Table 3-5 Resources That Only Administrators Can Edit

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>How to Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
<td>Administrators select <strong>Edit</strong> from the context menu on these resources. Editing these resources uses the same dialog that was used to define the resource when it is first added to JasperReports Server. Administrators can view the current definition of a resource or change the values that define a resource. For example, you could view the datatype of an input control, change a query, or upload a new file in a file resource. In the case of Domains, you also have access to the Domain Designer used when creating the Domain. You can add tables and fields, change filters, or change the display characteristics of items in the Domain. For more information about Domains, see the <em>JasperReports Server User Guide</em>.</td>
</tr>
<tr>
<td>Datatype</td>
<td></td>
</tr>
<tr>
<td>Domain</td>
<td></td>
</tr>
<tr>
<td>File</td>
<td></td>
</tr>
<tr>
<td>Input Control</td>
<td></td>
</tr>
<tr>
<td>List of Values</td>
<td>For all other resource types, see the procedure for creating it in Chapter 4, “Resources in the Repository,” on page 47.</td>
</tr>
<tr>
<td>Query</td>
<td></td>
</tr>
</tbody>
</table>

When editing a resource, there are several limitations:

- You can modify the name or description of the resource, but not its ID. If you must remove an ID, you need to create a new, similar resource and delete the old one.
- You cannot change the location of the resource. Some dialogs for editing a resource include the saved location, but the field is for information only. To change the location of the resource, see section 3.3.5, “Copying and Moving,” on page 38.
- 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.
3.3.7 Deleting Folders and Resources

Users with delete permission on a folder or resource can delete those objects from the repository. In order to delete a folder, the user must also have delete permission on all the resources and folders that the deleted folder contains, because the entire contents of the deleted folder are deleted as well.

Folders must be deleted one at a time, but multiple resources can be deleted together.

There is no undo from a delete.

The repository keeps track of which resources are referenced by other resources. It does not allow you to delete resources if they are still referenced by other resources. For example, an input type that is used by a report or a properties file that is 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, 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:**
1. Log on 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 that appears.

   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 only delete single folders at a time, although all contents of the folders, including subfolders are deleted. In the search results, you can select multiple resources and right-click to select Delete in the context menu.

3.4 Multiple Organizations in the Repository

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

3.4.1 Organization Folders

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

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

By default, users of an organization can also view and create folders and resources in any suborganization. To prevent this, administrators can change the permissions on the Organizations folder or individual organization folders.

The Organizations folder in every organization is a special folder that is managed by the server. Administrators cannot 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 section **2.1.3, “Default Folders for Organizations,” on page 21**.
3.4.2 Design Considerations

Careful design of the JasperReports Server repository leads to a clear and robust environment for your BI environment and easy yet 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 which are specific to a particular organization. This usually breaks down into several scenarios, depending on the resources that organizations need:

- Organizations have private resources - Organizations have separate data sources, reports, analysis views etc. This would be typical in an organization with departments. These private resources would be stored in each organization’s own folders, and perhaps only a few resources such as company logos would be shared between 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 analysis 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 have a hierarchical organization - You can have one organization containing other organizations. 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.

3.4.3 Referencing Resources in the Repository

All resources in the repository can be referenced by Universal Resource Identifiers (URIs), which specifies the resource name and folder path of the resource. 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.
  
  For example, 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 system admins, 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 only works if there is a folder at the root of the repository named images with a copy of the 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 analysis views, references to OLAP connections in OLAP views.
- 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, analysis views and OLAP connections, you must log in to that organization and do the tasks there. You cannot administer the resources as superuser or another organization’s admin.
- The three resources (report units, analysis 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 login as an admin of the organization using the references. See section 3.5.7, “Testing User Permissions,” on page 46.

### 3.4.4 Best Practices

The best practices for resources in a repository shared by multiple organizations are as follows:

- The system admin must login as an organization user in order to maintain or run organization resources
- Resources with absolute references to resources in organization folders only work for users within the organization or a parent organization
- If a JRXML that accesses organization resources with URIs must run across organizations, then 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.

### 3.5 Permissions

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

<table>
<thead>
<tr>
<th>Permission</th>
<th>Actions Granted on Repository Folders and Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Access</td>
<td>Users can never see or access the folder or resource either directly in the repository or indirectly when running a report, dashboard, or OLAP view.</td>
</tr>
<tr>
<td>Execute Only</td>
<td>Users can never see the folder or resource in the repository, but the reports, dashboard, or OLAP views that they run can access them.</td>
</tr>
</tbody>
</table>
| Read Only                | • See the folder or resource in any JasperReports Server dialog
• 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, or OLAP view
• Run a report in the background
• Schedule a report to run later |
| 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 + Delete    | • Add a subfolder
• Paste into a folder (copy or cut)
• Save a new Ad Hoc 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 report in the Ad Hoc editor or a dashboard in the designer
• Modify and overwrite an existing Ad Hoc 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) |
Permissions apply when browsing or searching the repository, as well as when using any dialog that accesses the repository, such as when 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 section 3.3.5, “Copying and Moving,” on page 38.
- Copying and cutting (moving) actions can only be completed if the user has Read + Write + Delete access to the folder in which the object is pasted. For more details, see section 3.3.5, “Copying and Moving,” on page 38.
- 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 or the contents of a folder is only allowed if no other resources rely on them. For more details, see section 3.3.7, “Deleting Folders and Resources,” on page 40.

### 3.5.1 Inheriting Permissions

According to the permission architecture, there is 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 of 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 this mechanism, administrators can manage large hierarchies of content and keep them secure. When the administrator sets 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 that are assigned on an organization’s top folder are inherited across the entire organization. Permissions that are 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.

### 3.5.2 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. Therefore, for every folder or resource, every user has a 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.

### 3.5.3 Administrator Permissions

The JasperReports Server authorization architecture distinguishes between administrators and all other users. Administrators are defined as users with either ROLE_SUPERUSER, ROLE_ADMINISTRATOR, or both. By design, system administrators with the ROLE_SUPERUSER always have irrevocable Administer access to the entire repository, including to 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 admin 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 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.

3.5.4 Execute-Only Permission

As in file systems, execute-only permission in JasperReports Server allows running reports, dashboards, 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 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, either accidentally or intentionally, by using the repository search.

As with all other permissions, execute-only permission is either role-based or user-based, so that certain users may access a resource from a running report, but not others.

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.

3.5.5 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 the permissions mechanism by viewing and setting permissions in the sample data, as described in the following section.

3.5.6 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 only set the permissions on that specific resource.

To set permissions on a folder or resource in the repository:
1. Log in as a user with administrative privileges.
2. In the repository, browse or search for the folder or resource.
3. Right-click the object and select Permissions... from the context menu.
The Permissions dialog opens. It shows 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 an asterisk (*).

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 cannot see the permission for the system admin (superuser) or for ROLE_SUPERUSER.

**Figure 3-5** shows the default role-based permissions on the sample Input Data Types folder as seen by the organization admin (jasperadmin). Members of certain roles can see and modify the input data types 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 cannot change the permission for their administrator role or user name, to prevent them from removing their ability to set permissions.

1. In the dialog, click **User** to view the permissions assigned to specific users. Click **Role** when viewing user permissions to toggle back.
2. For each user or role, you can select a new permission from the drop-down.
3. **Figure 3-6** shows the default user permissions on this folder. In the default installation, all permissions are defined by role; therefore, 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 gives the user joeuser the ability to see but not modify the Input Data Types folder and its contents. For all other end users, however, the folder is still execute-only due to the settings in **Figure 3-5**.

**Figure 3-5 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 cannot see the permission for the system admin (superuser) or for ROLE_SUPERUSER.

**Figure 3-5** shows the default role-based permissions on the sample Input Data Types folder as seen by the organization admin (jasperadmin). Members of certain roles can see and modify the input data types 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 cannot change 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.
6. **Figure 3-6** shows the default user permissions on this folder. In the default installation, all permissions are defined by role; therefore, 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 gives the user joeuser the ability to see but not modify the Input Data Types folder and its contents. For all other end users, however, the folder is still execute-only due to the settings in **Figure 3-5**.

**Figure 3-6 Permissions Dialog Showing Permissions by User**
6. Click **Apply** to save your changes. If you toggle between user and role permissions, you must click **Apply** first to save any changes you made.

7. Click OK to save your changes and close the permissions dialog when you are finished.

You can open several permissions dialogs for different resources or folders at the same time, as well as navigate 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 when 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.

### 3.5.7 Testing User Permissions

Once you have configured users, roles, and permissions, Jaspersoft recommends that you test the permissions granted to a few representative users. Testing is also recommended when you add new users, roles, and resources, and when you make any major modifications to your access control configuration.

**To test user permissions:**

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. In the repository, browse or search for the folders and resources to test.
7. Verify that JasperReports Server displays the expected folders and resources. Make a note of any objects that should be displayed but are not, and any objects that should be hidden but are displayed.
8. When you have verified the user’s permissions, click **Log Out**.

   Your own Home page appears.
9. To change the user’s permissions, edit the permissions in the repository and modify the user or role definitions.
10. Continue testing until the user’s permissions are satisfactory.
11. Repeat these steps with several representative users to ensure that your access control is properly configured. An access control configuration that hasn’t been tested doesn’t secure your data adequately.
CHAPTER 4 RESOURCES IN THE REPOSITORY

The repository in JasperReports Server stores the resources that can be combined to create a report from an uploaded JRXML file. The previous chapter introduced the repository and how to create folders and generic resources. This chapter goes into detail about how to create some key resources: data sources, queries, input controls, and file resources. These are the resources that users reference when uploading a JasperReport.

There are two scenarios for administering JasperReports Server, depending on how your end users create and consume reports.

- If you have users who are proficient at creating their own reports in iReport, they can upload them as JasperReports to the server. In this case, administrators must work with users to prepare the resources required by their reports.
- In the second scenario, administrators create and upload JasperReports to the server for their less technical users. Administrators still need to define all the resources for the reports that their users request.

This chapter contains the following sections:

- Data Sources
- Queries
- Input Controls
- File Resources

4.1 Data Sources

A data source is a resource in the repository that defines how and where to obtain the data displayed by reports or analysis views. Typically, it includes the location of the data and the details you need to access it, such as a user name and password. For example, to access an RDBMS (Relational Database Management System) that stores your data, you would create a data source that included the URI of the database server and user credentials.

Data sources are typically an RDBMS such as DB2, MySQL, Oracle, or PostgreSQL (to name but a few). 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. JasperReports Server configures and manages the connections to the database. By default, the maximum number of simultaneous connections for each data source is five.
- JNDI data source – Calls the JNDI service of the application server to locate a database connection. The application server manages the connections to the database and shares them between all applications that it hosts. The configuration of the application server determines the number of connections that are shared. Note that the application server connects to the database using JDBC, meaning that JNDI data sources return results in the same format as JDBC data sources.
JasperReports Server also supports big data in Apache Hadoop clusters accessed through Apache Hive. Because Hive uses the Hive Query Language (HiveQL) that is similar to SQL but distinct, it has its own data source type. In addition, JasperReports Server supports custom-made report data sources in the form of JavaBeans-based data sources. This section discusses JDBC, JNDI, Hadoop-Hive, and bean data sources.

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.

4.1.1 JDBC Data Sources

To access an RDBMS from JasperReports Server using JDBC you must have a driver, which must be accessible in the server’s classpath. The driver cannot be added on the fly from the user interface; it must already be in the classpath when the server starts before you can use it in a data source. When configuring a new JDBC data source that points to new types of database server, you should make sure that the JDBC driver you are using is available in the classpath of the JVM that launches the server.

If you need to add a new JDBC driver, it should be placed in the application server’s library folder, which is already in the JVM’s classpath. For example, if you use the Tomcat application server bundled with JasperReports server, place the driver file in <tomcat>/lib, which corresponds to <js-install>/<application-server-path>/lib. For example, if you installed the server from the installation executable on Windows, the path is C:\Program Files\jasperreports-server-x.x\apache-tomcat\lib. For more information on JDBC drivers, refer to the JasperReports Server Installation Guide.

To add a JDBC data source:
1. Log on as an administrator.
2. Click View > Repository and expand the folder tree to locate the folder in which you want to create the data source. If you installed the sample data, the suggested folder is Data Sources.
3. Right-click the folder and select Add Resource > Data Source from the context menu.
   The Add Data Source page appears.
4. In the Type field, select JDBC Data Source.
5. Fill in the other required fields, along with any optional information. The following table shows typical values for the driver and URL of common databases:

<table>
<thead>
<tr>
<th>Database</th>
<th>Driver and URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft SQL Server</td>
<td>com.microsoft.sqlserver.jdbc.SQLServerDriver</td>
</tr>
<tr>
<td></td>
<td>jdbc:sqlserver://&lt;host&gt;:1433;databaseName=&lt;DBname&gt;</td>
</tr>
<tr>
<td>Microsoft SQL Server with JTDS driver</td>
<td>net.sourceforge.jtds.jdbc.Driver</td>
</tr>
<tr>
<td></td>
<td>jdbc:jtds:sqlserver://&lt;host&gt;:1433/&lt;DBname&gt;</td>
</tr>
<tr>
<td>MySQL</td>
<td>com.mysql.jdbc.Driver</td>
</tr>
<tr>
<td></td>
<td>jdbc:mysql://&lt;host&gt;:3306/&lt;DBname&gt;</td>
</tr>
<tr>
<td>Oracle (thin driver)</td>
<td>oracle.jdbc.OracleDriver</td>
</tr>
<tr>
<td></td>
<td>jdbc:oracle:thin:@&lt;host&gt;:1521;&lt;DBinstance&gt;</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>org.postgresql.Driver</td>
</tr>
<tr>
<td></td>
<td>jdbc:postgresql://&lt;host&gt;:5432/&lt;DBname&gt;</td>
</tr>
</tbody>
</table>

The following figure shows values for connecting to the sample Foodmart database included in the sample data.

![Add Data Source](image)

**Figure 4-2  Creating a JDBC Data Source**
4.1.2 JNDI Data Sources

Adding a JNDI data source is very similar to adding a JDBC data source. The JNDI data source points to an existing connection that is defined in the application server and published as a JNDI resource or service. Instead of specifying a driver and database as you do with JDBC data sources, you only need to specify the JNDI service name in your application server.

To add a JNDI data source:
1. Log on as an administrator.
2. Click View > Repository and expand the folder tree to locate the folder in which you want to create the data source. If you installed the sample data, the suggested folder is Data Sources.
3. Right-click the folder and select Add Resource > Data Source from the context menu.
   The Add Data Source page appears.
4. In the Type field, select JNDI Data Source.
   The information on the page changes to reflect what’s needed to define a JNDI data source.
5. Fill in the required fields, along with any optional information.
   The service name is the name that the application server exposes through JNDI. The following figure shows values for connecting to the JNDI service for the Foodmart database included in the sample data.
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 A.4.5, “JNDI Services on Apache Tomcat,” on page 123.

7. When the test is successful, click **Submit**. The data source appears in the repository.

For details about configuring a JNDI database connection at the application server level and making it available to the server’s applications, refer to the documentation provided with your application server.

### 4.1.3 Hadoop-Hive Data Sources

Unlike traditional databases, Hadoop systems support huge amounts of data, often called big data. But this capability has a cost: high latency with access times on the order of 30 seconds up to 2 minutes. 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 OLAP connections.
- 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 section 7.5.1, “Ad Hoc Settings,” on page 96).
- You must configure your JVM memory to handle the expected data (see the JasperReports Server Installation Guide).
In general, reports based on Hadoop-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 immediately when you need it.

**To add a Hadoop-Hive data source:**

1. Log on as an administrator.
2. Click **View > Repository** and expand the folder tree to locate the folder in which you want to create the data source. If you installed the sample data, the suggested folder is Data Sources.
3. Right-click the folder and select **Add Resource > Data Source** from the context menu.
   The Add Data Source page appears.
4. In the Type field, select **Hadoop-Hive Data Source**.
   The information on the page changes to reflect what’s needed to define a Hadoop-Hive data source.

5. Fill in the required fields, along with any optional information.
   The Hive JDBC URL has the form: jdbc:hive://<hivehost>:10000/default
6. When done, click **Submit**. The data source appears in the repository.

**4.1.4 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.

Once the data source service bean is available through Spring, you can add the bean data source to the repository.

To add a bean data source:

1. Log on as an administrator.
2. Click View > Repository and expand the folder tree to locate the folder in which you want to create the data source. If you installed the sample data, the suggested folder is Data Sources.
3. Right-click the folder and select Add Resource > Data Source from the context menu. The Add Data Source page appears, as shown in Figure 4-5 on page 54.
4. In the Type field, select Bean Data Source. The information on the page changes to reflect what’s needed to define a bean data source.
5. Fill in the required fields, along with any optional information.
   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.
6. 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.
7. When the test is successful, click Submit.
4.2 Queries

JRXML reports use a query to select the data to be returned from the data source. The query can be defined in the JRXML itself, or it can be saved in the repository. A query in the repository can be re-used by multiple JasperReports. See the sample queries in the /SuperMart Demo/Common folder in the repository.

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. Reusing a query simplifies maintenance of reports, as well, since all the reports return the same data. 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.

See the JasperReports Server User Guide for complete instructions on using JRXML reports. For another means of adapting reports to different audiences, refer to the chapter on Domains in that manual.

Query resources can also be used to populate list input controls. For more information, see the chapter on cascading input controls in the JasperReports Server Ultimate Guide.

To create a reusable query resource:
1. Login as an administrator.
2. Click View > Repository and expand the folder tree to locate the folder in which you want to create the query. If you installed the sample data, the suggested folder is Input Data Types.
3. Right-click the folder and select Add Resource > Query from the context menu. The Add Query page appears.
4. Enter a name and optional description for the query and click **Next**. The resource ID is filled in automatically. The Link a Data Source page appears.

5. Select the data source and click **Next**. For this example, leave the default, but other options are presented:
   - **Do not link a data source.** If no data source is associated with the query, the server uses the data source that is associated with the report that references this query.
   - **Create a new data source.** You can define a local data source within this query resource that is not accessible to any other resource. This new data source overrides any data source specified in reports that use this query.
   - **Select data source from repository.** This creates a reference to a data source in the repository. The data source you select overrides any data source specified in reports that use this query.

After clicking **Next**, the Define the Query page appears.
6. Select **SQL** as the Query Language.
   The query language Domain (sl) is selected when opening Domain-based queries created in versions of the server prior to 3.7. It is used only for backward compatibility and should not be selected for new Domain-based queries.

7. Enter the following test in the **Query String** field:

```
SELECT * FROM orders
```

8. Click **Save**.

By default, JasperReports Server supports SQL, HQL (Hibernate), HiveQL (Hadoop-Hive) and Domain queries, while JasperReports supports several more (such as EJBQL, xPath and MDX). However, JasperReports Server can support queries in additional query languages if there is a properly-configured query executor implementation for each additional language when the server is deployed.

A specialized bean data source can be used to support multiple query languages. For information about bean data sources, refer to section 4.1.4, “Bean Data Sources,” on page 52. Another option is to add new types of data sources to the server, thus extending the reach of the JasperReports Server platform by leveraging one of its main extension points. Custom data sources are described in the **JasperReports Server Ultimate Guide**.

### 4.3 Input Controls

Any JasperReport can be parameterized so that its generated output is a function of values given at runtime (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 runtime 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 the list of discrete values that are 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 the above information to prompt the user to enter a value and validate the input. For example, consider a report that returns sales data for all of a company’s products; the user might input the name of 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** – Represented 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 Boolean.TRUE or Boolean.FALSE as values, depending on whether the box is checked.
- **Single value** – Represented 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** – Represented either as a drop-down list or a set of radio buttons. A single-select input control returns a single value.
- **Multi-select** – Represented as a list of values (scrollable) 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 the *JasperReports Server Ultimate Guide*.

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 section 3.2.3, “Local Resources and External References,” on page 34.

4.3.1 Datatypes

Datatypes are resources that define the format of a single-value input control, for example text or numerical value. The datatype determines what users can enter in the text field so that it corresponds to the type of the parameter in the report. Furthermore, the datatype can restrict the value, for example setting a range for a number or date, or enforcing a pattern such as an email address in strings. This is all controlled through the datatype associated with the input control.

Datatypes can represent one of four types:

- Text
- Number
- Date
- Date/time

**To create a datatype resource:**

1. Log on as an administrator.
2. Click **View > Repository** and expand the folder tree to locate the folder in which you want to create the datatype. If you installed the sample data, the suggested folder is Input Data Types.
3. Right-click the folder and select **Add Resource > Datatype** from the context menu.
   The Add Datatype page appears.

Ad Hoc reports 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, refer to 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” on page 61.
4. Enter a name and optional description for the datatype. The resource ID is filled in automatically.
5. Select the type of the datatype, as well as information related to the type.

   In this example, select **Text** as the type of our datatype; you have other options as well:
   - **Text** – For text datatypes, you can specify a regular expression in the **Pattern** field. The expression is used to validate the text that 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/Maximum is Strict** check boxes). If a **Strict** check box is selected, 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 Date/Time** – For these datatypes, there is a calendar widget in which you can select the desired minimum and maximum values and to make sure the configured date and date/time formats are used. To use the calendar, click the calendar icon.

6. When you have defined the properties of your datatype, click **Save**. The datatype resource appears in the repository.

We have created a very basic datatype for any type of text input.

### 4.3.2 Lists of Values

List of values are resources that define a static list of values for single-select or multi-select input controls. For each selection in the list, the list defines a label presented to the user and the value passed to the report when it runs. Depending on the type of input control, the end user selects one or more of these labels as radio buttons, check boxes, or drop-down lists.

**To create a list of values resource:**
1. Log on as an administrator.
2. Click **View > Repository** and expand the folder tree to locate the folder in which you want to create the list of values. If you installed the sample data, the suggested folder is Input Data Types.
3. Right-click the folder and select **Add Resource > List of Values** from the context menu.
   The Add List of Values page appears.

4. Enter a name and optional description for the datatype. The resource ID is filled in automatically.
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 only see the label in an input control that uses the list, and the report only receives the value. To remove an item, click Remove beside its value.
6. When you have defined all values in the list, click **Submit**. The list of values resource appears in the repository.

### 4.3.3 Creating an Input Control

The input control resource determines how the input control functions and appears. As with other resources, input controls can be created locally as part of a JasperReport, in which case they cannot be seen outside of the 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 if there is a mismatch in the names, but at runtime, **NULL** is passed in as the value of the parameter instead of actual values.
- 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 resource in the repository. To reference input controls in a JasperReport, see the *JasperReports Server User Guide*.

**To create an input control resource:**

1. Log on as an administrator.
2. Click **View > Repository** and expand the folder tree to locate the folder in which you want to create the input control. If you installed the sample data, the suggested folder is Input Data Types.
3. Right-click the folder and select **Add Resource > Input Control** from the context menu.
   The Add Input Control page appears.
4. Select the type of input control from the **Type** list.
   In this example, select **Single Value**.

5. Enter the prompt to display to users explaining how to use the control.
   For this example, use 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 from the prompt text. If you have used a different prompt, select the automatic name and replace it with the exact name of the parameter associated with the control. Remember, the parameter name must be the same as in the reports that use this input control.
   For this example, the parameter name is **title**. The description is optional.

7. Select options for the control.
   In this example, select **Mandatory** and **Visible**; you have other options as well:
   - **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 above:
   - Boolean types do not require any further information.
   - Single-value types require a datatype resource to characterize what the user may enter.
   - Single-select and multi-select types based on static lists require a list of values resource.
   - Single-select and multi-select types based on queries require a query resource.

9. In this single-value example, the Locate Datatype page appears. Choose the option to select a datatype from the repository and click Browse. From the repository dialog that appears, select `/datatypes/TextGeneralDatatype`, which is similar to the datatype we created in section 4.3.1, “Datatypes,” on page 57.

   If you choose to define a datatype, the wizard takes you through the same procedure as in section 4.3.1, “Datatypes,” on page 57. You can then define any datatype you need, but it is local to the input control and not reusable in other input controls.
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 step 5 is also used as the name for the resource.

### 4.4 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 that exist in your data. Because the queries use standard syntax, you can include filters in a WHERE clause. In the previous example, you could restrict the list of cities to a certain country.

By including parameters, you can also create cascading input controls. A cascading input control is one whose choices depend on the selection of a previous input control. For example, after the user selects a country, the available city values are restricted to the chosen country. Cascading input controls are query-based controls that contain parameters returned by other controls.

Cascading input controls help make input controls easier to use and faster to display. Certain parameters in reports have a natural hierarchy, such as countries and cities or years and quarters, and the cascading input controls let the user find values based on this hierarchy. Instead of selecting cities from one large list that may need to scroll, users can make a selection from a smaller list where all choices are visible. Also, displaying long lists make the web page slow to load, so cascading input controls that reduce the size of the list make it faster to load. If there were an especially large number of cities, more cascading input controls could be used to reduce the list, such as region or state. The values for each control are loaded only when the previous input has been selected, making for a convenient and speedy user experience.

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.

Creating standard input controls is documented in the *JasperReports Server Administrator Guide*. This chapter explains how to create input controls that are based on a query and parameterized by the results of another control, thus creating the cascading effect.

This chapter contains the following sections:
- Creating a Query-based Input Control
- Built-in Parameters for Query-based Input Controls
- Domain-based Queries
- Cascading Input Controls
4.4.1 Creating a Query-based Input Control

In this first example, we create a query-based input control that returns a long list of all cities for the user to choose from.

1. Log in as an administrator.
2. Browse the repository and select the folder where you want to create the query-based input control.
3. Right-click the folder and select Add Resource > Input Control. The Add Input Control dialog appears:

![Figure 4-13 Adding an Input Control - Naming](image)

4. Select the type of query-based input control from the type drop-down list. This choice 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.
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 4-14 Adding an Input Control - Locating the Query](image)
If you have a suitable query resource defined in the repository, you could select it here as an external reference. In this example, we’ll define a query resource locally inside the input control resource.

7. Click **Next** to define the local query resource. The query naming dialog appears:

![Figure 4-15 Adding an Input Control - Naming the Query](image)

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 data source link page appears:

![Figure 4-16 Adding an Input Control - Linking to a Data Source](image)

As with all query resources, the query resource 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 report. In this example, we take the default selection of not linking to a data source.
9. Click **Next**. The query definition page appears:

![Figure 4-17 Adding an Input Control - Defining the Query](image)

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`, and the `country` field is used to limit 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 section 4.4.3, “Domain-based Queries,” on page 66.

11. Click **Save** to complete the query definition. The parameter values page appears:

![Figure 4-18 Adding an Input Control - Setting Parameter Values](image)

On the parameter values page, you define which field in the results of the query are displayed, and which field contains values that become the parameter value when chosen.

a. First, specify the value column, which is the field whose value is passed to the report. The data type of the field must match the type of the corresponding parameter in the report.

b. Next, specify the visible columns, which are the fields whose values appear in the input control that the user chooses from. In the simplest case, enter 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 the example in Figure 4-18, the user may 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 field 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.

4.4.2 Built-in Parameters for Query-based Input Controls

The `LoggedInUser` and `LoggedInUsername` parameters are always available for query input controls; they are 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 are defined as parameters in the JRXML.

Table 4-1 Built-in Parameters for Query-based Input Controls

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoggedInUser</td>
<td>User</td>
<td>The user that is currently logged in. This parameter isn’t available in query input controls, but is used as parameter to the report.</td>
</tr>
<tr>
<td>LoggedInUsername</td>
<td>String</td>
<td>The user name of the current user.</td>
</tr>
<tr>
<td>LoggedInUserFullName</td>
<td>String</td>
<td>The full name of the current user.</td>
</tr>
<tr>
<td>LoggedInUserEmailAddress</td>
<td>String</td>
<td>The email address of the current user.</td>
</tr>
<tr>
<td>LoggedInUserEnabled</td>
<td>Boolean</td>
<td>Indicates whether the current user is enabled.</td>
</tr>
<tr>
<td>LoggedInUserExternallyDefined</td>
<td>Boolean</td>
<td>Indicates whether the current user is authenticated externally.</td>
</tr>
<tr>
<td>LoggedInUserTenantId</td>
<td>String</td>
<td>In the commercial editions, the name of the organization of the current user.</td>
</tr>
<tr>
<td>LoggedInUserRoles</td>
<td>Collection&lt;String&gt;</td>
<td>The roles assigned to the current user. This is helpful for parameters that use $X.</td>
</tr>
<tr>
<td>LoggedInUserAttributes</td>
<td>Map&lt;String, String&gt;</td>
<td>The profile attributes of the logged-in user. This parameter isn’t usable in query input control, but it is used as parameter to the report. If the user has no attributes, the parameter is an empty map.</td>
</tr>
<tr>
<td>LoggedInUserAttributeName</td>
<td>Collection&lt;String&gt;</td>
<td>The names of the profile 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.</td>
</tr>
<tr>
<td>LoggedInUserAttributeValue</td>
<td>Collection&lt;String&gt;</td>
<td>The values of the profile 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.</td>
</tr>
<tr>
<td>LoggedInUserAttribute_&lt;attribute-name&gt;</td>
<td>String</td>
<td>For the logged-in user, the value of the attribute matching the name passed as &lt;attribute-name&gt; (like att1). If there is no match, the parameter is empty. This parameter is only available if it is defined in a query or as a report parameter.</td>
</tr>
</tbody>
</table>
4.4.3 Domain-based Queries

In the case of reports that use a Domain as the data source, any query-based input controls must contain a query against the Domain. When defining the query as shown in Figure 4-17 on page 64, 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 is used only for backward compatibility and should not be selected for new Domain-based queries.

Domain queries have their own special syntax, the same that is 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 store cities, where the field with ID \texttt{ej\_store\_store\_city} is nested in the set with ID \texttt{expense\_join\_store}, you would use the following Domain query:

```xml
<query>
  <queryFields>
    <queryField id="expense_join_store.ej_store_store_city" />
  </queryFields>
</query>
```

The list contained inside the \texttt{<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:

```xml
<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 Figure 4-21, “The COUNTRY Input Control,” on page 70, you would 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:

- \texttt{expense\_join\_store.ej\_store\_store\_country}
- \texttt{expense\_join\_store.ej\_store\_store\_state}
- \texttt{expense\_join\_store.ej\_store\_store\_city}

creates a list of values such as the following for users to choose from (the separator \texttt{|} is added automatically):

- USA | CA | Los Angeles
- USA | CA | San Francisco
- USA | OR | Portland
- USA | WA | Redmond

The query language Domain ("sl") is selected when opening Domain-based queries created in JasperServer 3.5 or earlier. It is used only for backward compatibility and should not be selected for new Domain-based queries.
Finally, the Domain-based query also has the option to filter the query results, as shown in the following example:

```xml
<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 User Guide*. 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.

### 4.4.4 Cascading Input Controls

A cascading input control is one whose values depend on the selection made in a previous input control. Cascading input controls are created by using parameters in the query string of a related input control. In other words, the parameter defined by an input control may be used in another input control.

In the 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 would have two input controls, one for the state and one for the city:

- 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 for that state. The cities are displayed in the input control, and when the user selects one and submits it, the city name is passed as a parameter to the report.

The user makes two selections from much shorter lists, which is easier and quicker than using one long list of city and state names. The second input control is empty, showing no selections, until clicking on the first of the cascading input controls. If the user selects a different state in the first control, the list of cities in the second control updates accordingly.

Parameter substitution in query input controls follows the same approach as for JasperReports queries. Queries of all types of query connections can use parameter substitution, and $P, $P! and $X (for SQL queries) parameters are supported. For more information on using $P, $P! and $X to build dynamic queries, refer to the *JasperReports Ultimate Guide* and the *iReport Ultimate Guide*.

In almost all cases, the parameters appear in the filter (WHERE) clause of the query. Single-select query input controls return single values that are referenced with the $P syntax, and multi-select query input controls return collections that are handled by the $X syntax.

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.
4.4.4.1 Parameters in Input Control Queries

The example in this section shows how to create cascading input controls for selecting a country and a city. This is done by writing the query of the second input control (city) with a syntax that references a parameter name. The syntax is the same used by JasperReports in the report queries. A parameter is referenced using the following convention:

$P{parameter name}

So if we have an input control called COUNTRY, the query to get the cities from a hypothetical table called ACCOUNTS looks like this:

```sql
select city from ACCOUNTS where country = $P{COUNTRY}
```

When the user selects a country from the COUNTRY input control, the result is used to perform the query of the CITY input control, and the CITY input control is refreshed to show the result.

There are two additional ways to use a parameter in a cascading input control query. The first is used when the value held by parameter_name is not a simple value, but a chunk of the query (or in extreme cases even the whole query). It has the syntax:

$P!{parameter_name}

With the $P!{} syntax, the value of the parameter is treated as raw text. The server replaces the placeholder with the value of the referenced parameter without performing extra checking and value escaping, as is done when using the plain $P{} syntax.

Secondly, there is the $X{} syntax that is used when the value of a parameter is a collection. 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. Now, the cities are selected in a multi-select input control that returns a collection, and the $P{} syntax is insufficient for substituting a collection into an SQL query. The $X{} syntax appears as follows:

```sql
select city from ACCOUNTS where $X{IN, country, COUNTRIES}
```

When 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:

```sql
select city from ACCOUNTS
where country IN ('USA', 'Canada', 'Mexico')
```

$X{} takes three positional arguments:
- First is the collection operator, either IN or NOT IN.
- Second is the table column that is being compared.
- Third is the parameter that provides the collection of values, in other words, the name a multi-select input control.

The number of parameters that can be used in a query is arbitrary, just as the number of input controls that can be defined in a JasperReport is arbitrary. In addition to the standard input control parameters, a cascading input control query can use the built-in parameters described in Table 4-1, “Built-in Parameters for Query-based Input Controls,” on page 65.

4.4.4.2 Step-by-Step Example

In this example, we’ll create a simple report that displays all the accounts of a city, using the SugarCRM sample database shipped with JasperReports Server. This example uses iReport to create a report and the JasperReports Server plug-in to create the import control resources in the repository.

We start by creating a report with a parameter called CITY and the following report query:

```sql
select * from accounts where billing_address_city = $P{CITY}
```

In the detail band, we add three fields: name, shipping_address_city, and shipping_address_country. Then we publish the report on the server, using the Publish tool of the JasperReports Server plug-in in iReport.
Now define the input controls. Right-click the JasperReport node in the Repository Navigator and add the first input control by selecting Add > Input Control. This input control shows the list of countries in which accounts are present. It is not a cascading input control, but its value is used in the next control: the one that selects the city.

Set the name of this first input control to COUNTRY (the display name can be “Country”). Set the Input Control type to Single Select Query (this because we want to get the countries using an SQL query, but since this is just a common input control, we may use any other type of input control, like a list of values or even a multiple-select list of values).

Edit a local resource for the query, set a name for it (“query”) and set the query language to SQL.
The query is just a simple query to select the countries. For instance:

```sql
select distinct shipping_address_country from ACCOUNTS
order by shipping_address_country
```

To complete the local query resource, set the repository resource /datasources/JServerJdbcDS as the query’s data source. Finally, in the **Value and Visible Columns** tab, set the Value Column to `shipping_address_country` and make it (the only) visible column. The first input control, which selects the country, is now ready.

Now that we have an input control named COUNTRY, we can reference the COUNTRY parameter in any query-based input control. This is what we are going to do with the second input control. Set its name to CITY. Its definition is similar to the COUNTRY control, so the type must be **Single Select Query**. The query resource must be of type **SQL**; it’s used in the COUNTRY parameter’s `where` condition:

```sql
select distinct shipping_address_city from ACCOUNTS
where shipping_address_country = $P{COUNTRY}
order by shipping_address_city
```

This time the column to be used in the Value and Visible Columns field is `shipping_address_city`.

When you run the JasperReport, if everything has been correctly configured, the dialog box in **Figure 4-22 on page 70** appears. It consists of the two simple input controls, and the CITY control is not populated until the user selects a country.
4.5 File Resources

File resources are those that the administrator creates by uploading a file. Like other resources in this chapter, file resources are created by administrators so that they can be referenced by Jasper Reports. JasperReports server supports the following files:

Table 4-2 File Resource Types

<table>
<thead>
<tr>
<th>File Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSS</td>
<td>Cascading Style Sheet file that helps define the user interface as part of a theme.</td>
</tr>
<tr>
<td>Font</td>
<td>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 4.5.1, “Fonts,” on page 71).</td>
</tr>
<tr>
<td>Image</td>
<td>Any image format supported by the JVM (Java Virtual Machine), such as JPEG, GIF, and PNG. Images can be referenced in JasperReports, and also in CSS files.</td>
</tr>
<tr>
<td>JAR</td>
<td>Libraries that provide functionality for your reports (see section 4.5.2, “JAR Files,” on page 71).</td>
</tr>
<tr>
<td>JRXML</td>
<td>The definition of a report in JasperReports’ XML-based report definition language. A JRXML file can be uploaded separately for use in multiple JasperReports.</td>
</tr>
<tr>
<td>OLAP Schema</td>
<td>Defines the data in an OLAP cube, including how to aggregate the dimensions.</td>
</tr>
<tr>
<td>Resource Bundle</td>
<td>A Java .properties file containing key-value pairs for localization of reports (see section 4.5.3, “Resource Bundles,” on page 72)</td>
</tr>
<tr>
<td>Style Template</td>
<td>A JRTX file containing a style template that can be shared between JasperReports.</td>
</tr>
<tr>
<td>XML</td>
<td>XML file used in Domains and analysis to define data-level security.</td>
</tr>
</tbody>
</table>

The way in which fonts, JAR files and resource bundles are associated with reports is further explained in the following sections.

4.5.1 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 fonts that are 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 runtime. 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.

For details about working with fonts and PDF export, refer to the JasperReports documentation.

4.5.2 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 is 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.
4.5.3 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.

4.5.4 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:

1. Log in as an administrator.
2. Select View > Repository and locate the parent folder of the new resource in the left-hand Folders panel.
3. In the Folders panel, right-click the parent folder and select Add Resource > File from the context menu, then select the type of resource to add. In this example, select Add Resource > File > Font. The Add File dialog appears.
4. 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 or click Browse to locate a file on your file system.

Figure 4-23 shows the dialog for adding a Font file. All file resources are created by uploading a file in a similar fashion.

5. When done, click Submit. The new file resource appears in the selected folder in the Repository panel. A message confirming the addition also appears at the top of the page.
4.5.5 Editing a File Resource

The following example shows how to edit a file resource.

**To edit a file resource:**

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. The Edit File dialog appears much like the Add File dialog except for some properties that cannot be changed. In this example, we edit the font resource created in section 4.5.4, “Creating a File Resource,” on page 72.

![Edit File dialog](image)

*Figure 4-24 Editing a File Resource*

4. Use the Edit dialog to view or modify the resource definition and its values. **Figure 4-24** shows 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.
CHAPTER 5 THEMES

JasperReports Server 4.0 introduced a new look and feel, and more importantly, a complete mechanism to easily modify the appearance of the user interface (UI). This mechanism called themes is based on Cascading Style Sheets (CSS), the web standard for defining the appearance of HTML content.

The set of all CSS files and associated images that defines the appearance of the user interface is a theme. Themes are stored as file resources in folders 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 administrators can switch between them, providing an easy and quick way to change the user interface.

For deployments licensed to use organizations, administrators can set the theme individually on each organization, or rely on theme inheritance to use the same theme everywhere without needing to set it explicitly. The inheritance mechanism also 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.

This chapter contains the following sections:
- Introduction
- How Themes Work
- Administering Themes
- Working With CSS Files

5.1 Introduction

The new design of the JasperReports Server user interface is shown in figures throughout this book. A major feature of this redesign is the new themes architecture that supports it. The default appearance can easily be modified to suit your needs.

The JasperReports Server user interface is based on CSS (Cascading Style Sheets) files that define the styles of the elements appearing in the HTML, itself defined in and generated from JSP (JavaServer Pages) and JavaScript. A theme is a collection of CSS files and associated images that specify the appearance for all or part of the user interface. A theme only controls how the interface appears, for example fonts, colors, spacing, lines, and image elements of the UI. It does not control what appears, such as the contents of menus or the effect of clicking a button.

Themes are defined globally at the repository root and individually on every organization. Every user of a given organization sees the same theme, but different organizations can have their own themes. 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 provides special actions on theme folders for downloading and uploading themes as ZIP (compressed archive) files, and for activating the theme.
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.

The themes mechanism is 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 the new themes mechanism:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the default theme unchanged.</td>
<td>The new UI has been designed for clarity and ease-of-use. After a standard installation, the default theme is set at the root level and is automatically inherited by all organizations so that every user sees the server with this interface. If the default theme suits your needs, there is no need to customize it or develop new themes.</td>
</tr>
<tr>
<td>Quickly modify the default theme.</td>
<td>You can specify overrides of individual CSS rules or replace images in the system theme. It is easy to create or upload the new files and activate your customizations. The inheritance mechanism ensures that every organization uses this new theme, and be updated in real time when you modify it.</td>
</tr>
<tr>
<td>Create an entirely new theme.</td>
<td>With CSS experience or Jaspersoft Professional Services, you can change the entire look and feel of the server. The server UI can be tailored to match or blend in with nearly any other web design. The inheritance mechanism again ensures that every organization uses the new theme, while allowing you to manage the interface from a single set of files.</td>
</tr>
<tr>
<td>Override themes to customize the UI for every organization.</td>
<td>You can give each organization or suborganization a customization of the default theme, for example a new logo, while retaining all other aspects of the system theme. The benefit of this approach is that the system theme can still be modified and inherited by all organizations, while still retaining the organization-level overrides. This approach can be combined with the previous one, so that overrides are applied to the custom theme.</td>
</tr>
<tr>
<td>Create a new theme in every organization.</td>
<td>For SaaS vendors, each organization can be a different client that needs a special interface. The themes mechanism allows each organization to fully define the UI and still retain the override and inheritance mechanism for its own suborganizations. In such deployments, each organization admin can modify or create the appearance of his own user interface.</td>
</tr>
</tbody>
</table>

It is important to realize that a theme refers to two concepts simultaneously:
- A folder containing a set of CSS files and image files in the proper location in the repository.
- The entire appearance of the user interface after activating the theme’s files. However, through the inheritance mechanism, parts of the interface are defined in the files belonging to other themes. In fact, except for the default theme, the entire user interface is rarely defined in a single theme.

For example, a very simple theme named MyLogo contains a custom image file to replace the Jaspersoft logo, and nothing else. The rest of the interface is inherited from the default theme or some custom theme. Yet we say that MyLogo is the active theme, and every user in the same organization sees the MyLogo theme.

### 5.2 How Themes Work

Themes are stored in a special folder named Themes that appear at the root of the repository and in every organization. Each Themes folder contains a default theme that cannot 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 organization Themes, 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.

This chapter uses the following terminology to distinguish between root-level and organization-level themes. In the following table, the main folder of any organization is named Organization, and \textit{active-theme} is the name of the theme folder that has been activated:

<table>
<thead>
<tr>
<th>Name</th>
<th>Folder</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default theme</td>
<td>root &gt; Themes &gt; default</td>
<td>The unmodified user interface of JasperReports Server, as it appears at first installation. The default theme is defined in the default folder in the Themes folder at the root of the repository.</td>
</tr>
<tr>
<td>System theme</td>
<td>root &gt; Themes &gt; \textit{active-theme}</td>
<td>The active theme set at the root level. All users in all organizations see this theme unless there is an organization-specific theme that is activated and overrides some styles. The system theme is also used for the login page.</td>
</tr>
<tr>
<td>Inherited theme</td>
<td>Organization &gt; Themes &gt; default</td>
<td>The combination of all active themes in the parent organizations of a given organization, according to the inheritance rules. For any given organization, the theme inherited by that organization is stored in the default folder of that organization’s Themes folder.</td>
</tr>
<tr>
<td>Active theme</td>
<td>Organization &gt; Themes &gt; \textit{active-theme}</td>
<td>The theme that users of a given organization can see. administrator has set as 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.</td>
</tr>
</tbody>
</table>

When JasperReports Server is first installed, the default theme is active, so it is also the system theme. Also, in every organization including the default organization, the inherited theme is a copy of the default theme, and it is active by default for that organization.

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 such that the UI becomes unusable. In this situation you must re-install JasperReports Server to recover.

The following figure shows the default theme in the Themes folder at the root of the repository. The name of the folder (and its subfolders) are bold to indicate that it is the active theme.
5.2.1 Theme Files

A complete theme consists of exactly the following files, along with all referenced images:

1. theme.css  
2. pages.css  
3. containers.css  
4. buttons.css  
5. lists.css  
6. controls.css  
7. dataDisplays.css  
8. pageSpecific.css  
9. dialogSpecific.css  
10. overrides_ie7.css  
11. overrides_ie8.css  
12. overrides_custom.css

In addition, the default theme contains the file samples.css that is only used by the View > Samples page described in section 5.4.5, “User Interface Samples,” on page 87. 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 default theme stores referenced image files in a folder named images. There are approximately 80 image files in the default theme. There are two ways to change an image from the default theme:

- Use a folder named images and image files with the same name as the ones you want to replace.
- Modify the corresponding CSS rules to redefine the location where they can be found.

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 structure 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.
5.2.2 Inheritance Mechanism

In order to render the user interface, Jasper Reports Server must load each of the theme files. Because each file can be stored in multiple themes, the inheritance mechanism determines which file to load.

The server loads each of the CSS files listed in section 5.2.1, “Theme Files,” on page 78 in the order they are listed. To locate the file, the server looks in the following locations, in the following order:

1. The active theme folder for the user’s organization.
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 the inheritance mechanism.

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. 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.

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.

5.2.3 CSS Priority Scheme and Custom Overrides

Once the inheritance mechanism 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. 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 inheritance mechanism uses the new files 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.

Jaspersoft recommends 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 the file-based inheritance mechanism. 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 can copy a file that defines gray buttons with plain text, and you change the CSS rule to make the text bold, to create 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.

5.3 Administering Themes

Themes are sets of CSS and image files stored in a folder in the repository. The root of the repository and every organization 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, and for creating and modifying new theme folders. For information about how to work with CSS in themes, see section 5.4, “Working With CSS Files,” on page 85.

5.3.1 Setting the System Theme

1. Log into JasperReports Server as system administrator (superuser).
2. Click View > Repository and expand the Themes folder if necessary.
3. Right-click the new theme folder and select Set as Active Theme.

For example, the sample data includes a second theme called pods_summer that you can set as active.

As soon as the screen is refreshed, you see the effect of the new theme. Notice how the pods_summer theme changes the colors and the logo in the user interface with just the overrides_custom.css file and images.
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. Also, the system theme set here applies to the login page, as shown in the following figure.

The following procedures assume that the system theme is still set to the default theme.

5.3.2 Setting an Organization Theme

1. Log into JasperReports server as the organization admin (jasperadmin).
   In a server licensed to use multiple organizations, specify the organization ID or alias on the login page.
2. Click **View > Repository** and expand the Themes folder if necessary. The organization’s Theme folder is shown in Figure 5-5.

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.

![Figure 5-5 Organization Themes Seen by Organization Admin](image)

### 5.3.3 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 only applies to system admins. Organization admins cannot modify the **ROLE_ADMINISTRATOR** permission, even in suborganizations. They must request that the system admin perform the procedure for them.

1. Log into JasperReports Server as system administrator (**superuser**).
2. Click **View > Repository** and expand the Organizations folder.
3. Locate the name of the organization where you want to restrict access to themes and expand its folder.
4. Right-click the organization’s Themes folder name and select **Permissions**.
5. Change the permission for the ROLE_ADMINISTRATOR from Administer to Execute Only.
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.

6. To restrict access to all organizations, repeat step 3 to step 5 for every organization in the server, including suborganizations.

7. If you want to restrict access in the same way in all future organizations, repeat step 4 and step 5 in the Folder Template of every organization and suborganization in the server. For more information, see section 5.3.4.3, “Placing Themes in the Folder Template,” on page 85.

## 5.3.4 Creating Theme Folders and Files

There are three ways to create the folders and files that make up a theme:

- Create them directly as resources in the repository.
- Download and upload themes as ZIP (archive) files.
- In multi-organization deployments, placing theme folders in the Folder Template.

This section explains only how to store CSS files in the repository. For information about creating CSS file contents, see section 5.4, “Working With CSS Files,” on page 85.

### 5.3.4.1 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:

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 listed in section 5.2.1, “Theme Files,” on page 78.

5. To add images to your theme, create any image folders and upload image files with Add Resource > File > Image.

6. Repeat step 4 and step 5 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.

### 5.3.4.2 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:**

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 section 5.2.3, “CSS Priority Scheme and Custom Overrides,” on page 79 for an explanation of how to create a theme.

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:**

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.]

1. Log in as an administrator with access to the location where you want to upload the theme.
2. Click View > Repository and expand the Themes folder if necessary.
3. Right-click the Themes folder and select Upload a Theme.
4. In the dialog that appears, enter a name for your theme, which becomes the name of its folder, and browse to find the ZIP file on your computer. Click **Upload**.

   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 chosen Themes 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 section 5.3.4.1, “Creating Theme Folders and File Resources,” on page 83. 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.

### 5.3.4.3 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 theme folder and all of its contents are copied to the Themes folder of any new organizations that are created. 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 contains a Themes folder and an empty default 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 section 5.3.3, “Restricting Access to Themes,” on page 82.

### 5.4 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. Additionally, 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.
5.4.1 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 5.2.3, “CSS Priority Scheme and Custom Overrides,” on page 79. 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 principal steps in a theme development workflow are as follows:

<table>
<thead>
<tr>
<th>Step</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Download the default theme so you have a copy of the files and CSS rules that you want to modify.</td>
<td>Section 5.3.4.2, “Downloading and Uploading Theme ZIP Files,” on page 84.</td>
</tr>
<tr>
<td>3. Upload your new files to a test platform, and activate the theme or place them in an active theme.</td>
<td>Section 5.3.4.1, “Creating Theme Folders and File Resources,” on page 83.</td>
</tr>
<tr>
<td>4. Verify your changes wherever they occur in the UI.</td>
<td>Sections 5.4.3, “Test Platform,” on page 86 and 5.4.5, “User Interface Samples,” on page 87.</td>
</tr>
<tr>
<td>5. Repeat step 2 through step 4 for all your changes until the theme is finalized.</td>
<td></td>
</tr>
<tr>
<td>6. Deploy your theme to your users.</td>
<td>Section 5.3.1, “Setting the System Theme,” on page 80 or 5.3.2, “Setting an Organization Theme,” on page 81.</td>
</tr>
</tbody>
</table>

5.4.2 Firebug Plug-in for Firefox

One tool to help you find, modify, and view CSS rules in step 2 above is the Firebug plug-in for the Mozilla Firefox browser. Firebug displays the HTML, JavaScript, 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 and immediately see the effect on the web page.

The Firebug tool is 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 Firebug frame directly into the overrides_custom.css file. Firebug displays the entire rule from its original file, so the copy overrides it exactly. If you are modifying other files from the default theme, Firebug show 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, Firebug 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.

For more information and downloads, see the Firebug website.

5.4.3 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.

On all of these test platforms, you should 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 also use Firebug to look at the CSS rules that are involved, even if the errors do not show up on Firefox.

### 5.4.4 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 `analysisViewID` 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; }
```

### 5.4.5 User Interface Samples

When testing your theme, you should 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. An additional test is to look at the user interface samples with the theme you are developing.

The user interface sample page is a new page included with the redesign of the interface. It is only accessible to administrators:

1. Log in as administrator in your test environment at the level where you want to test your theme (superuser or jasperadmin of an organization or suborganization).
2. If you haven’t already done so, upload your theme to the Themes folder at this level. See section 5.3.4, “Creating Theme Folders and Files,” on page 83.
3. Select View > Samples from the main menu on any page.
4. Look at all the sample components in each of the sample galleries. For example, the buttons gallery shows all the different types of buttons in every possible action state.
5. When you click on the standard layouts, the sample replaces the samples page. Select View > Samples from the main menu again to return to the galleries.

The Samples page relies on an extra CSS file that is not required in a theme, but that can be included. The file samples.css is located in the default theme in the system-level Themes folder. If the sample elements do not appear as you expect, add this file to your theme and customize its rules as necessary. The rules in this file are not used anywhere else in the user interface, so it should not be included in your final theme.

Viewing the sample galleries can help you quickly find errors in your theme, especially if you are changing many rules and replacing entire files in your theme. Using these samples along with the testing procedures and tools described previously, you can verify that your theme properly implements the custom user interface that you intend. Having a well-tested theme minimizes the chances of errors when you activate the theme in your production server.
CHAPTER 6  IMPORT/EXPORT

The import and export utilities enable you to extract resources from, or add resources to, a JasperReports Server repository. The 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 or when moving between test and production environments.

This chapter contains the following sections:
- Running Import and Export
- Exporting Repository Resources
- Importing Repository Resources

6.1  Running Import and Export

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 in Linux, be sure to add the .sh file extension.

The output of the export command, and input to the import command, is called a catalog. It is a set of folders and files that represent the contents of the server’s internal database, including organizations, users, roles, scheduled jobs, and repository resources. The catalog can be exported either as a hierarchy of folders and files, or as a single zip file containing the same information.

Regardless of the catalog format, the contents of the catalog are 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 files is not publicly defined, and the data files aren’t meant to be accessed.

When using the import and export utilities, keep in mind the following:
- All command line options start with two dashes (--).
- You must specify either a directory or a zip file to export to or import from.
- Make sure the output location specified for an export is writable to the user running the command.
- URIs are repository paths originating at the root. For example, the URI of the report folder in the default organization is:
  /organizations/organization_1/reports
- Passwords are encrypted in the server’s internal database, but not in exported catalogs. Take appropriate measures to secure the catalog file from unauthorized access if you export users.

6.2 Exporting Repository Resources

Usage: js-export [OPTIONS]

Specifies repository resources such as reports, images, folders, and scheduled jobs to export to the file system. You can also export the internal definitions for scheduled jobs, users, roles, as well existing audit data. The export output is known as a repository catalog; it is either an archive file or a set of files in a folder structure:

<table>
<thead>
<tr>
<th>Option</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>--everything</td>
<td>Export everything except audit data: all repository resources, permissions, report jobs, users, and roles. This option is equivalent to:</td>
</tr>
<tr>
<td></td>
<td>--uris --repository-permissions --report-jobs --users --roles</td>
</tr>
<tr>
<td>--help</td>
<td>Displays brief information about the available options.</td>
</tr>
<tr>
<td>--include-access-events</td>
<td>Access events (date, time, and user name of last modification) are exported.</td>
</tr>
<tr>
<td>--output-dir</td>
<td>Path of a directory in which to create the output catalog folder.</td>
</tr>
<tr>
<td>--output-zip</td>
<td>Path and filename of the output catalog zip file to create.</td>
</tr>
<tr>
<td>--report-jobs</td>
<td>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 recursively in all subfolders.</td>
</tr>
<tr>
<td>--repository-permissions</td>
<td>When this option is present, repository permissions are exported along with each exported folder and resource. This option should only be used in conjunction with --uris.</td>
</tr>
<tr>
<td>--roles</td>
<td>Comma separated list of roles to export; if no roles are specified with this option, all roles are exported.</td>
</tr>
<tr>
<td>--role-users</td>
<td>When this option is present, each role export triggers the export of all users belonging to the role. This option should only be used in conjunction with --roles.</td>
</tr>
<tr>
<td>--uris</td>
<td>Comma separated list of folder or resource URIs in the repository.</td>
</tr>
<tr>
<td>--users</td>
<td>Comma separated list of users to export; if no users are specified with this options, all users are exported. When specifying users, you must give their organization ID, for example: --users superuser, &quot;jasperadmin</td>
</tr>
<tr>
<td>--include-audit-events</td>
<td>Include audit data for all resources and users in the export.</td>
</tr>
</tbody>
</table>

Examples:
- Export everything in the repository:
  js-export --everything --output-dir myExport
- Export the /reports/samples/AllAccounts report unit to a catalog folder:
  js-export --uris /organizations/organization_1/reports/samples/AllAccounts --output-dir myExport
• Export the /images and /fonts folders:
  
  ```bash
  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:

  ```bash
  js-export --uris / --repository-permissions --output-zip myExport.zip
  ```

• Export all resources and report jobs:

  ```bash
  js-export --uris / --report-jobs / --output-dir myExport
  ```

• Export the report jobs of the /reports/samples/AllAccounts report unit:

  ```bash
  js-export --report-jobs /organizations/organization_1/reports/samples/AllAccounts --output-dir myExport
  ```

• Export all resources and report jobs:

  ```bash
  js-export --uris / --report-jobs / --output-dir myExport
  ```

• Export the ROLE_USER and ROLE_ADMINISTRATOR roles along with all users belonging to either role:

  ```bash
  js-export --roles ROLE_USER, ROLE_ADMINISTRATOR --role-users --output-dir myExport
  ```

The `--uris` option allows you to specify one or more resource URIs. A URI can specify a resource such as a report. In this case, all associated resources (such as images, subreports, data sources, resource bundles, and class files) are exported. A URI can also specify a folder. If a folder is specified, the export operation exports all resources and folders contained in the folder. In addition, it recurses through all its subfolders.

When you export a user, the user’s roles and properties are exported in addition to the information specified in the export command.

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.

### 6.3 Importing Repository Resources

Usage: `js-import [OPTIONS]`

Reads a repository catalog from the your file system and creates the named resource in the JasperReports Server repository. The repository catalog must have been created with the `js-export` command, either as a ZIP archive file or a directory:

#### Table 6-2 Options in `js-import` Command

<table>
<thead>
<tr>
<th>Option</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>--help</td>
<td>Displays brief information about the available options.</td>
</tr>
<tr>
<td>--input-dir</td>
<td>Path for importing a catalog from a directory.</td>
</tr>
<tr>
<td>--input-zip</td>
<td>Path and filename for importing a catalog from a zip file.</td>
</tr>
<tr>
<td>--update</td>
<td>Resources in the catalog replace those in the repository if their URIs and types match.</td>
</tr>
<tr>
<td>--skip-user-update</td>
<td>When used with <code>--update</code>, users in the catalog are not imported or updated. Use this option to import catalogs without overwriting currently defined users.</td>
</tr>
<tr>
<td>--include-access-events</td>
<td>Restores access events (date, time, and username of last modification) on imported resources.</td>
</tr>
<tr>
<td>--include-audit-events</td>
<td>Imports any audit data that exists in the repository catalog.</td>
</tr>
</tbody>
</table>

Examples:

• Import the myExport.zip catalog archive file:

  ```bash
  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
  ```

The default behavior when a resource is found in the target repository that has the same URI as the resource that you are attempting to import is to skip the creation operation and leave the existing resource unchanged (no overwrite 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 of a different type than the existing resource, the server returns an error and skips the update operation.

When you import a user, if its roles exist in the repository, the user is given these 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 must have been created with access events. If you do not import access events, or if they don’t exist in the imported files, the date and time of the import are used.
CHAPTER 7  SYSTEM CONFIGURATION

You can change the default behavior of JasperReports Server by editing the system’s configuration files. Your changes take effect after you restart the server.

This chapter describes a subset of the options in the configuration files. Configuration of the auditing feature is covered in 8.2, “Configuring Auditing,” on page 116. Many more options are described in the JasperReports Server Installation Guide.

The file paths described in this chapter are all found under the <js-install> directory, which is the root of your JasperReports Server installation. Because these file locations vary with your application server, the paths specified in this chapter are partial. For example, the applicationContext.xml file is shown as residing in the WEB-INF folder; if you use Tomcat as your application server, the default path to this location is:

C:\Program Files\jasperreports-server-4.1\apache-tomcat\webapps\jasperserver-pro\WEB-INF

Use caution when editing the files described in this chapter. Inadvertent changes may cause unexpected errors throughout JasperReports Server that may be difficult to troubleshoot. Before changing these files, back them up to a location outside of your JasperReports Server installation.

This chapter includes:

• Security Framework Configuration
• Configuring Password Options
• Configuring the User Session Timeout
• Encrypting the Repository Database Password
• Configuring Ad Hoc
• Configuring System Logs
• Disabling the Domain Validation Check
• Configuring JasperReports Library
• Configuring the Heartbeat
• Removing Report Scheduling Interval Options
• Special Domain Support
• Configuring the Online Help
7.1 Security Framework Configuration

The JasperReports Server now has enterprise grade security through the integration of a comprehensive security framework. This new security framework protects the server against the common enterprise software security threats, such as:

- Cross-site scripting
- Cross-site request forgery
- SQL injection

The security framework restricts user input and values passed to reports to block any potential security threats. Administrators can monitor the server logs to search for evidence of attempted security breaches. For more information about logs, see section 7.6, “Configuring System Logs,” on page 103. However, input that was allowed in previous version of the server may be blocked, and users may see errors when entering values. In particular

- Parameter names and values cannot have tags (< and >). If your business data contains tags, you need to update the security configuration to allow them.
- SQL queries should start with SELECT and cannot have comments. If your reports or Domains have queries with comments, you need to either remove the comments or update the security configuration to allow them.

If users see recurring errors, administrators can examine logs to determine what input is not allowed. Preferably, users should modify their input to remove special characters that are security risks. If that is not feasible, administrators can configure the security framework to modify security rules or turn off the security restrictions, based on their perceived threat level. For assistance in configuring the security framework, please login to the Jaspersoft Support Portal and refer to the Customer Wiki.

These restrictions were added to comply with security guidelines for web applications. Removing the restrictions may make the server more vulnerable to web attacks.

7.2 Configuring Password Options

By default, passwords are stored in an encrypted format in the repository database. For information about how to change the way passwords are encrypted, refer to the JasperReports Server Installation Guide.

7.2.1 Enabling Password Expiration

If your users are externally-authenticated, do not enable this option.

If your security policies require your users to change their passwords at regular intervals, you can enable password expiration. In this case, JasperReports Server prompts users to change their passwords at the interval you specify. For example, if you set the password expiry to 90 days, the server prompts your users to change their passwords every three months. When a user’s password expires, the user cannot log in until she changes her password. The default value is 0; in this case, passwords don’t expire.

When this option is enabled, the server automatically enables the Change Password option on the Login page, even if allowUserPasswordChange is set to false.

<table>
<thead>
<tr>
<th>Password Administration Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration File</td>
</tr>
<tr>
<td>…\WEB-INF\jasperserver-servlet.xml (controls the Login page)</td>
</tr>
<tr>
<td>…\WEB-INF\applicationContext-security-web.xml (controls web services)</td>
</tr>
</tbody>
</table>
7.2.2 Allowing Users to Change their Passwords

In order to change their passwords, users click the Change Password link that appears on the Login page. Enable this link only if JasperReports Server authenticates your users. If your users are externally-authenticated, do not enable the option. To allow users to change their passwords, you must edit the following configuration file. Note that enabling the password expiration option (described in the previous section) automatically enables the ability of end users to change their passwords.

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>passwordExpirationInDays</td>
<td>0 &lt;default&gt;</td>
<td>Set the value to any positive, non-zero value to specify the number of days after which a password expires.</td>
</tr>
</tbody>
</table>

**Password Administration Option**

**Configuration File**

`...\WEB-INF\jasper-server-servlet.xml`

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowUserPasswordChange</td>
<td>false &lt;default&gt;</td>
<td>Set the value to true to enable the Change Password link. Any other value disables it.</td>
</tr>
</tbody>
</table>

7.3 Configuring the User Session Timeout

The user session timeout is the length of time a user’s session can remain inactive before the server automatically logs the user out. JasperReports Server now implements a pop-up reminder that tells users their session is about to expire and gives them the chance to continue without being logged out.

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;session-config&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;session-timeout&gt;</td>
<td>20 &lt;default&gt;</td>
<td>Set the number of minutes that a user session can remain idle before automatic logout. Set the value to 0 (zero) to prevent sessions from ever timing out.</td>
</tr>
</tbody>
</table>

**User Session Timeout**

**Configuration File**

`...\WEB-INF\web.xml`

Note that the session timeout also applies to how long a session remains in memory after a web services call finishes. If there is another web service call with the same credentials within the timeout period, the server reuses the same session. If the timeout is too short for this case, you may have performance issues caused by a high load of web service calls.

Conversely, if the timeout is too long, a user session may stay active for a long time (even indefinitely with a timeout of 0) if a user leaves his browser open. The risk of allowing long sessions is that the in-memory session is not updated with any role changes until the user logs out manually (ending the session) and logs in again (creating a new session).
7.4 Encrypting the Repository Database Password

The repository is stored as a private database that only JasperReports Server may access. The import and export utilities also access this database using the configuration settings in the file `<js-install>/buildomatic/build_conf/default/js.jdbc.properties`. In this file, the `metadata.jdbc.password` property displays the password to access the repository database in clear text. If the file permissions on your system are not sufficient, you may want to encrypt this password for added security.

To encrypt the repository database password:
1. Run the utility `js-encrypt-pass` with the `--pass` password option:
   ```bash
   % js-encrypt-pass.bat --pass mypassword
   ``
2. The utility encrypts `password` and displays the encryption key:
   ```plaintext
   Encrypting password: mypassword
   Add next string to js.jdbc.properties:
   metadata.jdbc.encryptedPassword=0x08,0x5a,0xf1,0x6d,0x96,0x13,0x3f,0x16,0xf2,0xfd
   ``
3. In the file `<js-install>/buildomatic/build_conf/default/js.jdbc.properties`, replace the property `metadata.jdbc.password` with the property and value that were output in the previous step. In this example, replace this:
   ```plaintext
   metadata.jdbc.password=mypassword
   ``
   With this:
   ```plaintext
   metadata.jdbc.encryptedPassword=0x08,0x5a,0xf1,0x6d,0x96,0x13,0x3f,0x16,0xf2,0xfd
   ```

7.5 Configuring Ad Hoc

Ad Hoc settings help you fine-tune the Ad Hoc Editor and the reports it creates. You must be logged in as a user with `ROLE_SUPERUSER` to manage the Ad Hoc settings and cache. This section includes:

- Ad Hoc Settings
- Ad Hoc Editor Configuration File Options
- Managing the Ad Hoc Cache
- Configuring Ad Hoc OLAP Filters

7.5.1 Ad Hoc Settings

The Ad Hoc settings apply to Ad Hoc reports based on Topics or Domains. Ad Hoc reports based on OLAP connections use the OLAP settings described in the `Jaspersoft OLAP User Guide`.

The Ad Hoc settings limit on the resources available for queries when Ad Hoc reports are designed and run. The options are:

- **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 report that is 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.

- **Ad Hoc Dataset Row Limit**. The maximum number of rows that an Ad Hoc report can return. JasperReports Server truncates the data 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 report while running its query. Setting this to a lower number may prevent exceptions from being displayed to users when they run Ad Hoc reports. Setting this to a higher number may prevent complex calculations from timing out, but it results in the use of more database connections.

7.5.1.1 Configuring the Ad Hoc Settings

Ad Hoc settings are reset every time the server restarts. If you make changes, you must redo them after every restart.
To set Ad Hoc settings in JasperReports Server:

1. Log in as system administrator (superuser by default).
2. Select Manage > Ad Hoc Settings.
3. In the Ad Hoc Filter List of Values Row Limit field, enter the maximum number of items to display in the Condition Editor when a user defines filters for an Ad Hoc report based on a Domain.
4. In the Ad Hoc Dataset Row Limit field, enter the maximum number of rows that an Ad Hoc report can return.
5. In the Ad Hoc Query Timeout (seconds) field, enter the number of seconds the server should wait before timing out an Ad Hoc report while running its query.
6. Click Save to make your changes effective. Or click Reset to return to the previously saved values.

7.5.1.2 Understanding Data Policies

Data policies determine how JasperReports Server handles data loading and processing for certain kinds of Ad Hoc reports. They determine how data is cached and where certain calculations occur. For example, you can specify that the data accessed by Domain-based reports is grouped, sorted, and aggregated in the database rather than having the server process it in memory.

By default, the server relies on the database to group, sort, and aggregate data returned by queries based on Domains when it is feasible. It also forces the server to only retrieve the fields that appear in the report (rather than the entire set of fields in the Domain). For queries based on JDBC data sources, such processing occurs in memory, and the entire result set defined in the Topic is returned.

If a query contains only aggregate values, JasperReports Server can generally transform it into an aggregate database query, but some functions (such as distinct count and average) must be calculated by reading all the detail rows from the database and performing the aggregation in memory. Note that independent check boxes control the behavior: one for Domains and another for JDBC data sources.

When these settings are disabled, the server loads the entire set of fields associated with a Domain or Topic into memory, and then applies the necessary grouping, sorting, and aggregation. This is also the case for Ad Hoc reports that do not rely on Domains or JDBC data sources; in these cases, the server processes the data in memory.

Generally, Jaspersoft recommends 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 complex sorting, grouping, or aggregation may perform better when the server optimizes the queries.
- 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 reports. If you have a large number of users creating and running Ad Hoc reports, 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 source of data is tuned for maximum performance, Ad Hoc reports may perform better when the server optimizes the queries. Note that, if your data source is hosted by MySQL, Jaspersoft recommends that you clear the check from the Optimize Queries for Domain-based Reports box.
- 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 reports 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 reports, Jaspersoft recommends 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.

7.5.1.3 Configuring the Data Policies

Data policies are reset every time the server restarts. If you make changes, you must redo them after every restart.
The data policies that you can set are:

- **Optimize Queries for JDBC-based Reports.** Selects grouping, sorting, and aggregation of queries for JDBC-based reports. Otherwise, the queries run unaltered in memory.

- **Optimize Queries for Domain-based Reports.** Selects grouping, sorting, and aggregation of queries for Domain-based reports. Otherwise, the queries run unaltered in memory.

**To set data policies in JasperReports Server:**

1. Log in as system administrator (superuser by default).
2. Select **Manage > Ad Hoc Settings**.
3. Select **Optimize Queries for JDBC-based Reports** to process queries for JDBC-based reports.
4. Select **Optimize Queries for Domain-based Reports** to process queries for Domain-based reports.
5. Click **Save** to make your changes effective. Or click **Reset** to return to the previously saved values.

These data policy settings do not retroactively update the existing reports in your repository. To change the data policy for an existing report, select the appropriate policy setting, open the report in the Ad Hoc Editor, and save the report.

**7.5.1.4 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. You can speed up the display of your data 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.

<table>
<thead>
<tr>
<th>Ad Hoc for Big Data</th>
<th>Configuration File</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>...\WEB-INF\applicationContext-adhoc.xml</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>calcMethod</td>
<td>DistinctCount</td>
<td>Change this property from value=&quot;sqlGroupBy&quot; (the default) to value=&quot;sqlUnionAll&quot;. The UnionAll is the modified query that provides distinct count computed by the database.</td>
</tr>
</tbody>
</table>

After making this change, redeploy the JasperReports Server webapp 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 calculated field, for example a Groovy-based field, anywhere in the report.
- There is a row or column group involving a time, timestamp, or date.

In these cases, Ad Hoc performs the distinct count summary calculations in memory, regardless of the calcMethod setting.
7.5.2 Ad Hoc Editor Configuration File Options

The following properties are among those that can be configured in the WEB-INF/applicationContext-adhoc.xml file:

Table 7-1 Configurable Properties in WEB-INF/applicationContext-adhoc.xml

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JrxmlScriptURI</td>
<td>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.</td>
</tr>
<tr>
<td>realmsURI</td>
<td>The repository location where Topics should reside. The default is /adhoc/topics.</td>
</tr>
<tr>
<td>defaultTheme</td>
<td>The name of the default style for Ad Hoc reports. This name must match a style defined in both a CSS and a JRXML file. The default is default.</td>
</tr>
<tr>
<td>aruFolder</td>
<td>The repository location where users are allowed to save their Ad Hoc reports. The default is /. This allows your users to save Ad Hoc reports anywhere. If you have a folder specifically for user content, specify this folder, for example, /userreports.</td>
</tr>
<tr>
<td>tempFolderName</td>
<td>The repository location where JasperReports Server saves intermediate versions of Ad Hoc reports as users run them from the editor. The default is /temp. Note that there is a temporary folders at the root, where the Ad Hoc Editor saves files created when superuser runs and saves reports, and in every organization. The JasperReports Server web interface only 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 under certain circumstances. As part of regular maintenance, you should periodically delete the files in these folders.</td>
</tr>
<tr>
<td>maxSafeGroupMembers</td>
<td>The maximum number of row groups or column groups a crosstab can display before the editor prompts the user to confirm that so many groups can be displayed. The default is 100. Set it higher to allow more groups to appear without prompting users; set it to a lower number to prompt users for smaller numbers of groups.</td>
</tr>
</tbody>
</table>

The repository URI locations are relative to the organization so that these settings apply to all organizations in the server instance. For example, for a user in the default organization, the URI /adhoc actually refers to /organizations/organization_1/adhoc.

7.5.3 Managing the Ad Hoc Cache

The Ad Hoc cache applies to Ad Hoc reports based on Topics or Domains. Ad Hoc reports based on OLAP connections use the OLAP cache. For a comparison of the two caches, see section 7.5.3.4, “Comparison with Jaspersoft OLAP,” on page 102. 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 reports. 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 hardware loads and delivers frequently-used datasets to the user quickly. Caching applies when reports are created as well as when they are run. You can configure the Ad Hoc cache to optimize response time for your usage patterns.

7.5.3.1 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 when different users run the same query, impairing performance. You can configure JasperReports Server to share cached datasets across users by adding the following lines to WEB-INF/applicationContext-
datarator.xml. The following code configures the cacheKeyInterceptor bean to ignore logged-in users’ credentials when creating the cache keys:

```xml
<property name="ignoredParameters">
  <list>
    ...
    <value>LoggedInUser</value>
    <value>LoggedInUsername</value>
  </list>
</property>
```

Restart JasperReports Server after adding the code.

### 7.5.3.2 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, JasperReports Server automatically removes datasets periodically. By default, datasets are removed from the cache if they are not accessed for 20 minutes. They are also cleared after 60 minutes, regardless of how recently they were accessed.

To configure the frequency with which the cache is automatically cleared, edit the following configuration file:

<table>
<thead>
<tr>
<th>Ad Hoc Cache Expiration</th>
<th>Configuration File</th>
</tr>
</thead>
<tbody>
<tr>
<td>...\WEB-INF\applicationContext-datarator.xml</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultTimeoutMinutes</td>
<td>dataSetCache</td>
<td>The number of minutes to wait before removing a dataset from the cache. Ensures that stale data is periodically replaced. The default is 60 minutes.</td>
</tr>
<tr>
<td>defaultUnusedTimeoutMinutes</td>
<td>dataSetCache</td>
<td>The number of minutes to wait after a dataset is used before removing it. The default is 20 minutes.</td>
</tr>
</tbody>
</table>

### 7.5.3.3 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 two values for every query:

- **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.

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 report with that query.

**To view queries and manually clear the Ad Hoc cache:**
1. In JasperReports Server, log in as system administrator (superuser by default).
2. Click **Manage > Ad Hoc Cache**.
   - The Ad Hoc Cache page appears, displaying all the datasets that are in the cache, sorted by age.
As shown in Figure 7-1, 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 only displays the first few lines of a query, as well as the data source.

3. To view the details of a specific query, including the full query string, click it 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.

4. To remove a dataset from the cache, close the Detail dialog if necessary, and click **Clear** beside the corresponding query.

5. To remove all datasets, click **Clear All** at the top of the Ad Hoc Cache page.
### 7.5.3.4 Comparison with Jaspersoft OLAP

The following table contrasts the key features of the Ad Hoc cache in JasperReports Server and Jaspersoft OLAP.

<table>
<thead>
<tr>
<th>Cache Feature</th>
<th>JasperReports Server</th>
<th>Jaspersoft OLAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structure of cache</td>
<td>Result caches are held at the query level: query text and language, plus datasource URI and query parameters.</td>
<td>Result caches are held at the analysis connection level: schema plus database connection.</td>
</tr>
<tr>
<td>Sharing</td>
<td>Not by default, but can be enabled as described in section 7.5.3.1 above.</td>
<td>There is only one cache; it is shared across all queries and users.</td>
</tr>
<tr>
<td>Security</td>
<td>Applied to cache control so that users are not allowed to see privileged data.</td>
<td></td>
</tr>
<tr>
<td>Populating</td>
<td>Queries populate the cache. You can also schedule reports to pre-populate the cache during off-hours.</td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>Limited by available JVM memory (heap). Not configurable.</td>
<td>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.</td>
</tr>
<tr>
<td>Automatic time-based cache policy</td>
<td>Configurable, as described in section 7.5.3.2 above.</td>
<td>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.</td>
</tr>
<tr>
<td>Clearing selected datasets manually</td>
<td>Configurable, as described in section 7.5.3.3 above.</td>
<td>Cache regions can be defined and cleared programmatically with APIs.</td>
</tr>
<tr>
<td>Clearing all datasets manually</td>
<td>Configurable, as described in section 7.5.3.3 above.</td>
<td>In JasperReports Server, select Manage &gt; OLAP Settings, then click Flush OLAP Cache. For additional methods, see the Jaspersoft OLAP Ultimate Guide.</td>
</tr>
</tbody>
</table>

### 7.5.3.5 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 report. Make sure that other reports 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 reports that present up-to-the-minute information from this data source. In this case, you do not want to retrieve old data out of the cache.

To disable the Ad Hoc cache for Topics and Domains, set the\texttt{defaultTimeoutMinutes} and \texttt{defaultUnusedTimeoutMinutes} parameters shown in section 7.5.3.2, “Configuring the Cache,” on page 100 both to 0 (zero).

To disable the OLAP cache for OLAP connections used in the Ad Hoc editor, check the\texttt{mondrian.rolap.star.disableCaching} setting on the Manage > 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:

```xml
<property name="applyQueryFilterInMemory" value="false"/>
<property name="applySecurityFilterInMemory" value="false"/>
<property name="applyDynamicFilterInMemory" value="false"/>
```
7.5.4 Configuring Ad Hoc OLAP Filters

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, there is a limit on the number of items in a filter. 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:

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxFilterValues</td>
<td>mdxDataStrategy</td>
<td>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.</td>
</tr>
</tbody>
</table>

7.6 Configuring System Logs

This section describes the settings that control the information JasperReports Server writes to its logs.

The log files contain important information about how the server is running. JasperReports Server uses the Apache log4j package to generate log files. Jaspersoft uses the slf4j facade to wrap log4j.

- The default log file is WEB-INF/logs/jasperserver.log.
- The default log configuration file is WEB-INF/log4j.properties.

7.6.1 Managing Log Settings

To set the current logging levels, click Manage > Log Settings; you must be logged in as superuser to access this menu.
The page lists some of the currently-enabled loggers (ones that typically need their logging levels adjusted from time to time) with their logging level. By selecting values from the drop-downs on the page, you can change the logging levels without restarting JasperReports Server.

These logging levels override the levels in the log4j.properties file. However, the override applies only until JasperReports Server is restarted. Logging levels revert to their log4j.properties settings when the server is restarted.

The four logging levels indicate the type of event that is recorded by a logger:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Level of Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERROR</td>
<td>Writes minimal information to the log describing serious program faults.</td>
</tr>
<tr>
<td>WARN</td>
<td>Writes error and warning messages to the log. Warning messages contain cautionary information to help you to decide whether the logged events require your attention.</td>
</tr>
<tr>
<td>INFO</td>
<td>Writes error, warning, and informational messages to the log. Informational messages describe significant events, such as those that affect application performance.</td>
</tr>
<tr>
<td>DEBUG</td>
<td>Writes 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.</td>
</tr>
</tbody>
</table>

JasperReports Server’s default root logger setting is WARN, as configured in log4j.properties. A logger that does not have an assigned value inherits the setting of its parent in log4j.properties.

The following table lists each logger name as it appears on the Log Settings page, the identifier to use to find a particular log in the log file, and a description of the logger.

<table>
<thead>
<tr>
<th>Logger Name</th>
<th>Identifier in log</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL query executer</td>
<td>JRJdbcQueryExe cuter</td>
<td>Logs SQL text and parameter values for queries that are run by the SQL query executer.</td>
</tr>
<tr>
<td>Input control value queries</td>
<td>valueQueryLog</td>
<td>Logs SQL text and parameter values for queries associated with input controls.</td>
</tr>
<tr>
<td>Cascading input control parameter resolution</td>
<td>FilterCore</td>
<td>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 changing the list of values for its input control.</td>
</tr>
<tr>
<td>Cascading input control query result caching</td>
<td>TokenControlLogic</td>
<td>Logs use of the cache for results of cascading input control queries.</td>
</tr>
<tr>
<td>Hibernate SQL</td>
<td>SQL</td>
<td>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.</td>
</tr>
<tr>
<td>Ad Hoc data policy logging</td>
<td>• CommonDomainDataStrategy &lt;br&gt;• SubFilterInputControlGenerator &lt;br&gt;• Others</td>
<td>Logs various activities of the Ad Hoc data policy implementations, which use SQL queries or in-memory operations to get datasets for Ad Hoc reports.</td>
</tr>
<tr>
<td>SQL generated for Domain queries</td>
<td>JdbcBaseDataSet</td>
<td>Logs SQL queries generated from queries using a Domain.</td>
</tr>
<tr>
<td>Connection handling for Domains</td>
<td>DataSourceResolverImpl</td>
<td>Logs use of JDBC connections used by Domains to run SQL queries.</td>
</tr>
</tbody>
</table>
You can add other loggers to the Log Settings page if you know their classnames.

**To add a logger to the page from the web interface:**

1. Click **Manage > Log Settings**.
2. Scroll to the bottom of the page.
3. Enter the logger’s classname in the text field. See the other properties on the page for guidance, for example:
   - `com.jaspersoft.ji.adhoc.action.AdhocCrosstabAjaxController`
4. Use the drop-down to set the logging level.

The logger is removed when the server is restarted. For information about adding loggers to this page permanently, see “**Adding a Logger to the Log Settings Page**” on page 106.
7.6.2 Log Configuration Files

Edit the log configuration file to set loggers, logging levels, and log output. Unlike changes made in the web interface, changes made directly to the log configuration files are persistent. Restart the server for your changes to take effect.

Logger names are defined in the Java source. Loggers can have any name, but the Jaspersoft convention is to give them their full class names. In the log4j properties file, the classname must be preceded by log4j.logger. For example, the classname org.acegisecurity.intercept is represented in the log4j.properties file as log4j.logger.org.acegisecurity.intercept is org.acegisecurity.intercept. If you want to add a new logger, find its classname in the source.

7.6.2.1 Configuring Logging Options

Depending on your whether you are configuring server logging or logging during import and export, edit a different file.

<table>
<thead>
<tr>
<th>Functionality to Log</th>
<th>File Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import/Export</td>
<td>&lt;js-install&gt;\buildomatic\conf_source\iePro</td>
</tr>
<tr>
<td>JasperReports Server</td>
<td>WEB-INF\log4j.properties in the JasperReports Server installation</td>
</tr>
</tbody>
</table>

If the logger is defined in the configuration file but is commented out, simply remove the comment character (#) to add the logger. Otherwise, add the logger’s classname and set it to the desired logging level.

The form of a logger definition should be:

```
log4j.logger.logger_classname = log_level, output_type
```

where:

- `logger_classname` is the name of the class you want to monitor.
- `log_level` is ERROR, WARN, INFO, or DEBUG
- `output_type` is a standard output type, such as stdout. For example:
  
  ```
  log4j.logger.org.springframework.webflow=DEBUG, stdout, fileout
  ```

Restart the server for your changes to take effect.

7.6.2.2 Adding a Logger to the Log Settings Page

If you know of a log4j logger that JasperReports Server uses, you can add it to the Log Settings page available to the superuser. To add a logger, edit a configuration file.

Because editing text files can be error-prone, Jaspersoft recommends that you add loggers from the web interface by entering them into the text field on the Log Settings page. Only edit the configuration file if you need to permanently add the logger.

To edit the list of loggers to be displayed on the page:

1. Edit the logger_descriptions.pro.properties file found under WEB-INF/bundles in your JasperReports Server installation.
2. Add a new line and specify the logger’s classname and a brief description of it.

   Entries should be in the form `<logger name> = <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 controls the labels for the English locale. You can specify labels for other locales by editing the logger description property file for that locale. For example, to add the label in French, add an entry to the logger_descriptions.pro.fr.properties file. For more information on supporting other languages, refer to “Localization” on page 133.
7.7 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 be time consuming. 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. You can disable the validation by editing the following configuration file:

<table>
<thead>
<tr>
<th>Login Page Administration Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration File</td>
</tr>
<tr>
<td>&quot;WEB-INF\applicationContext-semanticLayer.xml&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>skipDomainDatabaseValidation</td>
<td>slConfig</td>
<td>By default, this property is set to FALSE; in this case,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JasperReports Server validates Domain designs against their data sources. Set it to TRUE to disable this validation check.</td>
</tr>
</tbody>
</table>

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 use the Domain.

7.8 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 JasperReports Server behavior, from the way reports are exported into different file formats, to the default font to use.

These options can be set at different levels of granularity: Global (applies to all reports generated by the server), Report (defined in the JRXML and applies to a specific report), and Element (defined in the JRXML and applies to specific elements of the report). Global properties are defined in the WEB-INF/classes/jasperreports.properties file.

For more information about JasperReports Library configuration, see [http://jasperreports.sourceforge.net/config.reference.html](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
- Enabling the XHTML or HTML Exporter
- Enabling Flash or HTML5 for Pro Charts

7.8.1 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) file that contains a file called jasperreports_extension.properties, specifies a factory class. The specified 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 Ultimate Guide*. 
7.8.2 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. To do so, edit the following configuration file:

<table>
<thead>
<tr>
<th>Crosstab Report Configuration Option</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration File</strong></td>
</tr>
<tr>
<td><code>WEB-INF/classes/jasperreports.properties</code></td>
</tr>
<tr>
<td><strong>Property</strong></td>
</tr>
<tr>
<td><code>net.sf.jasperreports.crosstab.bucket.measure.limit</code></td>
</tr>
</tbody>
</table>

7.8.3 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 iReport Designer. Note that you can also apply a theme to individual chart elements, as well. Note that a chart theme can be included in a report unit as a resource; in this case, the theme is only available 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 may also be set as the global chart theme.

To set a theme as the default chart theme, edit the following configuration file:

<table>
<thead>
<tr>
<th>Global Report Theme</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration File</strong></td>
</tr>
<tr>
<td><code>WEB-INF/classes/jasperreports.properties</code></td>
</tr>
<tr>
<td><strong>Property</strong></td>
</tr>
<tr>
<td><code>net.sf.jasperreports.chart.ChartTheme</code></td>
</tr>
</tbody>
</table>

Jaspersoft recommends that you create your chart themes in Jaspersoft iReport Designer. Click **File > New > Chart Theme**, then use Jaspersoft iReport Designer to archive the new chart theme as a JAR.

![Chart themes do not apply to Ad Hoc chart reports.](chart.png)
7.8.4 Enabling the XHTML or HTML Exporter

By default, JasperReports Server exports to HTML format using an XHTML exporter. Unlike the default exporter from previous versions of the server (html), the XHTML exporter (xhtml) is more forgiving when exporting reports that have overlapping elements.

This setting affects all cases when HTML is exported, including when reports are exported from the report viewer and when they are scheduled to produce HTML output.

To use the older HTML exporter in JasperReports Server, edit the following configuration file:

<table>
<thead>
<tr>
<th>HTML Exporters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration File</td>
<td>…\WEB-INF\classes\jasperreports.properties</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>com.jaspersoft.jasperreports.export.html.type</td>
<td>Determines which of the HTML exporters is used. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• xhtml is the newest HTML exporter that handles overlapping report elements more gracefully than the html exporter.</td>
</tr>
<tr>
<td></td>
<td>• html is the default HTML exporter in versions of JasperReports Server prior to 4.2. Jaspersoft continues to support this exporter.</td>
</tr>
</tbody>
</table>

Note that the properties are mutually exclusive; you can only have one uncommented at a time.

If your reports include interactive elements, such as the table component (which supports sorting and filtering in the HTML preview), you must use the xhtml exporter in order to enable the interactive features; the html exporter doesn’t support them.

7.8.5 Enabling Flash or HTML5 for Pro Charts

By default, JasperReports Server renders Pro Charts (those based on Fusion Charts) using Adobe Flash. If Flash isn’t found in the client environment, the server renders the chart using HTML5, instead. For example, Pro Charts displayed on devices that run Apple’s iOS operating system are rendered using HTML5 because Flash isn’t available. Note that not all browsers support HTML5.

You can configure the server to default to HTML5 when rendering Pro Charts. In this case, if your browser doesn’t support HTML5, the chart isn’t rendered.

To render Pro Charts using HTML5, edit the following configuration file:

<table>
<thead>
<tr>
<th>Pro Charts Renderer</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration File</td>
<td>…\WEB-INF\classes\jasperreports.properties</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>com.jaspersoft.jasperreports.fusion.charts.render.type</td>
<td>Determines which of the following renderers is used:</td>
</tr>
<tr>
<td></td>
<td>• flash is the default renderer for Pro Charts. If Flash isn’t available, the server tries to render the chart in HTML5.</td>
</tr>
<tr>
<td></td>
<td>• html5 is the newest renderer for Pro Charts. Use it if you can’t support Flash.</td>
</tr>
</tbody>
</table>
Note that this property only applies to reports that rely on Pro Charts and only affects 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, you must 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.

### 7.9 Configuring the Heartbeat

During installation (or the first time an administrator logs in), you are prompted whether to participate in Jaspersoft’s Heartbeat program, which reports specific information to Jaspersoft about your implementation, such as the operating system, JVM, application server, RDBMS (type and version), and JasperReports Server edition and version number. Once this option is set, you can change it by editing the following configuration file:

<table>
<thead>
<tr>
<th>Login Page Administration Options</th>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration File</td>
<td>enabled</td>
<td>heartbeatBean</td>
<td>When this property is set to TRUE, JasperReports Server reports information about your environment to Jaspersoft once a week. When it is set to FALSE, information is not sent.</td>
</tr>
</tbody>
</table>

### 7.10 Removing Report Scheduling Interval Options

When users schedule reports, they can specify that the report run periodically at regular intervals. For simple recurrence, the default interval can be expressed in days, hours, or minutes. If it is necessary to prevent users from scheduling frequent reports, you can limit the intervals to days or hours by editing the following configuration file:

<table>
<thead>
<tr>
<th>Scheduling Options</th>
<th>Configuration File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>reportJobBeans.xml</td>
<td>Comment out the intervals you want to disable. For example, to disable the Minutes option, comment out the INTERVAL_MINUTE property.</td>
</tr>
</tbody>
</table>
By default, the \texttt{INTERVAL\_MINUTE} bean is:

```xml
<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>
```

To comment it out, enclose the bean in comment characters.

```xml
<!--
<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>
-->```

You can comment out the \texttt{INTERVAL\_HOUR} bean in the same way.

### 7.11 Special Domain Support

When you use Domains with certain specific database constructs, you may need to configure JasperReports Server:
- Enabling Oracle Synonyms
- Enabling CLOB Fields
- Enabling Proprietary Types

#### 7.11.1 Enabling Oracle Synonyms

By default, Domains cannot access synonyms in an Oracle database. Set the following property to enable them. If you access your Oracle database through JNDI, you also need to configure the JNDI connection.

Be aware that the Oracle metadata service works significantly slower when synonyms are in scope.

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeSynonymForOracle</td>
<td>jdbcMetaConfiguration</td>
<td>Set the value to TRUE: &lt;value&gt;true&lt;/value&gt;</td>
</tr>
</tbody>
</table>

**Configuration File**

```xml
.../WEB-INF/applicationContext-semanticLayer.xml
```
7.11.2 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, you must set the following options according to your database.

The Oracle JDBC driver implementation uses the `CLOB` JDBC type for CLOB fields.

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accessToUnderlyingConnectionAllowed</code></td>
<td><code>&lt;Resource name=&quot;jdbc/oracle&quot;...</code></td>
<td>If you use JNDI, add the following property: accessToUnderlyingConnection Allowed=&quot;true&quot;</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>jdbc2JavaTypeMapping</code></td>
<td><code>jdbcMetaConfiguration</code></td>
<td>This property contains a map of database field types to Java types. Find the line for CLOB that is commented out: &lt;!--entry key=&quot;CLOB&quot; value=&quot;&quot;/&gt; Modify it as follows: &lt;entry key=&quot;CLOB&quot; value=&quot;java.lang.String&quot;/&gt;</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>jdbc2JavaTypeMapping</code></td>
<td><code>jdbcMetaConfiguration</code></td>
<td>This property contains a map of database field types to Java types. Find the following lines: &lt;!--entry key=&quot;CLOB&quot; value=&quot;&quot;/&gt; &lt;!--entry key=&quot;LONGVARBINARY&quot; value=&quot;&quot;/&gt; And modify them as follows: &lt;entry key=&quot;CLOB&quot; value=&quot;java.lang.String&quot;/&gt; &lt;entry key=&quot;LONGVARBINARY&quot; value=&quot;java.lang.String&quot;/&gt;</td>
</tr>
</tbody>
</table>
7.11.3 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, such as NVARCHAR2 in Oracle. You can map these types with a special configuration.

As a prerequisite, the proprietary type must be logically equivalent to one of following Java classes:

- java.lang.Boolean
- java.lang.Byte
- java.lang.Character
- java.lang.Double
- java.lang.Float
- java.lang.Integer
- java.lang.Long
- java.lang.Short
- java.lang.String
- java.math.BigDecimal
- java.sql.Date
- java.sql.Time
- java.sql.Timestamp
- java.util.Date

Proprietary Database Type Mapping

<table>
<thead>
<tr>
<th>Configuration File</th>
<th>Properties</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
</table>
| applicationContext-semanticLayer.xml | jdbc2JavaType Mapping | jdbcMeta Configuration | To modify the generic mapping, edit this line: <entry key="NUMERIC" value="java.math.BigDecimal"/>
Add any secondary key to the OTHER key, following this example: <entry key="OTHER">
<map>
<entry key="NVARCHAR2" value="java.lang.String"/>
</map>
</entry>
Java 1.6 supports java.sql.Types.NVARCHAR, therefore you should add it as a generic mapping, not under OTHER: <entry key="NVARCHAR" value="java.lang.String"/>

7.12 Configuring the Online Help

JasperReports Server 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

```text
...\WEB-INF\applicationContext-webHelp.xml
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showHelp</td>
<td>webHelp</td>
<td>Determines whether the help links are displayed in the JasperReports Server web UI. Valid values are <code>true</code> and <code>false</code>. The help link appears on the login page (Online Help) as well as to the left of Logout on other pages.</td>
</tr>
<tr>
<td>hostURL</td>
<td>webHelp</td>
<td>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.</td>
</tr>
<tr>
<td>pagePrefix</td>
<td>webHelp</td>
<td>Defines the default page name to pass to the web server hosting the help system. The only valid value is <code>Default_CSH.htm</code> for this property.</td>
</tr>
<tr>
<td>helpContextMap</td>
<td>webHelp</td>
<td>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.</td>
</tr>
</tbody>
</table>
CHAPTER 8 AUDITING

Auditing stores key events that are of interest to system administrators, such as login/logout time, user, report generated, report details, and object sizes. The audited events can be saved and archived. Recently audited events can be moved to the archive automatically after a specified number of days.

This chapter contains the following sections:
- Audited Events
- Configuring Auditing
- Using the Audit Domains
- Importing and Exporting Audit Data

8.1 Audited Events

In broad terms, an audit event is any atomic operation that can be monitored by the audit system. Event properties and attributes are features of the event; they can be defined internally or in custom code.

The following table lists the defined audit events and the information that is 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.

<table>
<thead>
<tr>
<th>Event</th>
<th>Information Collected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log in or log out</td>
<td>Time and user ID (recorded for every event)</td>
</tr>
<tr>
<td>Log in as</td>
<td>User logged in as</td>
</tr>
</tbody>
</table>
| Run report (from repository or from Ad Hoc Editor) 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  
| | Query execution time*  
| | Report rendering time*  
| | Caching parameters  
| | Errors that occurred |
## 8.2 Configuring Auditing

Configuration settings for audit logging are in the configuration file `applicationContext-audit.xml`.

### 8.2.1 Enabling Auditing

By default, auditing is off.

To enable auditing, locate the `GenericBeanUpdater` bean in the configuration file set and enable the following property:

```xml
<entry key="com.jaspersoft.jasperserver.api.logging.audit.domain.AuditEvent" value="true"/>
```

To disable auditing, set the value to `false`.

---

**Audited Events, continued**

<table>
<thead>
<tr>
<th>Event</th>
<th>Information Collected</th>
</tr>
</thead>
</table>
| 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 in Ad Hoc |
| 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 |
| Permissions added, updated, or deleted | • Resource or folder referenced  
• Previous permissions (before update) |
| User added, updated, or deleted, also user password change | • User ID  
• User name  
• Email  
• Enabled flag  
• External flag  
• Profile attributes |
| Role added, updated, or deleted | • Role ID  
• Role name  
• Role organization |
| Organization added, updated, or deleted | • Organization ID  
• Organization description |

* Execution and rendering times are measured in milliseconds.
8.2.2 Archive Options

Archiving is on when auditing is on. However, you should set the archiving interval to a level appropriate for your organization. Use the bean `auditServiceTarget`.

<table>
<thead>
<tr>
<th>Auditing Archive Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration File</td>
</tr>
<tr>
<td>...\WEB-INF\applicationContext-audit.xml</td>
</tr>
</tbody>
</table>

### Configuration File

<table>
<thead>
<tr>
<th>Property</th>
<th>Bean</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxAuditEventAge</td>
<td>auditServiceTarget</td>
<td>The number of days to keep audit data in the active database. Older data is moved to the archive.</td>
</tr>
<tr>
<td>maxAuditEventAge</td>
<td>auditServiceTarget</td>
<td>The number of days to keep the data. Older data is purged (deleted). Zero, the default, means old data is never purged.</td>
</tr>
<tr>
<td>cronExpression</td>
<td>auditEventArchiverTrigger</td>
<td>Defines the frequency of the archiving job in cron syntax.</td>
</tr>
<tr>
<td>cronExpression</td>
<td>auditEventPurgerTrigger</td>
<td>Defines the frequency of the audit purge job in cron syntax.</td>
</tr>
</tbody>
</table>

The `cronExpression` properties use standard cron syntax. For example, `0 0 3 * * ?` runs a job once a day at 3 a.m.

8.2.3 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.

In the file, event types and their properties are listed in the `enabledEventsMapping` map (`<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 a 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,exception" /> --
```

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" />
  ```

8.3 Using the Audit Domains

You can use the Ad Hoc Editor to explore the audit data and generate reports. Two Domains were created for this purpose:

- Audit Domain – Use this Domain to run reports on current audit data; they run against the active audit database.
- Audit Archive Domain – Use this Domain to run reports on archived data; they run against the archive database.

The contents of both Domains are identical—the only difference is the database that is accessed in each case.

In Ad Hoc, the Domains are on the **Domains** tab. In the repository, the Domains are in the Public/Audit folder. In both cases, the audit Domains are accessible only to administrators.
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 next section documents the contents of the audit Domains.

### 8.3.1 Domain Items

The following tables describe the items in both audit Domains (Audit Domain and Audit Archive Domain), which correspond to the information that is recorded for each event. When creating a report based on either Domain, choose the items that correspond to the type of event you want to report on.

Domain items in the following table are used in general events as well as repository events:

<table>
<thead>
<tr>
<th>Domain Item</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Date event occurred.</td>
</tr>
<tr>
<td>Prop Long Value</td>
<td><em>clob</em> value of event property, such as query.</td>
</tr>
<tr>
<td>Prop Type</td>
<td>Type of event property, such as destination folder, as per event map in configuration file.</td>
</tr>
<tr>
<td>Prop Value</td>
<td><em>string</em> value of event property, such as folder name.</td>
</tr>
<tr>
<td>Time</td>
<td>Time event occurred.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Type of event, such as save resource, as per event map in configuration file.</td>
</tr>
<tr>
<td>Request Type</td>
<td>Repository request type of event.</td>
</tr>
<tr>
<td>Resource Type</td>
<td>Repository type of resource accessed in event.</td>
</tr>
<tr>
<td>Resource URI</td>
<td>URI of repository resource.</td>
</tr>
</tbody>
</table>

Domain items in the following table are recorded for user events:

<table>
<thead>
<tr>
<th>Domain Item</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-mail</td>
<td>E-mail address of event user (user at time of event).</td>
</tr>
<tr>
<td>Enabled</td>
<td>Whether event user is currently a user.</td>
</tr>
<tr>
<td>External</td>
<td>Whether event user was an external user.</td>
</tr>
<tr>
<td>Full Name</td>
<td>Full name of event user.</td>
</tr>
<tr>
<td>Password Changed</td>
<td>Whether event was a change of password</td>
</tr>
<tr>
<td>Organization</td>
<td>Organization of event user.</td>
</tr>
<tr>
<td>User Name</td>
<td>UserID of event user.</td>
</tr>
</tbody>
</table>

Domain items in the following table are recorded for role events:

<table>
<thead>
<tr>
<th>Domain Item</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>External</td>
<td>Whether role was defined in an external system.</td>
</tr>
<tr>
<td>Role Name</td>
<td>Name of the role in the event.</td>
</tr>
<tr>
<td>Organization</td>
<td>Organization of the role in the event.</td>
</tr>
</tbody>
</table>
Domain items in the following table are recorded when a report is generated:

<table>
<thead>
<tr>
<th>Domain Item</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Date the report is generated.</td>
</tr>
<tr>
<td>Time</td>
<td>Time the report is generated.</td>
</tr>
<tr>
<td>Resource URI</td>
<td>URI of repository resource accessed for report.</td>
</tr>
<tr>
<td>Resource Type</td>
<td>Repository type of resource accessed for report.</td>
</tr>
<tr>
<td>Datasource URI</td>
<td>URI of data source accessed for report.</td>
</tr>
<tr>
<td>Query Execution Time</td>
<td>Time to execute the query in the database.</td>
</tr>
<tr>
<td>Report Rendering Time</td>
<td>Time to prepare query results for display.</td>
</tr>
<tr>
<td>Report Execution Time</td>
<td>Total time to execute the report (query execution + report rendering).</td>
</tr>
<tr>
<td>Query</td>
<td>Specification of the report query.</td>
</tr>
<tr>
<td>User Name</td>
<td>UserID of event user.</td>
</tr>
<tr>
<td>Organization</td>
<td>Organization of event user.</td>
</tr>
<tr>
<td>Crosstab Group Field</td>
<td>Field name used in crosstab</td>
</tr>
</tbody>
</table>

### 8.3.2 Sample Reports

A number of sample Ad Hoc reports based on the audit Domains are provided in the Public/Audit/Audit Reports repository folder. As with all audit material, these reports are visible only to administrators. The reports provide an example of the data that is recorded and how it can be used for auditing. You can also modify these reports in the Ad Hoc Editor to suit your auditing requirements:

- Audit Report – Generic example of an audit report showing commonly audited events.
- Performance Crosstab Report – A crosstab that shows average performance of reports that were run.
- Performance Report – Generates a list of reports that were run and 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 reports that were run.
- User Activity Report – Generates a list of reports run by a specific user.

The parameters for these reports and the report contents are blank by default, because auditing is disabled by default and no audit data exists. To view these reports, first enable auditing as described in section 8.2, “Configuring Auditing,” on page 116, then wait for user activity to generate events.

The same reports are also provided in the Public/Audit/Audit Reports/Archived Audit Reports folder. These reports are identical to those listed above, except they use the Audit Archive Domain and run against the archived audit data. These reports are blank until you enable auditing and archiving and until audit data is archived by the audit archive configuration.

### 8.4 Importing and Exporting Audit Data

Audit data can be imported and exported with the utilities described in chapter Chapter 6, “Import/Export,” on page 89.

- To export audit data, use the --include-audit-events option in the export command.
- To import audit data, import the folder that contains audit data, and use the option --include-audit-events in the import command.

Data in temp folders is not exported.
APPENDIX A  TROUBLESHOOTING

This appendix contains the following sections:

- **Number of Users Exceeded**
- **Field Names Disappear in Ad Hoc Editor**
- **Scheduler Sending Multiple Emails**
- **Adding Data Sources**

A.1 **Number of Users Exceeded**

When the number of users in the internal database is larger than allowed by your license, the login page displays a warning, but users can still log in. After a 24-hour grace period, users can no longer log in, and administrators receive an email. 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.
- Remove users 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 section 2.2, “Managing Users,” on page 23.

A.2 **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 Report field labels, use the resource bundle mechanism:

1. Create a resource bundle (*.properties) and associate each of your field label with a unique key. Use unicode escape sequences such as \u00e0 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 that the Topic needs to support, and users see the labels for the locale they set in their browser.
A.3 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 reports that are scheduled, 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 <WAR-file>/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.

A.4 Adding Data Sources

When adding a data source to JasperReports Server, there are several areas that can cause errors.

A.4.1 JDBC Driver in Classpath

JasperReports Server ships with drivers for several databases: DB2, Oracle, PostgreSQL, and SQL Server. It is important to ensure that the driver for the database that you want to set up as a data source is included in the server’s classpath.

Refer to the JasperReports Server Installation Guide for the location of the drivers and how to set the classpath.

A.4.2 JDBC Driver Name

When creating a data source, you type in the name of the JDBC driver for your database. Because the names for drivers are different for each database, and typing class names is prone to error, make sure you typed the exact class name for your database:

<table>
<thead>
<tr>
<th>Database</th>
<th>JDBC Driver Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>MySQL</td>
<td>com.mysql.jdbc.Driver</td>
</tr>
<tr>
<td>Ingres</td>
<td>com.ingres.jdbc.IngresDriver</td>
</tr>
<tr>
<td>Oracle</td>
<td>oracle.jdbc.OracleDriver</td>
</tr>
<tr>
<td>DB2</td>
<td>com.ibm.db2.jcc.DB2Driver</td>
</tr>
<tr>
<td>SQL Server</td>
<td>com.microsoft.sqlserver.jdbc.SQLServerDriver</td>
</tr>
</tbody>
</table>
A.4.3 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 should have the appropriate select permissions to query the tables within your database for the reports you want to generate.
- If you accept the defaults during installation of JasperReports Server on Linux from an RPM using apt-get, rpm, or yum, the bundled PostgreSQL only allows 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 permissions name the specific host from which connections are allowed. Otherwise, when testing the JDBC connection, a connection may not be allowed even though the user name and password are correct. For example, this setting for MySQL users is documented at [http://dev.mysql.com/doc/refman/5.1/en/adding-users.html](http://dev.mysql.com/doc/refman/5.1/en/adding-users.html).

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.

A.4.4 JDBC Database 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:

<table>
<thead>
<tr>
<th>Database</th>
<th>Default JDBC Database URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>PostgreSQL</td>
<td>jdbc:postgresql://localhost:5432/jasperserver</td>
</tr>
<tr>
<td>MySQL</td>
<td>jdbc:mysql://localhost:3306/jasperserver?useUnicode=true&amp;characterEncoding=UTF-8</td>
</tr>
<tr>
<td>Ingres</td>
<td>jdbc:ingres://localhost:II7/jasperserver;CURSOR=READONLY;auto=multi</td>
</tr>
<tr>
<td>Oracle</td>
<td>jdbc:oracle:thin:@localhost:1521:orcl</td>
</tr>
<tr>
<td>DB2</td>
<td>jdbc:db2://localhost:50000/jsprsrvr:driverType=4;fullyMaterializeLobData=true;fullyMaterializeInputStreams=true;progressiveStreaming=2;progressiveLocators=2;currentSchema=JSPRSRVR</td>
</tr>
<tr>
<td>SQL Server</td>
<td>jdbc:sqlserver://localhost:1433;databaseName=jasperserver;SelectMethod=cursor</td>
</tr>
</tbody>
</table>

A.4.5 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[-pro]/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-pro/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 the driver for your database connection is in the `<tomcat>/lib` folder. This is the same database-specific driver class that is given in section A.4.2, “JDBC Driver Name,” on page 122.
- If you installed JasperServer from a WAR file, Tomcat may have created a separate copy of context.xml in `<tomcat>/conf/Catalina/Localhost/jasperserver-pro.xml`. See the corresponding section in the troubleshooting appendix of the JasperReports Server Installation Guide.

### A.4.6 JNDI Services on JBoss

After defining JNDI Services on the JBoss application server, JasperReports Server does not automatically detect the new services. In order to use the new JNDI services as data sources in the server, follow these steps:

1. Define and deploy a JNDI data source in the JBoss administrator console.
2. Modify the file `<jboss>/webapps/jasperserver-pro/WEB-INF/web.xml` to include a data source reference to this new JNDI service.
3. Modify `<jboss>-web.xml` to include a reference to this data source.
4. Because the deployment configuration files such as web.xml were modified, redeploy the JasperReports Server application.

Now you can define JNDI data source in the repository, as described in section 4.1.2, “JNDI Data Sources,” on page 50.
These instructions assume that both JasperReports Server and Liferay have been installed properly. You can get Liferay from http://www.liferay.com; for instructions about installing it, refer to its documentation. For instructions about installing JasperReports Server, refer to the JasperReports Server Installation Guide.

This chapter also assumes:
- You are using Liferay deployed in Tomcat.
- `<js-install>` is the root of your JasperReports Server installation.
- `<liferay-install>` is the root of your Liferay installation. This path cannot contain any spaces.
- `<liferay-install>\<tomcat-dir>` is the root of the Tomcat instance running Liferay.

The instructions in this chapter describe how to deploy the portlet WAR that is included in the JasperReports Server distribution. If you are implementing the JasperReports Server portlet from a different distribution, refer to the instructions provided in that distribution.

If you are upgrading from a previous version of JasperReports Server (called JasperServer) in which you deployed the Liferay portal, you must take additional steps; pay careful attention when following the instructions in this chapter.

### B.1 Changing Liferay’s Port Numbers

By default, both Liferay and JasperReports Server run on port 8080. You must change Liferay’s listening port to avoid this conflict. For example, you can change Liferay’s listening port to 7080. You must also update a number of other Liferay ports.

**To configure Liferay’s port numbers:**

1. In the server.xml file (found at `<liferay-install>\<tomcat-dir>\conf\`), change all the occurrences of port numbers. This table suggests effective replacement ports:

<table>
<thead>
<tr>
<th>Port</th>
<th>Default Value</th>
<th>Suggested Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Port</td>
<td>8005</td>
<td>7005</td>
</tr>
<tr>
<td>Non-SSL Connector Port</td>
<td>8080</td>
<td>7080</td>
</tr>
<tr>
<td>AGP 1.3 Connector Port</td>
<td>8009</td>
<td>7009</td>
</tr>
<tr>
<td>Proxy Connector Port</td>
<td>8082</td>
<td>7082</td>
</tr>
</tbody>
</table>
2. Also change the port numbers in the server-minimal.xml file (found at `<liferay-install><tomcat-dir><conf>`):

<table>
<thead>
<tr>
<th>Port</th>
<th>Default Value</th>
<th>Suggested Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Port</td>
<td>8005</td>
<td>7005</td>
</tr>
<tr>
<td>Catalina Connector Port</td>
<td>8080</td>
<td>7080</td>
</tr>
<tr>
<td>AGP 1.3 Connector Port</td>
<td>8009</td>
<td>7009</td>
</tr>
</tbody>
</table>

3. Start the Liferay Portal by entering the following at the command prompt:

<table>
<thead>
<tr>
<th>OS</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux</td>
<td><code>&lt;liferay-install&gt;&lt;tomcat-dir&gt;&lt;bin&gt;&lt;startup.sh&gt;</code></td>
</tr>
<tr>
<td>Windows</td>
<td><code>&lt;liferay-install&gt;&lt;tomcat-dir&gt;&lt;bin&gt;&lt;startup.bat&gt;</code></td>
</tr>
</tbody>
</table>

### B.2 Configuring JasperReports Server to Accept Web Services Calls

Configure JasperReports Server to accept web service calls from Liferay by updating:

`<js-install>/webapps/jasperserver-pro/WEB-INF/applicationContext-security.xml`

**To configure JasperReports Server to accept web services from trusted hosts:**

2. Locate the `trustedIpAddress` property in the applicationContext-security-web.xml file found at `<js-install>/webapps/jasperserver-pro/WEB-INF`.
3. Add the IP address of your Liferay Portal host. For example, if Liferay is running on a computer with the IP address 123.45.6.789, the `trustedIpAddress` property would be:

   ```xml
   <property name="trustedIpAddress">
   <list>
     <value>123.45.6.789</value>
   </list>
   </property>
   ```

   You can also use the value `localhost` or `127.0.0.1` for this property.

### B.3 Configuring Liferay to Access JasperReports Server

The JasperReports Server portlet uses web services to retrieve information from JasperReports Server. This parameter specifies the JasperReports Server repository web service URL. It’s normally of the form:

- `http://<host:port>/jasperserver-pro/services/repository`
- `https://<host:port>/jasperserver-pro/services/repository`

For example:

`http://123.45.6.789:7080/jasperserver-pro/services/repository`

**To configure Liferay to access JasperReports Server:**

1. Locate the `jasperserver_repository_ws_url` parameter of the portlet.xml file found at `<liferay-install><tomcat-dir>/webapps/<JasperServerPortlet-x.x.x>/WEB-INF`. 
Set its value to the URL of the JasperReports Server repository web service. For example:

```xml
<init-param>
  <name>jasperserver_repository_ws_url</name>
  <value>http://123.45.6.789:8080/jasperserver-pro/services/repository</value>
</init-param>
```

2. Restart Liferay.

### B.4 Setting the Portal Organization

You can connect to only one Portal organization but you can set the organization to any name.

**To set the Portal organization:**

2. Locate the following line in JavaBean mtPortletAuthenticationProcessingFilter in the applicationContext-multitenancy-web.xml file found at `<js-install>/webapps/jasperserver-pro/WEB-INF:

   ```xml
   <property name="portletOrganizationId" value="portal"/>
   ```

3. Set the value of `portal` to the desired organization.

See also the note in B.8, “Testing the JasperReports Server Portlet,” on page 129.

### B.5 Testing Liferay

**To test Liferay:**

1. Enter your Liferay Portal URL in your browser’s address bar. For example:
   
   ```
   http://<liferay host>
   ```

2. Login to Liferay; by default the sample data in Liferay includes a default Admin user with these credentials:
   - **User name:** bruno@7cogs.com
   - **Password:** bruno

3. If the sample data is used, the default administrative user becomes valid:
   - **User name:** test@liferay.com
   - **Password:** test

If the portal appears, Liferay is properly installed.

### B.6 Deploying the JasperReports Server Portlet WAR File

If you are upgrading from a previous version of JasperReports Server (then called JasperServer) in which you deployed the portlet WAR file, you must delete the webapps/Jaspersoft folder of the application server hosting Liferay. This deletes libraries used in older versions that conflict with libraries in the latest version. Once this folder is deleted, you can deploy the new portlet WAR.

**To deploy the portlet WAR file,**

1. Copy JasperServerPortlet-3.7.0.war to `<liferay-install>\<tomcat-dir>\deploy`
2. Ensure that Liferay is running.
3. Login on the Liferay Portal as a user with administrative privileges. For example, bruno@7cogs.com. The default password is bruno.
   If you just started Liferay, it attempts to deploy the WAR file (this step may take several moments, depending on your environment).

4. When Liferay finishes the deployment, click **Add Application > Jaspersoft > JasperServer Portlet** appears.
   If the JasperReports Server portlet appears, it is deployed.

Note that you can run multiple instances of the portlet at once.

### B.7 Configuring a Default Report to Display

By default, the portlet displays all the reports that the current user is allowed to view. You can configure the portlet to display a specific report, instead.

In the portlet.xml file, add the following entries to specify a default report:

- **full_resource_path**: This parameter specifies the full report Path.
- **resource_type**: The only supported value is report
- **number_of_parameters**: This specifies the number of parameters the report requires, including optional and required parameters.
- **js_resource_parameter_<parameter>**: For each parameter of the report, append prefix **js_resource_parameter_** to the name. In this example, the report /reports/samples/EmployeeAccounts takes a parameter named EmployeeID; it is defined as **js_resource_parameter_EmployeeID**.
- **modifiable**: This value has to be 1 [one] so end users can choose different reports at run time. 1 is the only supported value. For details of this parameter, please refer to the JSR-168 specification.

For example, the portlet configuration section of the portlet.xml file might be similar to the following:

```xml
<portlet-preferences>
  <preference>
    <name>full_resource_path</name>
    <value>/reports/samples/EmployeeAccounts</value>
    <modifiable>1</modifiable>
  </preference>

  <preference>
    <name>resource_type</name>
    <value>report</value>
    <modifiable>1</modifiable>
  </preference>

  <preference>
    <name>number_of_parameters</name>
    <value>1</value>
    <modifiable>1</modifiable>
  </preference>

  <preference>
    <name>js_resource_parameter_EmployeeID</name>
    <value>beth_id</value>
    <modifiable>1</modifiable>
  </preference>
</portlet-preferences>
```

If you remove this section, the JasperReports Server Portlet displays a list of reports that the current user can access.
B.8 Testing the JasperReports Server Portlet

To test the portlet:
1. Login to Liferay as described in B.5, “Testing Liferay,” on page 127.
3. If the portlet doesn’t appear, add it as described in B.6, “Deploying the JasperReports Server Portlet WAR File,” on page 127.

When Liferay connects to JasperReports Server, JasperReports Server looks for the Portal organization; it creates this organization if it doesn’t exist. The server also looks for the user ID; if the ID is found, the server allows access; if it isn’t found, the user is created and assigned the ROLE_PORTLET and ROLE_USER roles. If your JasperReports Server instance hosts multiple organizations, the user is mapped to the Portal organization.

B.9 JasperReports Server Portlet Configuration Options

You can configure several parameters that control the JasperReports Server portlet. They are located in the <liferay-install>/<tomcat-dir>/webapps/<JasperServerPortlet-x.x.x>/WEB-INF/portlet.xml file. You must restart Liferay for these changes to take effect.

The values described in this section are case-sensitive.

This table lists the configurable parameters of the portlet.

<table>
<thead>
<tr>
<th>Portlet Parameters</th>
<th>Code &amp; Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
<td><strong>Code &amp; Comment</strong></td>
</tr>
<tr>
<td>portal_server</td>
<td><code>&lt;init-param&gt;</code>&lt;br&gt;<code>&lt;name&gt;portal_server&lt;/name&gt;</code>&lt;br&gt;<code>&lt;value&gt;liferay&lt;/value&gt;</code>&lt;br&gt;<code>&lt;/init-param&gt;</code>&lt;br&gt;The only supported value for this parameter is liferay.</td>
</tr>
<tr>
<td>show_logo</td>
<td><code>&lt;init-param&gt;</code>&lt;br&gt;<code>&lt;name&gt;show_logo&lt;/name&gt;</code>&lt;br&gt;<code>&lt;value&gt;true&lt;/value&gt;</code>&lt;br&gt;<code>&lt;/init-param&gt;</code>&lt;br&gt;This parameter indicates whether the company logo (at the top right-hand corner of the portlet) is displayed. Any value other than true hides the logo.</td>
</tr>
<tr>
<td>show_return_to_report_list_icon</td>
<td><code>&lt;init-param&gt;</code>&lt;br&gt;<code>&lt;name&gt;show_return_to_report_list_icon&lt;/name&gt;</code>&lt;br&gt;<code>&lt;value&gt;true&lt;/value&gt;</code>&lt;br&gt;<code>&lt;/init-param&gt;</code>&lt;br&gt;The parameter indicates whether the Return to Report List icon is displayed. Any value other than true hides the icon.</td>
</tr>
</tbody>
</table>
### Portlet Parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Code &amp; Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>show_return_to_parameter_icon</td>
<td><code>&lt;init-param&gt;</code>&lt;br&gt;  <code>&lt;name&gt;show_return_to_parameter_icon&lt;/name&gt;</code>&lt;br&gt;  <code>&lt;value&gt;true&lt;/value&gt;</code>&lt;br&gt;  <code>&lt;/init-param&gt;</code>&lt;br&gt;  The parameter indicates whether the Report Options icon should be shown. Any value other than true hides the icon.</td>
</tr>
<tr>
<td>company_logo</td>
<td><code>&lt;init-param&gt;</code>&lt;br&gt;  <code>&lt;name&gt;company_logo&lt;/name&gt;</code>&lt;br&gt;  <code>&lt;value&gt;jaspersoft-logo.png&lt;/value&gt;</code>&lt;br&gt;  <code>&lt;/init-param&gt;</code>&lt;br&gt;  This parameter specifies the image to use when show_logo is set to true. The picture must be located under <code>&lt;liferay-install&gt;/tomcat-dir&gt;/webapps/&lt;JasperServerPortlet-x.x.x&gt;/WEB-INF/images/</code>.</td>
</tr>
<tr>
<td>company_logo_hyper_link</td>
<td><code>&lt;init-param&gt;</code>&lt;br&gt;  <code>&lt;name&gt;company_logo_hyper_link&lt;/name&gt;</code>&lt;br&gt;  <code>&lt;value&gt;http://www.jaspersoft.com&lt;/value&gt;</code>&lt;br&gt;  <code>&lt;/init-param&gt;</code>&lt;br&gt;  This parameter specifies the hyperlink URL to request when a user clicks the company logo.</td>
</tr>
<tr>
<td>report_directory</td>
<td><code>&lt;init-param&gt;</code>&lt;br&gt;  <code>&lt;name&gt;report_directory&lt;/name&gt;</code>&lt;br&gt;  <code>&lt;value&gt;/&lt;/value&gt;</code>&lt;br&gt;  <code>&lt;/init-param&gt;</code>&lt;br&gt;  This parameter specifies a folder in the repository that should be used to populate the Report List page in the portlet.</td>
</tr>
<tr>
<td>number_of_reports_per_page</td>
<td><code>&lt;init-param&gt;</code>&lt;br&gt;  <code>&lt;name&gt;number_of_reports_per_page&lt;/name&gt;</code>&lt;br&gt;  <code>&lt;value&gt;5&lt;/value&gt;</code>&lt;br&gt;  <code>&lt;/init-param&gt;</code>&lt;br&gt;  This parameter specifies the pagination limit for the Report List page. It specifies how many reports to display per page. When there are more reports than the specified number, end users must click a link to go to next page.</td>
</tr>
</tbody>
</table>
This table shows examples of the updated files and values that must be edited to enable Liferay integration with JasperReports Server:

<table>
<thead>
<tr>
<th>Examples of Liferay-related Files and Values</th>
<th>Code &amp; Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>File</strong></td>
<td></td>
</tr>
</tbody>
</table>
| portlet.xml                                  | `<init-param>`
|                                             |   `<name>jasperserver_repository_ws_url</name>`
|                                             |   `<value>http://123.45.6.789:8080/jasperserver-pro/services/repository</value>`
|                                             |   `<!-- You can also use the value localhost or 127.0.0.1 -->` |
|                                             |   `</init-param>` |
| Default Location: `<liferay-install>/tomcat-dir>/webapps/`<br>`<JasperServerPortlet-x.x.x>/WEB-INF/` |                |
| applicationContext-security                  | `<property name="trustedIpAddress">`<br>`<list>`<br>`<value>123.45.6.789</value>`<br>`<!-- You can also use the value localhost or 127.0.0.1 -->`<br>`</list>`<br>`</property>` |
| By default, this file is found at `<js-install>/webapps/jasperserver-pro/WEB-INF/` |                |
| Browser URL                                  | Users point their browsers to http://Liferay_host:7080/c/portal. |

**B.10 Setting up JasperReports Library Hyperlinks for Use in a Portlet Environment**

Because the portal environment is different than that of JasperReports Server when runs stand-alone, the behavior of links in reports is also different. Although any valid JasperReports Library link is rendered in the portal environment, the hyperlinks behave differently than if the report is run directly in JasperReports Server. For example, some hyperlinks might become static text when running in a portal server.

This section assumes that you are using JasperSoft iReport Designer to set up the hyperlinks.
The JasperReports Server portlet supports these types of links:

<table>
<thead>
<tr>
<th>Hyperlink</th>
<th>Reference Tab</th>
<th>Link Parameters Tab</th>
<th>ToolTip Tab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points to an external URL when the Hyperlink Type is CUSTOM or Reference</td>
<td>Hyperlink Reference Expression value must start with http:// or https://. When users click the link in a portlet, the current browser window displays the new web page. The user leaves the portal environment. If Hyperlink is invalid, static text is rendered without links.</td>
<td>Parameters and values that the target web site receives.</td>
<td>Tooltip for the hyperlink.</td>
</tr>
<tr>
<td>Points to a particular page of the same report when the Hyperlink Type is LocalPage.</td>
<td>Hyperlink Reference Expression value specifies the target page number of the current report.</td>
<td>Parameters and values that the target web site receives. For multiple-valued parameters, use the same parameter name for each parameter value.</td>
<td>Tooltip for the hyperlink.</td>
</tr>
<tr>
<td>Points to a particular page of a different report when the Hyperlink Type is RemotePage.</td>
<td>Hyperlink Reference Expression is used in specification of the target page URI.</td>
<td>Hyperlink Page Expression value specifies the target page number of the target report.</td>
<td>Tooltip for the hyperlink.</td>
</tr>
</tbody>
</table>

Notes:
- If type None is chosen, no hyperlink is rendered.
- For both Hyperlink Target values Self and Blank, the target report is displayed in the same portlet window.
- For Hyperlink Type RemoteAnchor and LocalAnchor, no links are rendered.
APPENDIX C  LOCALIZATION

By default, JasperReports Server is presented in the English language (US version), but it supports other languages, as well, with translation packs that include data formats and resource bundles. The supported languages are Chinese (Simplified), French, German, Japanese, and Spanish. The translation packs are included in your JasperReports Server instance by default; to view the application in a specific locale, select it before logging in.

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 a few examples.

For information about localizing Domains, Topics, and reports, refer to the JasperReports Server User Guide.

This chapter contains the following sections:

- Configuring JasperReports Server for the Default Multi-byte Fonts
- UTF-8 Configuration
- Creating a Locale
- Configuring JasperReports Server to Offer a Locale
- Character Encoding and Fonts
- JasperBabylon

C.1  Configuring JasperReports Server for the Default Multi-byte 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.

C.2  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.
C.2.1 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.

To configure Tomcat to support UTF-8:
1. Open the conf/server.xml file and locate the following code:

```xml
<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:

```xml
URIEncoding="UTF-8"
```

3. For example, after your changes, the section might read:

```xml
<!-- 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.

C.2.2 JBoss

Since JBoss uses Tomcat as its web connector, a configuration change like the one you made for Tomcat (C.2.1, “Tomcat,” on page 134) also has to be made for JBoss. In this case, the server.xml file is located in the Tomcat deployment directory, typically server/default/deploy/jbossweb-tomcat55.sar. Make the same code changes, then restart JBoss.

C.2.3 PostgreSQL

JasperReports Server requires PostgresSQL 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';
```
C.2.4 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 WEB-INF/context.xml. The following is a sample resource definition from that file:

```xml
<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 ignores the context.xml file, instead requiring an XML file to define JNDI data sources in the deployment directory, which is typically server/default/deploy. The following is an example of a resource definition in one of those XML files:

```xml
<local-tx-datasource>
  <jndi-name jdbc/jasperserver /> 
  <connection-url>
    jdbc:mysql://localhost/jasperserver?useUnicode=true&amp;characterEncoding=UTF-8
  </connection-url>
  <driver-class com.mysql.jdbc.Driver /> 
  <user-name jasperadmin /> 
  <password jasperadmin /> 
  <min-pool-size 5 /> 
  <max-pool-size 20 /> 
  <idle-timeout-minutes 0 /> 
  <metadata>
    <type-mapping mySQL /> 
  </metadata>
</local-tx-datasource>
```

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
```

C.2.5 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. As of JasperServer 1.2, 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 found at: [http://www.oracle.com/technology/tech/globalization/pdf/TWP_NCHAR_MIGRATION_10gR2.pdf](http://www.oracle.com/technology/tech/globalization/pdf/TWP_NCHAR_MIGRATION_10gR2.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 bin/setclasspath.sh (Linux) or bin/setclasspath.bat (Windows):

1. Locate the following line in the script:

<table>
<thead>
<tr>
<th>Linux</th>
<th># Set the default -Djava.endorsed dirs argument</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>rem Set the default -Djava.endorsed dirs argument</td>
</tr>
</tbody>
</table>

2. Add the following line before it:

<table>
<thead>
<tr>
<th>Linux</th>
<th>JAVA_OPTS=&quot;$JAVA_OPTS &quot;-Doracle.jdbc.defaultNChar=true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>set JAVA_OPTS=%JAVA_OPTS% -Doracle.jdbc.defaultNChar=true</td>
</tr>
</tbody>
</table>

Since JBoss also uses JAVA_OPTS to pass options to the JVM, you can add the same JAVA_OPTS line to bin/run.sh (Linux) and bin/run.bat (Windows). Add it before this line:

<table>
<thead>
<tr>
<th>Linux</th>
<th># Setup the java endorsed dirs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>rem Setup the java endorsed dirs</td>
</tr>
</tbody>
</table>

### C.3 Creating a Locale

Translation is only one aspect of localization. Creating a locale includes these tasks:

- Translating labels and messages.
- Changing date formats.
- Changing format masks.

The tasks in this section require you to edit these files:

<table>
<thead>
<tr>
<th>File</th>
<th>Location</th>
<th>Purpose of Edits</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.properties files</td>
<td>WEB-INF/bundles</td>
<td>Translating labels and messages</td>
</tr>
<tr>
<td>jasperserver_config.properties</td>
<td>WEB-INF/bundles</td>
<td>Changing date formats</td>
</tr>
<tr>
<td>adhoc_masks</td>
<td>WEB-INF/bundles</td>
<td>Changing format masks</td>
</tr>
</tbody>
</table>

### C.3.1 About Properties Files

JasperReports Server and Jaspersoft OLAP resource bundle files are found in the WEB-INF/bundles directory. JasperReports Server includes a default locale (for example, jasperserver_messages.properties), which is written in the U.S. English.

A resource bundle consists of the *.properties files that contain all the labels and messages used in JasperReports Server and Jaspersoft OLAP, customized for a particular locale.
If you use the JasperReports Server portlet to display JasperReports Server content in a portal (such as Liferay), the deployed portlet includes properties files, as well:

### Files in Default Resource Bundles

<table>
<thead>
<tr>
<th>Component Using the File</th>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JasperReports Server</td>
<td>adhoc_messages.properties</td>
<td>Labels and messages specific to JasperReports Server Professional and Enterprise; includes text for the Ad Hoc Editor.</td>
</tr>
<tr>
<td></td>
<td>jasperserver_messages.properties</td>
<td>Core labels and messages used in JasperReports Server.</td>
</tr>
<tr>
<td></td>
<td>pro_nav_messages.properties</td>
<td>Labels and messages specific to JasperReports Server Professional and Enterprise; includes text for the Home page and menu bar.</td>
</tr>
<tr>
<td></td>
<td>LicenseMessages.properties</td>
<td>Labels and messages used by JasperReports Server when validating licenses.</td>
</tr>
<tr>
<td></td>
<td>jasperserver_config.properties</td>
<td>Configuration properties for dates and date times.</td>
</tr>
<tr>
<td></td>
<td>calendar.properties</td>
<td>Labels and messages used by the calendar control.</td>
</tr>
<tr>
<td>Jaspersoft OLAP</td>
<td>ja_mondrian.properties</td>
<td>Labels and messages used in Jaspersoft OLAP.</td>
</tr>
<tr>
<td></td>
<td>ja-pro_messages.properties</td>
<td>Labels and messages specific to Jaspersoft OLAP Professional and Enterprise; includes text for the Professional and Enterprise user interface.</td>
</tr>
<tr>
<td></td>
<td>jpivot_messages.properties</td>
<td>Labels and messages used in Jaspersoft OLAP.</td>
</tr>
<tr>
<td></td>
<td>Mondrian_exception_messages.properties</td>
<td>MDX validation error messages specific to the internal analysis engine.</td>
</tr>
</tbody>
</table>

If you use the JasperReports Server portlet to display JasperReports Server content in a portal (such as Liferay), the deployed portlet includes properties files, as well:

<table>
<thead>
<tr>
<th>File</th>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jaspersoft_portlet_message. properties</td>
<td>WEB-INF/bundles under the location where the portlet is deployed. For example: C:\liferay\webapps&lt;portlet_context_name&gt;\WEB-INF\bundles (where &lt;portlet_context_name&gt; is the name specified when the JasperReports Server portlet was deployed).</td>
<td>Labels and messages that appear in the portlet, including the help text. Note that this does not include text in specific reports; to localize the report's content, you must upload its resource bundle to the repository.</td>
</tr>
</tbody>
</table>

Jaspersoft iReport Designer (iReport) and the iReport Plug-in for JasperReports Server (iReport plug-in) have their own resource bundles, including:

<table>
<thead>
<tr>
<th>Component</th>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iReport</td>
<td>ireport.properties</td>
<td>Labels and messages used in Jaspersoft iReport Designer.</td>
</tr>
</tbody>
</table>

C.3.2 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:

<table>
<thead>
<tr>
<th>File Type</th>
<th>File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default resource bundle</td>
<td>jasperserver_messages.properties</td>
</tr>
<tr>
<td>English</td>
<td>jasperserver_messages_en.properties</td>
</tr>
<tr>
<td>French</td>
<td>jasperserver_messages_fr.properties</td>
</tr>
<tr>
<td>French in Switzerland</td>
<td>jasperserver_messages_fr_CH.properties</td>
</tr>
</tbody>
</table>

For a list of Java-compliant locales, please refer to the Java web site.

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).

The data format masks described in this section are used in the Domains and in the Ad Hoc Editor; they appear in Ad Hoc reports as well as JRXML reports based on Domains; they are not applicable to Jaspersoft OLAP.

C.3.3 Changing Format Masks and Date Formats

Each locale may have its own format masks and rules for displaying dates and datetime values. This section describes the steps you must take to update these options for your new locale.

The data format masks described in this section are used in the Domains and in the Ad Hoc Editor; they appear in Ad Hoc reports as well as JRXML reports based on Domains; they are not applicable to Jaspersoft OLAP.

C.3.3.1 Changing Date and Datetime Formats

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:

```java
    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.

### C.3.3.2 Changing Data Format Masks

The Ad Hoc Editor allows you to create custom reports based on a Topic or Domain. Such reports support localizable data format masks that determine how values appear. 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`.

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:

<table>
<thead>
<tr>
<th>Data Format Mask Properties</th>
<th>Appearance in en_US Locale</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADH_100_MASK_date_0 = short,hide</td>
<td>3/31/09</td>
</tr>
<tr>
<td>ADH_100_MASK_date_1 = long,hide</td>
<td>Mar 31, 2009</td>
</tr>
<tr>
<td>ADH_100_MASK_date_2 = short,medium</td>
<td>March 31, 2009</td>
</tr>
<tr>
<td>ADH_100_MASK_date_3 = medium,medium</td>
<td>Mar 31, 2009 23:59:59</td>
</tr>
<tr>
<td>ADH_100_MASK_int_0 = #,##0</td>
<td>-1,234</td>
</tr>
<tr>
<td>ADH_100_MASK_int_1 = 0</td>
<td>-1234</td>
</tr>
<tr>
<td>ADH_100_MASK_int_2 = $#,#00;($#,###0)</td>
<td>($1,234)</td>
</tr>
<tr>
<td>ADH_100_MASK_int_3 = #,##0;(#,##0)</td>
<td>(1234)</td>
</tr>
<tr>
<td>ADH_100_MASK_dec_0 = #,##0.00</td>
<td>-1,234.56</td>
</tr>
<tr>
<td>ADH_100_MASK_dec_1 = 0</td>
<td>-1234</td>
</tr>
<tr>
<td>ADH_100_MASK_dec_2 = $#,#0.00;($#,###0.00)</td>
<td>($1,234.56)</td>
</tr>
<tr>
<td>ADH_100_MASK_dec_3 = $#,#00;(#,###0)</td>
<td>($1,234)</td>
</tr>
</tbody>
</table>

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:

```java
    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, refer to Sun’s website:

- [http://download.oracle.com/javase/6/docs/api/java/text/DateFormat.html](http://download.oracle.com/javase/6/docs/api/java/text/DateFormat.html)

By default, monetary values in Ad Hoc reports 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.

### C.4 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:

<table>
<thead>
<tr>
<th>File Name</th>
<th>Location</th>
<th>Purpose of Edits</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationContext-security.xml</td>
<td>WEB-INF</td>
<td>Specifying additional locales</td>
</tr>
<tr>
<td>jasperserver-servlet.xml</td>
<td>WEB-INF</td>
<td>Specifying additional time zones</td>
</tr>
</tbody>
</table>

#### C.4.1 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 locales from which they choose is defined in `applicationContext-security.xml`. Edit this file to add a new locale.
To add a new locale:
1. Edit the applicationContext-security.xml file and locate the bean named userLocalesList. For example:

```xml
<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):

```xml
<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 web site.

C.4.2 Specifying Additional Time Zones

By default, JasperReports Server assumes the user’s time zone is the same as the time zone of the JasperReports Server host. However, the Login page allows users to choose a different time zone. The list from which they choose is defined in applicationContext.xml file.

To add a time zone:
1. Open the applicationContext.xml file and locate the userTimeZonesList bean. For example:

```xml
<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.

For more information about Java-compliant time zones, please refer to the Java web site.

### C.4.3 Setting a Default Time Zone

If you want JasperReports Server to use a time zone that is different from the host computer, you can set a specific time zone in Java. It becomes the default time zone for all users, but they may 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.

<table>
<thead>
<tr>
<th>JVM Settings for Default Time Zone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
</tr>
<tr>
<td>--------------------</td>
</tr>
<tr>
<td>Windows</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Linux</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Both</td>
</tr>
</tbody>
</table>

### C.5 Character Encoding and Fonts

Depending on the third-party software you use and the locales you support, you may also have to configure JasperReports Server and its host. The steps described in this section are only necessary under certain circumstances, such as if you plan to use a character encoding form that UTF-8 cannot handle or if you need to change the font options for Jaspersoft OLAP charts.

The tasks in this section require you to edit these files:

<table>
<thead>
<tr>
<th>File Name</th>
<th>Location</th>
<th>Purpose of Edits</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationContext.xml</td>
<td>WEB-INF</td>
<td>Changing character encoding</td>
</tr>
<tr>
<td>jpivot_internal_messages.properties</td>
<td>WEB-INF/internal</td>
<td>Specifying chart fonts for Jaspersoft OLAP Community</td>
</tr>
<tr>
<td>Ja_pro_internal_messages.properties</td>
<td>WEB-INF/internal</td>
<td>Specifying chart fonts for Jaspersoft OLAP Professional and Enterprise</td>
</tr>
<tr>
<td>userConfig.xml</td>
<td>WEB-INF/pivot/print</td>
<td>Embedding fonts in PDF</td>
</tr>
</tbody>
</table>
C.5.1 Changing Character Encoding

To use a character encoding form other than UTF-8, you must configure JasperReports Server, your application server, and your database server.

C.5.1.1 Configuring JasperReports Server

To configure JasperReports Server for a different encoding form, you must edit the applicationContext.xml file.

To specify a different encoding form:
1. Open the applicationContext.xml file and locate the following bean. It is configured for UTF-8:

   ```xml
   <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:

   ```xml
   <bean id="encodingProvider" class="com.jaspersoft.jasperserver.api.common.util.StaticCharacterEncodingProvider">
       <constructor-arg value="UTF-16"/>
   </bean>
   ```

3. Save the file.

C.5.1.2 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 associated with your application server and database server. For Tomcat, you can specify a different character encoding by following steps similar to those described in C.2.1, “Tomcat,” on page 134 and C.2, “UTF-8 Configuration,” on page 133.

This step is necessary only if you plan to support locales that requires a different character encoding, such as UTF-16. In addition to this change, your application server and database must be configured to use the character encoding you require. For more information, refer to the documentation associated with your third party software.

C.5.1.3 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:
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:

   ```sql
   create database foodmart_ja encoding='utf8';
   ```
2. The URL of any OLAP data source that JasperReports Server accesses must be properly configured in the /ji-pro/META-INF/context.xml file. For example, the URL definition for the Foodmart sample database might be similar to the following:

```xml
<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:

```sql
jdbc:postgresql://localhost:5432/foodmart_ja
```

### C.5.2 Working with Fonts

While the fonts that 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 rather than to the application. This section describes steps you may need to take, depending on the functionality you use and the locales you support.

#### C.5.2.1 Enabling East Asian Fonts

The default JRE configuration 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 the computers’ JRE.

**To configure a Microsoft Windows computer (XP and later) for East Asian fonts:**

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 the following:

    ```
    fontconfig.properties
    ```
C.5.2.2 Configuring 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:
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

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.

C.5.2.3 Embedding Fonts in PDF Output

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 of your reports, 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 save reports 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 documentation associated with FOP, which is available at:

http://xmlgraphics.apache.org/fop/

You can embed any Unicode font using this procedure, though larger font files may have significantly larger memory footprints. In order to keep memory requirements small, Jaspersoft recommends that you use the smallest font file you can, such as SimHei to support Chinese, Japanese, and Korean.
C.6 JasperBabylon

JasperBabylon allows Jaspersoft’s open source community to edit and share locale resource bundles. The repository is public, and includes translations for several applications (notably, Jaspersoft iReport Designer, iReport Plug-in for JasperReports Server, and the open source version of JasperReports Server, which is JasperReports Server Community Project). You can access the repository by pointing your browser to the JasperBabylon web site:

http://www.jasperforge.org/jasper babylon

More information, including usage instructions, is available on the JasperBabylon web site. To make updates on this site, you must sign up for a contributor ID, which is free.
Ad Hoc Editor
The interactive report designer in JasperReports Server Professional and Enterprise editions. Starting from a collection of fields predefined in a Topic or selected from a Domain, the Ad Hoc Editor lets you drag and drop report elements to draft, preview, and finalize reports. Like JRXML reports, Ad Hoc reports can be run, printed, and scheduled within JasperReports Server. In addition, Ad Hoc reports may be reopened in the Ad Hoc Editor, further modified, and saved.

Audit Archiving
To prevent audit logs from growing too large to be easily accessed, the system 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 private 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 a Domain, a field whose value is calculated from a user-written formula that may include any number of fields, operators, and constants. A calculated field is defined in the Domain Designer, and it becomes one of the items to which the Domain’s security file and locale bundles can apply.

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

CrossJoin
An MDX function that combines two or more dimensions into a single axis (column or row).
Cube
The basis of most 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 parameterize the data to display. For example, you can narrow down the data to a specific date range. Embedded web content, such as other web-based applications or maps, make dashboards more interactive and functional.

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

Data Policy
In 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 > Ad Hoc Settings.

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.

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.

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**

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

**Group**

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

**Hierarchy Level**

In 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.

**iReport Designer**

An open source tool for graphically designing reports that leverage all features of the JasperReports Library. The Jaspersoft iReport Designer lets you drag and drop fields, charts, and sub-reports into a canvas, and also define parameters or expressions for each object to create pixel-perfect reports. iReport Designer outputs the JRXML of the report or uploads it directly to JasperReports Server.
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.

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 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.

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 iReport Designer and JasperReports Server. JRXML is an open format that uses the XML standard to define precisely all the structure and configuration of a report.

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 used to populate an OLAP view.

Mondrian Schema Editor
An open source Eclipse plug-in for creating Mondrian OLAP schemas.

Mondrian XMLA Source
A server-side XMLA source definition of a remote client-side XML/A connection used to populate an OLAP view using the
XMLA standard.

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

Navigation Table
The main table in an 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 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.
It is the entry point to analysis operations, such as slice and dice, drill down, and drill through.

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 manage all of their accounts, roles, and repository objects. The default organization admin in each organization is the
jasperadmin account.
Outlier
A fact that seems incongruous when compared to other member’s facts. For example, a very low sales figure or a very high number of helpdesk tickets. Such outliers may indicate a problem (or an important achievement) in your business. 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 \( \text{†} \).

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.

Repository
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 iReport. Applications can access the repository through the web service API. Administrators use the import and export utilities to back up the repository contents.

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.

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.

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.

WCF

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 populate an OLAP view.
## INDEX

### Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;theme</td>
<td>parameter 87</td>
</tr>
</tbody>
</table>

### A

**About JasperReports Server window** 18  
access control  
  - administering 42  
  - authentication 15  
  - authorization 16  
  - object-level permissions 44  
  - permissions 42  
  - repository 12  

**Ad Hoc**  
  - caching 99  
  - comparison with Jaspersoft OLAP caching 102  
  - data formats 139  
  - data policies 97  
  - date formats 139  
  - fetch time 100  
  - queries 96, 99  
  - query time 100  
  - reports 96–99, 101, 117, 119  
  - result sets 99  

adding  
  - bean data sources 52, 53  
  - datatypes 57  
  - folders 36  
  - JDBC data sources 48  
  - JNDI data sources 50  
  - lists 58  
  - queries 54  
  - resources 37  

**administering JasperReports Server**  
  - See also access control.  
  - Ad Hoc 96–99  
  - auditing 115  
  - bean data sources 52  
  - chart themes 108  
  - configuration files 93  
  - crosstab limit 108  
  - data policies 96, 97  
  - data sources 47  
  - datatypes 57  
  - Domain validation 107  
  - folders 36  
  - fonts 71  
  - general tasks 11  
  - heartbeat 110  
  - HTML exporter 109  
  - HTML5 for Pro Charts 109  
  - input controls 56  
  - JAR files 71  
  - JasperReports Library 107  
  - JasperReports Server configuration 99  
  - JDBC data sources 48  
  - JNDI data sources 50  
  - levels of administration 10  
  - Liferay 125  
  - lists 58  
  - localization 133, 136, 140  
  - logging in as a user 46  
  - multiple administrators 12, 19  
  - object-level permissions 44  
  - online help 113  
  - organizations 10, 19, 40  
  - passwords 94, 96  
  - portlets 128  
  - queries 54, 96  
  - report intervals 110  
  - repository 11, 31, 41  
  - resource bundles 72  
  - resources 37  
  - roles 14, 27  
  - summary of tasks 19  
  - Talend Integration Suite 28
users 14, 15, 19, 23
XHTML exporter 109
administrators
  general tasks 11
  levels of administration 10
  organization administrator 19
  system administrator 10
Adobe Flash 109
Apache Hadoop. See Hive-Hadoop data sources.
Apple iOS 109
Asian fonts 133, 144
attributes, profile. See profile attributes.
auditing
  administering 115
  archiving audit events 115, 117
  audit Domains 117
  audit reports 117, 119
  audited events 115
  configuring 116
  importing and exporting audit data 119
authentication. See access control.
authorization. See access control.
B
beans. See data sources.
big data 51, 98
browsing the repository 13
C
caching. See Ad Hoc.
cascading input controls 61
character encoding
  See also fonts, locales, and localization.
  encoding forms 143
  JBoss 134, 135, 136
  JDBC connections 144
  JDBC data sources 135
  JDBC drivers 135
  JNDI data sources 135
  MySQL 135
  non-UTF-8 character encoding 142
  OLAP data sources 144
  OLAP views 143
  Oracle databases 135
  PostgreSQL 134
  Tomcat 134, 136, 143
  Unicode databases 143
  UTF-8 133
charts, themes for 108
CLOB 112
copying
  folders 38
  resources 13, 38
creating
datatypes 57
folders 36
input controls 59
lists 58
queries 54
resources 13
crosstab report, Out of Memory errors 108
CSS files 78, 85
custom data sources 48
D
data formats 139
data policies 96, 97
data sources
  administering 47
  bean 52
  custom 48
  JDBC 48
  JNDI 50
  queries 55
dataset caching 99
datatypes
  administering 57
  and input controls 59, 60
  creating 57
  for input controls 57
  types 57
date formats 138, 139
defaults
  changing 93
  chart themes 108
  Domain validation 107
  installation 10
  Liferay ports 125
  locale 136
  Oracle character sets 135
  Oracle synonyms 111
  PDF fonts 145
  report interval scheduling option 110
  roles 27
  time zone 142
  users 23
deleting
  folders 40
  resources 13, 40
demo. See Supermart.
Domains
  adapting reports to audiences 54
  audit Domains 117
  JDBC column types 113
  validation 107
E
editing folders and resources 37
events. See logging.
exporting
audit data 119
overview 89
resources 90

F
Firebug plug-in 86
Flash 109
Folder Template 21, 85
folders
creating 36
deleting 40
editing 37
Folder Template 21
moving 38
Public 12
fonts
See also character encoding, locales, and localization.
administering 71
East Asian 133, 144
in the repository 71
Jaspersoft OLAP 145
localization 144
multi-byte 133
non-UTF-8 fonts 142
OLAP views 145
PDF files 145
troubleshooting in exported files 71
format masks 138
Fusion Charts 109

H
Hadoop-Hive data sources 51
heartbeat 17, 110
help configuration 113
Hive Query Language 48
HiveQL 56
HTML exporter 109
HTML5 109

I
importing
audit data 119
overview 89
resources 91
input controls
adding 59
administering 56
and datatypes 57, 59, 60
cascading 61
creating 59
list of values for 58
list type 58
parameters for cascading input controls 67, 68
parameters for query-based input controls 67
query-based 61
saving 58, 59, 61
types 57

internationalization. See localization.

iReport. See Jaspersoft iReport Designer.

J
JAR files, administering 71
JasperAnalysis. See Jaspersoft OLAP.
JasperBabylon 146
JasperETL 28
JasperReports Library
configuring 107
creating reports 9
extending 107
setting up hyperlinks 131
structure of a JasperReport 33
JasperReports Server
configuration 99
plug-in for iReport 68
Jaspersoft iReport Designer
chart themes 108
creating reports with 9
resource bundles 137
setting up JasperReports Library hyperlinks 131
Jaspersoft OLAP 27, 102, 145
JBoss 134
JDBC data sources 48
JNDI data sources 50
JRXML files
reference syntax. See repository.
resource bundles 72
js-export command 90
js-import command 91

L
languages. See localization.
license expiration 18
Liferay
configuring 125
elements 131
testing the configuration 127
lists
administering 58
creating 58
list type input controls 58
locales
See also character encoding, fonts, and localization.
administering 140
character encoding 142
creating 136
data formats 139
date formats 138, 139
format masks 138
non-UTF-8 character encoding 143
properties 136
resource bundles 136, 146
time zones 141
localization
   See also character encoding, fonts, and locales.
   fonts 144
   JasperBabylon 146
   JasperReports Server 136
   Jaspersoft iReport Designer 137
   OLAP views 143, 145
   UTF-8 133
logging
   audit events. See auditing.
   system events 103
logging in as a user 46
M
maxFilterValues 103
mdxDataStrategy 103
moving
   folders 38
   resources 13, 38
   multi-byte fonts 133
multiple organizations. See organizations.
MySQL 97, 135
N
NVARCHAR2 113
O
OLAP views 16, 143, 145
online help 113
Oracle
   character sets 135
   CLOB 112
   NVARCHAR2 113
   synonyms 111
organizations
   admin roles 27
   administering 19
   default 10
   folder structure 21
   multiple 10, 19, 40
   Organizations folder 21
   Portal organizations 127
   searching for 20
   single 10, 19
   suborganizations 10
Organizations folder 21
Out of Memory errors 108
overrides_custom.css file 79
P
parameterized queries. See queries.
passwords
   auditing 118
   encrypting 96
   expiration 94
   users changing 95
PDF
   embedding fonts 145
   troubleshooting fonts in 71
permissions. See access control.
plug-in for iReport 68
port numbers 125
portlets
   and hyperlinks in reports 131
   and Liferay 126
   and multiple organizations 129
   and web services 126
   changing the default report list 130
   changing the logo 129, 130
   configuring 128, 129
   deploying 127
   integrating Liferay 125
   parameters 129
   port numbers 125
   role 27, 129
   testing 129
   upgrading JasperReports Server 127
PostgreSQL 134
Pro Charts 109
profile attributes
   and auditing 116
   and query-based input controls 65
   defined 16
   updating in the database 26
   viewing users’ 24
properties files
   creating 136
   resource bundles 138
Public folder 12
Q
queries
   administering 54
   and Ad Hoc reports 96, 99
   creating 54
   multiple query languages 56
   parameters for cascading input controls 67
   parameters for query-based input controls 67
   query executors 56
   saving 56
   using 54
query-based input controls 61
R
repo syntax 33
reports
   and cascading input controls 70
   creating 9
   JasperReport 33
   JasperReports Library 9
   Jaspersoft iReport Designer 9
   report intervals 110
   scheduling 110
   troubleshooting fonts in exported reports 71
repository
See also resources.
access control 12
administering 11, 31
browsing 13
configuring 41
design issues 41
encryption 96
events 118
folders 11
importing and exporting 89
Organizations folder 21
Public folder 12
reference syntax 33
referencing resources 41
resources 12, 33
sample data 12
searching 13, 28
structure 11
UTF-8 encoding 134, 135
resource bundles
in the repository 72
See also localization.
resources
adding 13, 37
browsing 13
copying 13
cutting 13
deleting 13, 40
editing 37
fonts 71
importing and exporting 89
in repository 33
JAR files 71
moving 38
pasting 13
queries 54
resource bundles 72
resource references in JRXML files. See repository.
searching 13
result sets 99
roles
administering 14, 27
administrative 19
and users 27
default roles 27
object-level permissions 44
searching for 28
S
saving
input controls 58, 59, 61
queries 56
searching the repository 13, 28
session timeout 95
suborganizations. See organizations.
SuperMart demo and roles 27
support, finding product version 18
system admin roles 27
system logging 103
T
Talend Integration Suite (TIS EE) 28
themes
See also chart themes.
&theme parameter 87
active theme 77
CSS files 78, 85
custom overrides 79
default theme 77
downloading 84
Firebug plug-in 86
in the Folder Template 85
inheritance mechanism 79
inherited theme 77
Jaspersoft iReport Designer 108
permissions 82
recovering from inadvertent changes 87
sample galleries 87
system theme 77
Themes folder 76
uploading 84
URL parameter 87
ZIP archive 83
third-party APIs 71
time zones 141
Tomcat 134
translation packs 133
U
Unicode Transformation Format (UTF-8). See character encoding.
Universal Resource Identifiers (URI) 41
user interface samples 87
users
administering 14, 15, 23
authenticating 15, 16, 42
changing passwords 95
default users 23
logging in as 46
object-level permissions 44
roles 27
searching for 24
session timeout 95
V
version of the software 18
views 16
W
web services 117, 126
X
XHTML exporter 109