

CA Identity Manager

Connector Xpress Guide

r12



Second Edition

This documentation and any related computer software help programs (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by CA at any time.

This Documentation may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of the Documentation for their own internal use, and may make one copy of the related software as reasonably required for back-up and disaster recovery purposes, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the provisions of the license for the Product are permitted to have access to such copies.

The right to print copies of the Documentation and to make a copy of the related software is limited to the period during which the applicable license for the Product remains in full force and effect. Should the license terminate for any reason, it shall be the user's responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

EXCEPT AS OTHERWISE STATED IN THE APPLICABLE LICENSE AGREEMENT, TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO THE END USER OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED OF SUCH LOSS OR DAMAGE.

The use of any product referenced in the Documentation is governed by the end user's applicable license agreement.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Copyright © 2009 CA. All rights reserved.

CA Product References

This document references the following CA products:

- CA Identity Manager
- CA SiteMinder® Web Access Manager
- CA Security Command Center (SCC)
- CA Audit
- eTrust® Directory, also known as CA Directory

Contact CA

Contact Technical Support

For your convenience, CA provides one site where you can access the information you need for your Home Office, Small Business, and Enterprise CA products. At <http://ca.com/support>, you can access the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Provide Feedback

If you have comments or questions about CA product documentation, you can send a message to techpubs@ca.com.

If you would like to provide feedback about CA product documentation, complete our short [customer survey](#), which is also available on the CA support website, found at <http://ca.com/support>.

Contents

Chapter 1: Connector Xpress	9
Terminology Changes	9
Connector Xpress Overview	10
Mappings	11
Supported Databases	11
Supported Directories	11
Communication with Other Components	12
Data Sources	13
Set Up Data Sources	13
Utilities	13
Password Caching	14
Connector Xpress Window	14
Provisioning Servers Tree	15
Edit Pane	16
Metadata Editor	17
Operations Bindings Editor	18
 Chapter 2: Installing Connector Xpress	 19
Installation Pre-requisites	19
Install Connector Xpress	19
How You Install the Java CS or Connector Xpress Silently	20
JDBC Naming Attribute	20
Where User Preferences Are Stored	20
Set Preferences	21
Where Settings Are Stored	21
 Chapter 3: Configuration	 23
Tuning the Database Used by the Provisioning Server	23
Provisioning Server Configuration	23
Add and Configure a Provisioning Server	24
Edit Provisioning Server Details	24
Remove a Provisioning Server	25
Connect to a Provisioning Server	25
Connector Server Configuration	26
Create a Connector Server Configuration	26
Edit a Connector Server Configuration	27

Set a Managing Connector Server	28
Remove a Connector Server Configuration	28
Update a Java CS Password	29
Configuration Data Location	29
Java CS Routing Rules Location	29
Endpoint Type Metadata Location	30
Endpoint Type Metadata and Java CS Routing Rules	30
View Endpoint Type Metadata and Java CS Routing Rules	30
Java CS Routing Rules Hierarchy	30
Endpoint Type Distinguished Name	30
Connector Xpress Local Storage	31

Chapter 4: Creating Connectors 33

How To Create and Deploy Connectors	34
Projects	35
How You Use the Connector Xpress Wizard	38
View the Metadata Tree	39
Save the Connector	39
Connector Deployment	39
Connector Undeployment	41
Table Attribute Mappings	41
Mapping Against One Endpoint and Acquiring Against Another Endpoint	42
Map Group Members (JNDI)	43
Type Mapping	43
Edit Metadata	43
Metadata and Operations Bindings Editors Considerations	44
Re-deploy Metadata	44
DYN Schema Extensions	45
Support for Mandatory Attributes on a JDBC Endpoint	46
Metadata Descriptions	50
Stored Procedures	70

Chapter 5: Importing and Exporting Data 75

Import XML Data Model	75
Export Data Model to an XML File	75
Merge a Modified Data Model	76
Mapping Summary File	76
Export a Mapping Summary File	77

Chapter 6: Managing Endpoints **79**

How You Acquire and Manage Endpoints	80
Acquire, Explore and Correlate an Endpoint	81
Create a New Endpoint Type	82
Delete an Endpoint Type	82
Clean an Endpoint Type	83
Delete an Endpoint	83

Chapter 7: Screens and Dialogs **85**

Set Preferences Dialog	85
Select Data Source for new project Dialog	86
Source Types Dialog	87
Edit Source Dialog	87
Explore/Correlate Endpoint Dialog	90
Enter Password for Data Source Dialog	90
Create New Endpoint Type Dialog	91
Create New Endpoint Dialog	91
Provisioning Server Password Required Dialog	92
Edit Connector Server Configuration Dialog	93
Provisioning Server Details Dialog	94
Edit Metadata for Endpoint Type Dialog	95

Chapter 8: Wizard Screens **97**

General Screen	97
Stored Procedures Screen (JDBC Only)	98
Map Account Create pre, op and post screens	99
Map Account Modify pre, op and post screens	101
Map Account Delete pre, op and post screens	103
Map Account Modify RN pre, op and post screens	105
Map Group Modify pre, op and post screens	107
Map Group Create pre, op and post screens	109
Map Group Delete pre, op and post screens	111
Map Group Modify RN pre, op and post screens	113
Map Account or Group Table Screen	115
Map Account or Map Group Confirm Screen	118
Maximum Length of LDAP String Used to Represent an Attribute's Value	121
Map Group Table Screen	121
Map Membership Table Screen (JDBC Only)	125
Map Containers Screen (JNDI Only)	126

Chapter 9: Examples

127

How Acme Corporation Promotes a Connector from a Test to a Production Environment 127

Chapter 1: Connector Xpress

This section contains the following topics:

[Terminology Changes](#) (see page 9)

[Connector Xpress Overview](#) (see page 10)

[Mappings](#) (see page 11)

[Supported Databases](#) (see page 11)

[Supported Directories](#) (see page 11)

[Communication with Other Components](#) (see page 12)

[Data Sources](#) (see page 13)

[Utilities](#) (see page 13)

[Password Caching](#) (see page 14)

[Connector Xpress Window](#) (see page 14)

Terminology Changes

The following table shows the new Provisioning Manager terminology used in this document that replaces the previous eTrust Admin terminology:

Old eTrust Admin Terminology	New Provisioning Terminology
eTrust Admin Help	Provisioning Manager Help
eTrust Admin	Provisioning Server
eTrust Admin Server	Provisioning Server
eTrust Admin Manager, GUI, Win-32 GUI	Provisioning Manager
Directory, Directories	Endpoint, Endpoints
Namespace	Endpoint Type
Policy, Policies, or Provisioning Policy	Account Template, Account Templates
Roles	Provisioning Roles
Distributed SuperAgent Framework	Connector Server Framework
SuperAgent	C++ Connector Server
Administrative Directory or Administrative Repository	Provisioning Directory
Option	Connector

Connector Xpress Overview

Connector Xpress is a CA Identity Manager utility for managing dynamic connectors, mapping dynamic connectors to endpoints, and establishing routing rules for endpoints. You can use it to configure dynamic connectors to allow provisioning and management of SQL databases and LDAP directories.

Connector Xpress lets you create and deploy custom connectors without the technical expertise generally required when creating connectors managed by the Provisioning Manager.

You can also set up, edit, and remove a connector server configuration (both Java and C++) using Connector Xpress.

The primary input into Connector Xpress is the native schema of an endpoint system. For example, you can use Connector Xpress to connect to a RDBMS and retrieve the SQL schema of the database. You can then use Connector Xpress to construct mappings from those parts of the native schema that are relevant to identity management and provisioning. A mapping describes how an element of the native schema is represented in the provisioning layer.

Connector Xpress generates metadata that describes, to a dynamic connector, the runtime mappings to a target system.

The output of Connector Xpress is a metadata document produced when you complete your mappings. The metadata is an XML file that describes the structure of your connector to the Java CS.

It describes the Provisioning Server classes and attributes and how they are mapped to the native schema.

The metadata is used to create dynamic endpoint types on one or more Provisioning Servers.

Note: For more information on using Connector Xpress, see the *Connector Xpress Guide*, in the *Identity Manager bookshelf*.

Mappings

Mappings consist of the following:

- One or more **class mappings**—Each class mapping describes a single class of provisioning objects in the Provisioning Server's DIT. For JDBC, a class mapping maps to a single database table. For LDAP, a class mapping can map to multiple native LDAP object classes. Account class mappings have some special handling because they are important to Provisioning Server operations like policy syncing.
- Multiple **attribute mappings**—Each class mapping can have multiple attribute mappings, which map individual values in the Provisioning Server's provisioning objects to single values in objects on the managed endpoint system.

Supported Databases

Connector Xpress supports the following databases:

- Oracle
- Microsoft SQL Server
- Ingres
- DB2
- DB2 for z/OS
- Sybase

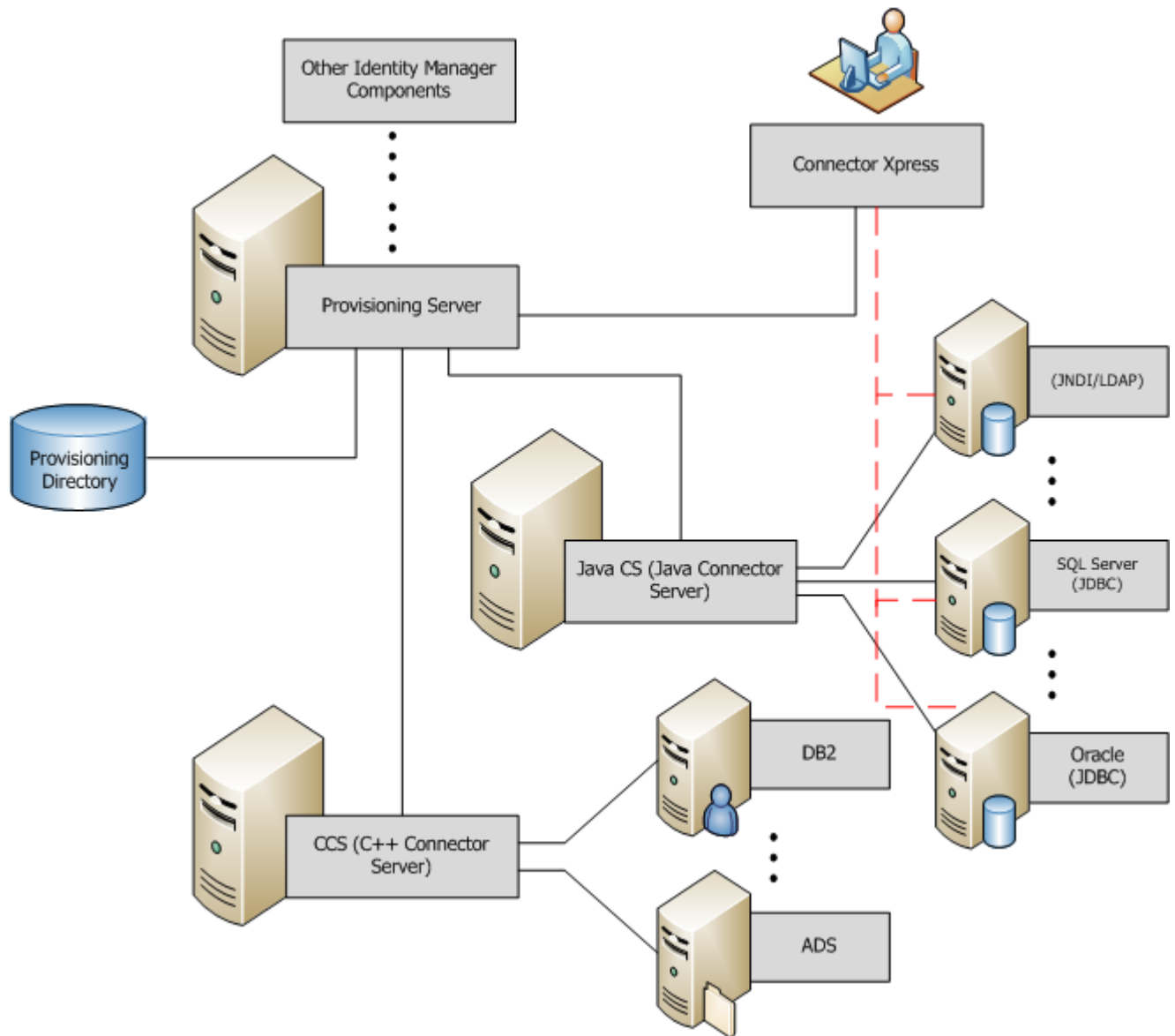
Supported Directories

Connector Xpress supports the following directories:

- CA Directory
- Sun One Directory
- Novell Directory

Communication with Other Components

The diagram below shows the direct and indirect communications between Connector Xpress and other components in the distributed system:



Note: The dashed red lines indicate that the direct communications between Connector Xpress and endpoints are read-only. These occur when setting up data sources.

Data Sources

A data source is a reference to a server or other source of schema information that Connector Xpress can construct a connector for.

Connector Xpress supports the following data sources from supported vendors:

- **JDBC**—Any service accessible via the JDBC API, typically a relational database server.
- **JNDI**—Any service accessible via the JNDI API, typically an LDAP server.

Set Up Data Sources

You can connect to a data source to read its schema and then use Connector Xpress to map the schema to a new Provisioning Server endpoint type. Connector Xpress maintains a list of data sources that you configure.

Note: You can also set up data sources using the Connector Xpress Wizard. When you create a Connector Xpress project, the wizard will prompt you to set up your data sources.

To set up data sources

1. Click Tools, Data Sources.
The Select Data Source for new Project dialog appears.
2. Click Add.
The Source Types dialog appears.
3. Select an available data source type from the list, and then click OK.
The Edit Source dialog appears, specific to the type of data source you are adding.
4. Complete the fields on the dialog, and click OK.
You have successfully configured the data source type you selected.

Utilities

Connector Xpress has the following utilities you can use to create manage connectors:

- Metadata editing and deployment
- Add, populate, clean and remove dynamic endpoints
- Set managing CS for endpoint types
- Acquire, explore, correlate, and remove endpoint systems

More Information:

[Edit Metadata](#) (see page 43)

[Managing Endpoints](#) (see page 79)

[Set a Managing Connector Server](#) (see page 28)

Password Caching

Connector Xpress caches the passwords used with each Provisioning Server only when the application is open. When the application is closed, Connector Xpress does not permanently record passwords. When Connector Xpress needs a password, and the password is not cached, or the cached password causes a failed authentication attempt, Connector Xpress prompts you to enter the password.

Connector Xpress Window

The Connector Xpress window contains the following fields in the upper left pane; depending on the type of project you have open.

Upper left pane

Metadata

Displays the metadata for the currently open project in the Edit pane.

Bindings

Displays the bindings for the currently open project in the Edit pane.

Wizard

Starts the Connector Xpress Wizard.

Note: Connector Xpress displays the upper left pane only when you open a project.

More Information:

[Metadata Editor](#) (see page 17)

[Edit Pane](#) (see page 16)

[Operations Bindings Editor](#) (see page 18)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Provisioning Servers Tree

The Provisioning Servers tree (in the lower left pane) lets you manage the endpoints and connectors of multiple Provisioning Servers. The Provisioning Servers tree contains the following fields:



Refreshes the currently selected node in the Provisioning Tree.

Note: The Refresh button is only available when you select a tree node that has a changeable list of child nodes. For example, the button is not available for a particular endpoint as this type of node always has Policies and Endpoint child nodes only, whereas for the endpoints node of a particular endpoint, the button is available.



Displays the actions you can perform on the currently selected object in the Provisioning Tree. This is equivalent to right-clicking the currently selected object in the Provisioning Tree.

Provisioning Servers node

Lets you add and configure Provisioning Server connection details. When you add a Provisioning Server, Connector Xpress maintains the details of the Provisioning Server so that you can access it each time you run Connector Xpress.

Servers Object node

Displays each registered Provisioning Server on the network. You can use this node to edit or remove the Provisioning Server connection details.

Endpoint Types node

Displays a list of endpoint types. You can use this node to create a new endpoint type.

Endpoints Object node

Lets you deploy and edit metadata, delete an endpoint type, acquire an endpoint, clean an endpoint type, for DYN endpoint types and set the managing connector server only for non-DYN endpoint types.

Endpoints node

Groups endpoints of a particular endpoint type together.

Endpoints Name node

Represents an endpoint that you have acquired on the Provisioning Server. You can use this node to perform an explore or correlate of the endpoint for DYN endpoints, delete an endpoint, or set the managing connector server only for non-DYN endpoints.

Policies node

Groups the policies of a particular endpoint type together.

Note: A default policy is provided for all endpoint types.

CS Config Object node

Displays a list of CS configurations. You can use this node to manually edit the existing configuration for the server's branch DN routing rules, delete the configuration entry for the C++ Connector Server or Java CS, and set the managing connector server password.

CS Configs node

Displays logical groupings of C++ Connector Server and Java CS configurations. These contain the DSFConfig style routing rules for managing branch DNS. You can use this node to add a new routing rules configuration object for a C++ Connector Server or Java CS service.

Edit Pane

The Edit pane (rightmost pane), contains the following fields, depending on which task you are performing:

Task Shortcuts

Identifies the common Connector Xpress tasks for quick access.

Metadata Editor

Lets you edit metadata for the currently open project.

Operations Bindings Editor

Lets you edit bindings for the currently open project.

Connector Xpress Wizard screens

Displays the Connector Xpress Wizard screens.

More Information:

[Metadata Editor](#) (see page 17)

[Operations Bindings Editor](#) (see page 18)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Metadata Editor

You can use the Metadata Editor to manually edit and redeploy the metadata for a dynamic endpoint type.

The Metadata Editor contains the following fields:

XML Tree

Displays an editable XML tree view of the metadata.

Node Type

Specifies the node types that you can add using the Add button. The contents of the drop-down list will depend on the type of node currently selected in the XML tree.

Add

Adds a new node to the XML tree of the type specified in the Node Type drop-down list.

Edit

Edits the currently selected node in the XML tree.

Remove

Removes the currently selected node from the XML tree.

Search

Searches metadata values and names by their native field name or bean property name.

Important! We recommend that you use the Metadata Editor at your own risk. Removal of or alteration of important objects in the metadata may render the metadata inoperable.

More Information:

[Edit Metadata](#) (see page 43)

[Metadata and Operations Bindings Editors Considerations](#) (see page 44)

Operations Bindings Editor

You can use the Operations Bindings Editor to edit or view the stored procedures you have configured for an endpoint type.

Operation Bindings drop down list

Lists the types of operation bindings you can use.

- Method
- Script
- Script function

Add

Adds a new node to the Operations Bindings tree of the type specified in the Operation Binding drop-down list.

Edit

Lets you edit the currently selected node in the Operations Bindings tree.

Remove

Removes the currently selected node from the Operations Bindings tree.

Search

Searches binding values by their procedure parameters or property name.

Important! We recommend that you use the Operations Bindings Editor at your own risk. Removal of or alteration of important objects in the metadata may render the metadata inoperable.

More Information:

[Stored Procedure Column Type Considerations](#) (see page 71)

[Stored Procedure Bindings](#) (see page 72)

Chapter 2: Installing Connector Xpress

This section contains the following topics:

[Installation Pre-requisites](#) (see page 19)

[Install Connector Xpress](#) (see page 19)

[How You Install the Java CS or Connector Xpress Silently](#) (see page 20)

[JDBC Naming Attribute](#) (see page 20)

[Where User Preferences Are Stored](#) (see page 20)

[Set Preferences](#) (see page 21)

[Where Settings Are Stored](#) (see page 21)

Installation Pre-requisites

- CA Identity Manager Server
- CA Identity Manager Provisioning Server

Install Connector Xpress

Important! Be sure to disable all antivirus software before installing Java CS, Java CS SDK, and Connector Xpress.

Note: You do not need to install Connector Xpress on the same computer as the Java CS or any other server component, including the Provisioning Server.

To install Connector Xpress

1. Locate the Identity Manager Provisioning Components download or other media.
2. Start the Product Explorer.
3. Select Install Products, Clients.
4. Select Connector Xpress.
5. Follow the onscreen instructions to complete the installation.

How You Install the Java CS or Connector Xpress Silently

The Java CS and Connector Xpress both support a silent mode of installation. To run a silent install you need to create a response file. The process for the Java CS and Connector Xpress is the same:

1. Do *either* of the following:

- Enter the following command, and then follow the instructions in the file to manually update the parameters in the file.

```
setup -options-template template_file
```

- Enter the following command and run through the installation:

```
setup -options-record response file.
```

The response file records the input parameters.

2. Start the silent installation using one of the previously prepared response files.

Note: You must use fully qualified path names when generating and running with response files. For example, *responsefile.txt* is not valid but *C:\responsefile.txt* is valid.

JDBC Naming Attribute

You can only map the account and group naming attributes to columns which are required in the native table they are mapped to (that is, an attribute that does not allow null values). You are strongly advised to ensure that these columns act as primary keys for the tables containing them, or at the very least, have a unique index applied to them.

Where User Preferences Are Stored

Connector Xpress stores user preferences through the Java Preferences API. On Windows, this stores data in the following Registry key:

```
HKEY_CURRENT_USER\Software\JavaSoft\Prefs
```

On UNIX, this stores data in the user's home directory under the following folder:

```
.java/.userPrefs
```

Note: The uninstaller will not remove this data, so subsequent installs will preserve user preferences.

Set Preferences

You can change various aspects of the way Connector Xpress looks and behaves, such as its look and feel, search limits, and command parameters.

To set preferences

1. Click Tools, Preferences.

The Connector Xpress Preference dialog appears.

2. Complete the following fields on the dialog, and then click OK.

Connector Xpress applies and saves your preferences.

More Information:

[Set Preferences Dialog](#) (see page 85)

Where Settings Are Stored

Settings are stored using the Java Preferences API. The API stores the values in the Registry on Windows and in a file under the user's home directory on Solaris.

On Windows, the value is stored in the following Registry key:

HKEY_CURRENT_USER\Software\JavaSoft\Prefs\com\ca\iam\conman

Chapter 3: Configuration

This section contains the following topics:

[Tuning the Database Used by the Provisioning Server](#) (see page 23)

[Provisioning Server Configuration](#) (see page 23)

[Connector Server Configuration](#) (see page 26)

[Configuration Data Location](#) (see page 29)

Tuning the Database Used by the Provisioning Server

We recommend that you tune your Provisioning Server. If you do not tune the Provisioning Server, the performance of the Provisioning Server may suffer when there are large numbers of objects stored in Dynamic Namespaces (JDBC and JNDI).

Use the eTrust Directory `dxtunedb` command after you add a large number of objects to the Provisioning Directory. With the introduction of the Dynamic Namespace option, execute the following Ingres commands following each `dxtunedb` command:

- `optimizedb -zr499 -zu240 <database name> -rsubsearch -acid`
- `optimizedb -zr499 -zu240 <database name> -rdit -aparent -ardnkey -aeid`
- `sql <database name> > set trace point RD010 \g`

Provisioning Server Configuration

Connector Xpress lets you manage connectors on a server by adding, editing, and removing Provisioning Servers.

Add and Configure a Provisioning Server

To manage connectors on a server, you need to add and configure a Provisioning Server.

To add and configure a Provisioning Server

1. In the Provisioning Servers tree, right-click the Provisioning Servers node, then click Add Server.

The Provisioning Server Details dialog appears.

2. Complete the fields in the dialog and then click OK.

Connector Xpress adds the Provisioning Server to the Provisioning Servers tree.

More Information:

[Provisioning Servers Tree](#) (see page 15)

[Provisioning Server Details Dialog](#) (see page 94)

Edit Provisioning Server Details

When the details of your Provisioning Server change, for example, when you upgrade your hardware, you may need to edit the details of your Provisioning Server.

To edit Provisioning Server details

1. In the Provisioning Servers tree, right-click the server whose details you want to edit, then click Edit Server.

The Provisioning Server Details dialog appears.

2. Complete the fields in the dialog, and then click OK.

Connector Xpress changes the details of the Provisioning Server.

More information:

[Provisioning Servers Tree](#) (see page 15)

[Provisioning Server Details Dialog](#) (see page 94)

Remove a Provisioning Server

When the details of your Provisioning Server change, for example, when you remove or replace your hardware, you may need to remove the details of your Provisioning Server from the Provisioning Servers tree.

To remove a Provisioning Server

1. In the Provisioning Servers tree, right-click the Provisioning Server that you want to remove, and then click Remove Server.
2. When prompted, confirm that you want to remove the Provisioning Server.
Connector Xpress removes the Provisioning Server.

More information:

[Provisioning Servers Tree](#) (see page 15)

Connect to a Provisioning Server

To connect to a Provisioning Server, you need to provide a password for the Provisioning Server.

To connect to a Provisioning Server

1. Expand the Provisioning Servers tree, and then double-click the Provisioning Server you want to connect to.
The Provisioning Server password dialog appears.
2. Type the password, and then click OK.

Note: If you enter an invalid password when attempting to connect to the Provisioning Server, then the connection will fail and Connector Xpress will display an error node on the Provisioning Servers tree. To re-connect with a different password, you must refresh the Provisioning Servers tree.

More information:

[Provisioning Server Password Required Dialog](#) (see page 92)

[Provisioning Servers Tree](#) (see page 15)

Connector Server Configuration

Connector server configuration lets you configure how your Provisioning Server will route individual endpoints to connector servers.

You can create multiple configurations and then associate the appropriate configuration with a given endpoint.

Note: You must [create the connector server configuration](#) (see page 26) for the endpoint before deploying the connector.

Create a Connector Server Configuration

To specify how your Provisioning Server will route endpoint types or individual endpoints to connector servers, you create a connector server configuration.

To create a connector server configuration

1. Expand the server in the Provisioning Servers tree where you deployed your connectors.
2. Right-click on the CS Configs node and then select New CS Config.

The Connector Server Configuration dialog appears.

3. Complete the fields on the dialog, and then click OK.

You have specified how your Provisioning Server will route endpoint types or individual endpoints to connector servers.

More information:

[Provisioning Servers Tree](#) (see page 15)

[Edit Connector Server Configuration Dialog](#) (see page 93)

Edit a Connector Server Configuration

To change how your Provisioning Server will route endpoint types to a connector server, you can edit the connector server configuration.

To edit a connector server configuration

1. Expand the server in the Provisioning Servers tree where you deployed your connectors.
2. Expand the CS Configs node.
3. Right-click the server configuration you want to edit, and then select Edit CS Config.

The Connector Server Configuration dialog appears.

4. Complete the fields on the dialog, and then click OK.

You have changed how your Provisioning Server will route endpoint types or individual endpoints to connector servers.

More information:

[Provisioning Servers Tree](#) (see page 15)

[Edit Connector Server Configuration Dialog](#) (see page 93)

Set a Managing Connector Server

You can select a server to manage a particular endpoint type or endpoint. You can also you associate two or more Connector Servers with an endpoint type. You may need to do this when you have alternate Provisioning Servers and you have a different CS configured for each Provisioning Server.

To set a managing connector server

1. In the Provisioning Servers tree, right-click the endpoint type or endpoint you want to manage, and then click Set Managing CS.

The Select Connector Servers dialog appears.

2. To specify the Connector Server or Connector Servers you want to manage, do *one* of the following:

- a. Select the Connector Server or Connector Servers from the list, and then click OK.

Connector Xpress adds the endpoint type or endpoint to the managed branches of the selected Connector Servers.

- b. Select the Use the default CS for Provisioning Server check box, and then click OK.

Connector Xpress uses the default connector server for the Provisioning Server to manage the endpoint type or endpoint.

More information:

[Provisioning Servers Tree](#) (see page 15)

[Edit Connector Server Configuration Dialog](#) (see page 93)

Remove a Connector Server Configuration

To remove a Connector Server configuration, you can delete it from the Provisioning Servers tree.

To remove a connector server configuration

1. Expand the server in the Provisioning Servers tree where you deployed your connectors.
2. Expand the CS Configs node.
3. Right-click the server configuration you want to edit, and then select Delete CS Config.
4. When prompted, confirm that you want to delete the Connector Server configuration.

More information:

[Provisioning Servers Tree](#) (see page 15)

Update a Java CS Password

To update the password for a Java CS

1. In the Provisioning Servers tree, expand the CS Configs node.
2. Right click the CS Config you want to edit, then click Edit CS Config
The Edit Connector Server Configuration dialog appears.
3. Complete the fields on the dialog, and then click OK.

More information:

[Provisioning Servers Tree](#) (see page 15)

[Edit Connector Server Configuration Dialog](#) (see page 93)

Configuration Data Location

The configuration data created using Connector Xpress is stored in various locations throughout the distributed system as described in the following sections.

More information:

[Java CS Routing Rules Location](#) (see page 29)

[Endpoint Type Metadata Location](#) (see page 30)

[Endpoint Type Metadata and Java CS Routing Rules](#) (see page 30)

[View Endpoint Type Metadata and Java CS Routing Rules](#) (see page 30)

[Java CS Routing Rules Hierarchy](#) (see page 30)

[Endpoint Type Distinguished Name](#) (see page 30)

[Connector Xpress Local Storage](#) (see page 31)

Java CS Routing Rules Location

The Java CS routing rules are located in the following location:

```
eTSAName=[CSNAME],eTSAContainerName=SAs,eTNamespaceName=CommonObjects,dc=[DOMAIN]  
,dc=eta
```

Endpoint Type Metadata Location

The endpoint type metadata is located in the following location:

`eTNamespaceName=[NAMESPACE] ,dc=[DOMAIN] ,dc=eta`

Endpoint Type Metadata and Java CS Routing Rules

The endpoint type metadata and Java CS routing rules are stored on the Provisioning Server.

View Endpoint Type Metadata and Java CS Routing Rules

To view the endpoint type metadata and Java CS Routing Rules, connect to the Provisioning Server using an LDAP browser such as JXplorer.

Java CS Routing Rules Hierarchy

Each CS has an entry under the following:

`eTSAName=CS
name,eTSAContainerName=SAs,eTNamespaceName=CommonObjects,dc=server,dc=eta`

For each endpoint type managed by the CS, there is an entry like the following:

`eTSABranchDN : eTNamespaceName=Oracle Server,dc=server`

Endpoint Type Distinguished Name

The endpoint type's distinguished name is as follows:

`eTNamespaceName= MyEndpointType,dc=Domain,dc=TopLevelProvisioningServer`

On each dynamic endpoint type object, the following attributes are used to store metadata:

eTMetaData

(Mandatory) Stores the XML document containing all wizard mappings, excluding stored procedures and operation bindings.

eTTopBindingsMetaData

(Optional) Stores the XML document containing stored procedures and operation bindings.

Connector Xpress Local Storage

Connector Xpress project files (.con files) are stored on the local file system.

Connector Xpress data source configurations are stored under Connector Xpress user preferences.

Note: You cannot export Connector Xpress data source configurations.

More Information:

[Projects](#) (see page 35)

[Set Preferences](#) (see page 21)

[Where User Preferences Are Stored](#) (see page 20)

Chapter 4: Creating Connectors

This section contains the following topics:

[How To Create and Deploy Connectors](#) (see page 34)

[Table Attribute Mappings](#) (see page 41)

[Mapping Against One Endpoint and Acquiring Against Another Endpoint](#) (see page 42)

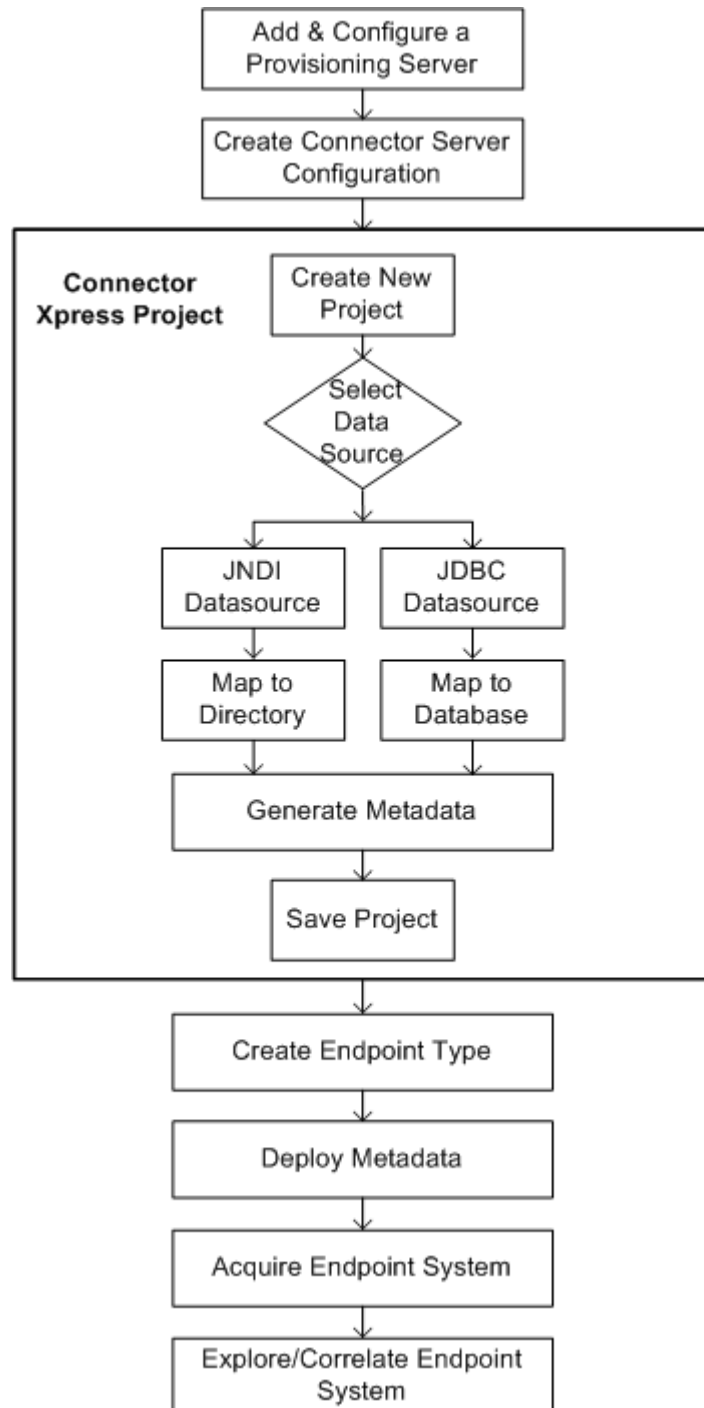
[Map Group Members \(JNDI\)](#) (see page 43)

[Type Mapping](#) (see page 43)

[Edit Metadata](#) (see page 43)

How To Create and Deploy Connectors

The following diagram shows the process you follow to create and deploy a connector using Connector Xpress.



To create and deploy a connector, you need to perform the following steps:

1. [Add and configure a Provisioning Server](#). (see page 24)
2. [Create a connector server configuration](#). (see page 26)
3. [Create a Connector Xpress project](#): (see page 36)
 - a. [Setup data sources](#) (see page 13).
 - b. [Complete the Connector Xpress Wizard](#) (see page 38) to create the appropriate endpoint type metadata.
4. Deploy the new endpoint type metadata by:
 - a. [Creating a new endpoint type](#) (see page 82) type on the Provisioning Server.
 - b. [Selecting a managing connector server](#) (see page 28).
 - c. [Deploying the connector](#) (see page 40).
5. [Acquire, Explore and Correlate the Endpoint](#) (see page 81).

Projects

Connector Xpress lets you create projects that specify how connectors are configured and deployed to the Provisioning Server.

A project represents a connector in Connector Xpress. A project includes:

- The schema retrieved from a data source.
- The configuration of metadata that describes how Connector Xpress should map the schema to the endpoint.

You can save, edit, and re-open projects for later use.

Open an Existing Project

To open an existing project, click File, Open Project, and choose the file you want to open from the Open Project dialog.

Connector Xpress opens the project and displays the metadata in the Edit pane of the Connector Xpress window.

Reopen a Recent Project

To reopen a recent project, click File, Reopen Recent, and select the project you want to open.

Connector Xpress opens the project and displays the metadata in the Edit pane of the Connector Xpress window.

Create a Project

To create a connector, you can create projects that specify how connectors are configured and deployed to the Provisioning Server.

To create a project

1. Click File, New Project.

The Select Data Source for new Project dialog appears.

2. If the data source you want to use is displayed in the list, then go to step 4.

3. If the data source you want to use is not displayed in the list, do the following:

- a. Click Add.

The Source Types dialog appears.

- b. Select an available data source type from the list, and then click OK.

The Edit Source dialog appears, specific to the type of data source you are adding.

- c. Complete the fields on the dialog to configure the data source type you selected, and then click OK.

The Select Data Source for new Project dialog appears and the data source you added appears in the list.

4. Select the data source you want to use from the list, then click OK.

The Enter Password for Data Source Dialog appears.

5. Enter the password for the selected data source and then click OK.

The Connector Xpress Wizard appears.

6. Next, you need to [complete the Connector Xpress Wizard](#) (see page 38) to create the appropriate endpoint data.

7. [View the Metadata Tree](#) (see page 39).

8. Next, you need to [save the connector](#). (see page 39)

Note: If you modify a project without saving, start the wizard, and then cancel the wizard, you will lose all modifications. If you cancel a wizard, Connector Xpress closes the project and does not save any modifications.

9. Next, you need to [deploy the connector](#) (see page 40).

More Information:

[Select Data Source for new project Dialog](#) (see page 86)

[Source Types Dialog](#) (see page 87)

[Edit Source Dialog](#) (see page 87)

[Projects](#) (see page 35)

Edit a Project

If you need to change the details of a project, you can edit the metadata in the Connector Xpress project using the Connector Xpress Wizard.

To edit a Connector Xpress project

1. [Open an existing project](#) (see page 35).
Connector Xpress opens the project, and displays the metadata in the rightmost pane of the Connector Xpress window.
2. In the upper left pane of the Connector Xpress window, click Wizard.
The Enter Password for Data Source dialog appears.
3. Complete the fields on the dialog and then click OK.
The Connector Xpress Wizard appears.
4. [Complete the Connector Xpress Wizard](#) (see page 38) to edit the appropriate endpoint type data.
5. [Save the connector.](#) (see page 39)
6. [Deploy the connector](#) (see page 40) if required.

More Information:

[Projects](#) (see page 35)

[Enter Password for Data Source Dialog](#) (see page 90)

How You Use the Connector Xpress Wizard

You can create a dynamic connector using the Connector Xpress Wizard to map database tables to provisioning accounts and groups.

Consider the following guidelines when mapping database tables:

- (JDBC) Because space padding can require additional exploration, try using VARCHAR rather than CHAR types where possible for columns mapped to naming attributes.
- (JDBC and JNDI) If using a default value for Boolean column, ensure the default value is a valid Boolean.

To map database tables to provisioning accounts and groups, you use the Connector Xpress Wizard to do the following:

1. Specify the name, details, and version of the new endpoint.
2. (Optional) Register pre-existing database stored procedures against selected operations on accounts and groups.
3. Select the schema and account/group table or account/group objectClass, and map selected columns from the specified Account/Group Table or Account/Group Object Class to Provisioning Manager LDAP attributes.
4. Review and correct, if necessary, the columns that Connector Xpress has created.
5. Select the schema and group table or group object class to map the table columns to managed attributes.
6. Review and correct, if necessary, the group mapping types that you have created.
7. Select a separate lookup table for associations between account and group objects and map the relevant attributes.
8. Specify the object classes that you want to use as containers in your connector, and their naming attributes.

Note: If you modify a project without saving, start the wizard, and then cancel the wizard, you will lose all modifications. If you cancel a wizard, Connector Xpress closes the project and does not save any modifications.

More Information:

[General Screen](#) (see page 97)

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[Map Account or Group Table Screen](#) (see page 115)

[Map Account or Map Group Confirm Screen](#) (see page 118)

[Map Group Table Screen](#) (see page 121)

[Map Membership Table Screen \(JDBC Only\)](#) (see page 125)

[Map Containers Screen \(JNDI Only\)](#) (see page 126)

View the Metadata Tree

After [completing the Connector Xpress Wizard](#) (see page 38), you can view the newly created metadata from the Metadata Tree in the Edit pane of the Connector Xpress window.

To view the newly created metadata from the Metadata Tree, locate the Edit Pane of the Connector Xpress windows and expand the properties.

The attributes defined on the account display.

Save the Connector

Once you have created a new dynamic connector, you can save the connector project for future deployment.

To save the connector

1. Click File, Save Project or Save Project As.

The File Explorer appears.

2. Specify the folder to save the connector project in, the file name and the file type for the saved connector, and click Save.

Connector Xpress saves the connector.

Note: We recommend that you use the default file extension (.con)

Note: We recommend that you keep a copy of the connector project. This is because you cannot view the stored procedure bindings after you deploy a connector, unless you open the Connector Xpress project file where your bindings are stored.

More Information:

[Projects](#) (see page 35)

Connector Deployment

You can use Connector Xpress to configure metadata settings and to deploy the connector. Connector Xpress lets you save metadata settings to a local file and then deploy a connector from that file. This can be useful in cases where the same settings apply on multiple Provisioning Servers, for example, when you move connectors from a test environment to a production environment after testing is complete.

Deploy the Connector

Once you have created a connector, you can deploy the metadata for that connector to a Provisioning Server.

Note: This procedure assumes that you have [created the connector server configuration](#) (see page 26).

To deploy the connector

1. [Open the existing project](#) (see page 35) that contains the metadata you want to deploy.

Connector Xpress opens the project and displays the metadata in the Edit pane of the Connector Xpress window.

2. In the Provisioning Servers tree, expand the Provisioning Servers node and then choose the server where you want to deploy the connector.

The Provisioning Server Password Required dialog appears.

3. Complete the fields on the dialog to specify the password for the server, and click then OK.

4. Expand the server, and then right-click on Endpoint Types

The Create New Endpoint Types dialog appears.

5. Complete the fields on the dialog to define the name of your new endpoint type and then click OK.

6. Right-click on your new endpoint and select Acquire Endpoint.

The Create New Endpoint dialog appears.

7. Complete the fields on the dialog to specify the name and password for your new endpoint, and then click OK.

8. (Optional) Under the endpoints node, right-click on your new endpoint and choose Explore/correlate endpoint.

The Explore/Correlate Endpoint dialog appears.

9. Complete the fields on the dialog to specify how Connector Xpress explores and correlates the endpoint, and then click OK.

Connector Xpress deploys the connector to the Provisioning Server.

More information:

[Provisioning Servers Tree](#) (see page 15)

[Create New Endpoint Dialog](#) (see page 91)

[Explore/Correlate Endpoint Dialog](#) (see page 90)

[Projects](#) (see page 35)

Connector Undeployment

You can use Connector Xpress to undeploy connectors. This deletes the connector from the Provisioning Server's persistent storage as well as from the Java Connector Server. You can also delete connectors (or instances of their parent endpoint types) using the Provisioning Manager user interface. In both cases, you do not need to restart the Java CS, and Connector Xpress does not delete data on the endpoint.

Table Attribute Mappings

When you define your table and attribute mappings, you can make the following choices in the fields on the wizard screens control. The choices you make define which pre-defined attribute is used and what the behavior will be.

- Data Type
 - String case sensitive
 - String case insensitive
 - Boolean
 - Integer
 - Long
 - Double
 - Date
 - DateTime
 - Time
 - Float
- Sync Type
 - Boolean Sync
 - Integer Sync
 - String Sync
- Multivalued - either enabled or disabled

For example, the choices you make set the limits as described in the following table on how many pre-defined attributes there are for that type and behavior:

Type and behaviour	Limit
single-valued non-capability integer	10
single-valued capability integer	10

Type and behaviour	Limit
multi-valued non-capability integer	10
multi-valued capability integer	10
single-valued sensitive non-capability string	30
single-valued insensitive non-capability string	30
single-valued sensitive capability string	10
single-valued insensitive capability string	10
multi-valued sensitive non-capability string	10
multi-valued insensitive non-capability string	10
multi-valued sensitive capability string	10
multi-valued insensitive capability string	10
single-valued non-capability Boolean	10
single-valued capability Boolean	10
single-valued binary	10

Mapping Against One Endpoint and Acquiring Against Another Endpoint

If you perform mapping against one endpoint and acquire a directory referring to another, make sure they have exactly the same tables and/or stored procedures (JDBC) or LDAP schema (JNDI).

Map Group Members (JNDI)

In the Connector Xpress wizard, mapping to the group class member attribute achieves the same result as the Map Membership Table screen does for JDBC. This attribute is hardwired to contain values of type DN (Distinguished Name) which are expressed relative to the root of the endpoint directory and enumerate the accounts belonging to each group.

When member is mapped for the group class then the reverse DN attribute memberOf is automatically mapped for the account class. This attribute is virtual, meaning that its value is expensive to retrieve, as it has to be computed from group.member rather than being directly looked up. We therefore recommend that you should request it with caution.

Some JNDI vendors, notably Novell eDirectory, actually expose the account.memberOf attribute in their schema. However to guarantee consistent behaviour across all vendors you are prohibited from mapping it explicitly. Instead, the Java CS implements it as a virtual attribute.

More Information:

[Map Membership Table Screen \(JDBC Only\)](#) (see page 125)

Type Mapping

In Connector Xpress, an *edittype=int* maps to a plain LDAP string attribute, unless you explicitly request integer sync.

Edit Metadata

You can manually edit the metadata for a project.

Important! Use the metadata editor at your own risk. Removal or alteration of important objects in the metadata may render it inoperable.

To edit metadata

1. [Open an existing project](#) (see page 35).
2. In the Provisioning Servers tree, right-click the endpoint type you want to edit the metadata for, then click Edit Metadata.

The Edit Metadata for Endpoint Type dialog appears and displays the metadata in an XML Tree.

3. Edit the metadata as required, then click Save.

Connector Xpress redeploys the metadata.

More information:

[Projects](#) (see page 35)

[Provisioning Servers Tree](#) (see page 15)

[Edit Metadata for Endpoint Type Dialog](#) (see page 95)

[Operations Bindings Editor](#) (see page 18)

[Metadata Editor](#) (see page 17)

Metadata and Operations Bindings Editors Considerations

Important! You can use the Metadata and Operations Bindings editors to verify the parameters you entered using the wizard, however we recommend that you use the Metadata and Operation Bindings editors at your own risk. Removal or alteration of important objects in the metadata may render it inoperable. This also applies to manually configuring script operation bindings, which Connector Xpress does not support in this release.

More Information:

[Metadata Editor](#) (see page 17)

[Operations Bindings Editor](#) (see page 18)

[Stored Procedure Bindings](#) (see page 72)

Re-deploy Metadata

When a connector project is open and metadata is available, you can push the metadata to the endpoint type on the Provisioning Server. You can only deploy metadata to dynamic endpoint types.

Note: If the endpoint type already has metadata, the new metadata cannot alter fundamental aspects of the data model such as adding attribute names or types.

To deploy metadata

1. [Open an existing project](#) (see page 35).
2. In the Provisioning Servers tree, right-click the endpoint type you want to deploy data to, and then click Deploy Metadata.

The Deploy Metadata dialog appears.

3. Confirm that you want to deploy the metadata.

Connector Xpress deploys the metadata to the selected endpoint type and overwrites any existing metadata.

More information:

[Projects](#) (see page 35)

[Provisioning Servers Tree](#) (see page 15)

DYN Schema Extensions

The DYN Schema has been extended for the following attributes:

- Container objects (eTDYNContainerXXX) are extended to 10.
- Generic objects (eTDYNObjectXXX) are extended to 35.
- All capability attributes (for example, eTDYN-str-multi-ic-, eTDYN-str-ic-, eTDYN-str-c-, eTDYN-bool-c-, eTDYN-int-multi-c-, eTDYN-int-c-) are extended to 99, with the exception of the multi-valued case-sensitive attributes (eTDYN-str-multi-c-), which are extended to 500.
- The non-capability multi-valued case-sensitive or case-insensitive attributes (for example, eTDYN-str-multi-, eTDYN-str-multi-i-) are extended to 500.
- The generic cached (that is, data location equals BOTH) multi-valued case-sensitive attributes (eTDYN-str-multi-ca-) are extended to 99. These attributes are included in every DYN object.
- 99 connection-specific cached multi-valued case-sensitive attributes (eTDYN-str-multi-ca-sec-) are added for DYN Directory only. These attributes are added in the "encryptwith" line of the DYN password attribute.

Support for Mandatory Attributes on a JDBC Endpoint

To support attributes that the JDBC endpoint defines as mandatory (that is, NOT NULL), but do not need to be mandatory from the provisioning point of view, the JDBC connector allows you to use an empty value for the provisioning attribute.

When you create an account and are mapping a mandatory attribute on the endpoint to a provisioning attribute that is non- mandatory, for example, a description field, and you do not enter a value for the description field, the JDBC connector maps the empty provisioning attribute to the endpoint attribute by populating the NOT NULL column on the endpoint database with spaces.

To support mapping of non-mandatory fields in Provisioning Manager, the Java Connector Server contains a `NullValueClassConverter` that converts empty values in non-mandatory provisioning attributes to spaces in mandatory attributes on the endpoint.

For example, empty values can occur on a legacy database system has a description field that is NOT NULL on the table being mapped to a user account. However, the description field is not set to mandatory in the Provisioning Manager, which means you do not have to enter a description for a user to create an account. The `NullValueClassConverter` is used to store the empty value.

The property `nullValue` in the `pluginConfig` class in the `NullValueClassConverter` is set to a space by default. You can change `nullvalue` to other values, but changing `nullvalue` [requires additional configuration of the endpoint database](#) (see page 50).

For JDBC endpoints, the Java Connector Server loads the converter automatically. For other types of endpoints, for example, JNDI, [manually configure the Java Connector Server to load the converter](#) (see page 49).

How to Enable Support for Mapping of Non-mandatory Fields in Provisioning Manager

To enable support for mapping of non-mandatory fields in Provisioning Manager to not-null columns in a JDBC database using the JDBC connector, add the metadata attribute *useSpecialNullValue* to the Connector Xpress Dyn mapping to each attribute that you want the *NullValueClassConverter* to convert. The Java Connector Server checks for the presence of this metadata attribute before enabling the converter.

Note: You can also use this procedure to enable support for other types of endpoints.

To enable support, do the following:

1. Use Connector Xpress or another editor to edit the metadata for your endpoint type.
2. Add the new Boolean metadata attribute *useSpecialNullValue* to the attribute that you want the *NullValueClassConverter* to convert, and set the Boolean value to true.

Note: Add the metadata attribute *useSpecialNullValue* to the Connector Xpress Dyn mapping to each attribute that you want to convert. The Java Connector Server checks for the presence of this metadata attribute before enabling the converter.

3. Set the *isRequired* metadata attribute to false.
4. Repeat this procedure for all attributes that you want to convert.

Example: Enable support for mapping of non-mandatory fields in Provisioning Manager

This example uses Connector Xpress to edit the metadata. The Description attribute of your account is mapped to eTDYN-str-01.

To enable support for mapping of non-mandatory fields

1. In Connector Xpress, right-click the endpoint type you want to edit the metadata for, then click Edit Metadata.

The Edit Metadata for Endpoint Type dialog appears.

2. Expand the eTDYN-str-01 node, then select the metadata subnode.
3. Click the Add button.

Connector Xpress adds a new metadata attribute node to the tree.

4. Name the new metadata node *useSpecialNullValue*.
5. Select Boolean from the type drop-down list.
6. Select the check-box next to the drop-down list.

The value of the *useSpecialNullValue* attribute is set to *true*.

7. Select the *isRequired* metadata attribute on eTDYN-str-01.
8. Clear the check-box next to the drop-down list.

The value of the *isRequired* attribute is set to *false*.

How to Enable Support for Mandatory Attributes on Endpoints other than JDBC Endpoints

To support endpoints other than JDBC endpoints that define attributes as mandatory (that is, NOT NULL), but do not need to be mandatory from the provisioning point of view, do the following:

1. [Configure the Java Connector Server to load the NullValueClassConverter.](#) (see page 49)
2. [Enable support for mapping of non-mandatory fields in Provisioning Manager](#) (see page 47).

Configure the Java Connector Server to load the NullValueClassConverter

The NullValueClassConverter plugin is shipped with the JDBC connector. To use the NullValueClassConverter for endpoints other than JDBC endpoints, configure the Java Connector Server to load the converter.

Note: The converter is loaded automatically for JDBC endpoints.

Note: The property *metadataPropNames* has a value of *useSpecialNullValue*. Add the *metadataPropNames* metadata attribute to the Connector Xpress Dyn mapping to each attribute that plugin handles. The Java Connector Server checks for the presence of this metadata attribute before enabling the plugin.

To enable the plugin, configure the plugin in an override connector.xml for your endpoint.

To configure the Java Connector Server to load the NullValueClassConverter

1. Rename the file SAMPLE.connector.xml in C:\Program Files\CA\Identity Manager\Connector Server\conf\override\jdbc to connector.xml.
2. Add the following to your class plugin config of your connector.xml file:

```
<property name="classPluginConfigs">
  <list>
    <bean class="com.ca.jcs.cfg.MetaPluginConfig">
      <property name="pluginClass">

        <value>com.ca.jcs.converter.meta.NullValueClassConverter</value>

      </property>
      <property name="pluginConfig">

<bean
class="com.ca.jcs.converter.meta.NullValueClassConverter$NullValueConverterCo
nfig">

          <property name="nullValue">
            <value> </value>
          </property>x
        </bean>
      </property>x
      <property name="metadataPropNames">
        <list>
          <value>useSpecialNullValue</value>
```

```
        </list>
      </property>
    </bean>
  </list>
</property>
```

Note: The property `nullValue` in the `pluginConfig` class in the `NullValueClassConverter` is set to a space by default. To change the null value the plugin uses, change the `nullValue` in the config bean.

Change the Default Value Used to Store an Empty Value

The property `nullValue` in the `pluginConfig` class in the `NullValueClassConverter` is set to a space by default. To change the default value used to store an empty value, change the `nullValue` in the config bean in the class `plugin config` of your `connector.xml` file.

Metadata Descriptions

The following topics describe the metadata used in all layers of the architecture including Connector Xpress, JIAM, and the Java CS.

More information:

[Metadata for Endpoint Types](#) (see page 50)

[Metadata for Connectors](#) (see page 52)

[Metadata for Classes](#) (see page 53)

[Metadata for Classes or Attributes](#) (see page 67)

Metadata for Endpoint Types

abbreviation

Specifies the short abbreviated name of the endpoint type.

Type: String

accountclass

Specifies the LDAP object class of the account objects.

Type: String

endPointClass

Specifies the LDAP objectClass of the end-point objects.

Type: String

eta.legacyPolicyContainer

If true, indicates that the policy container is not under the endpoint type, but under the CommonObjects endpoint type.

Specifies the location of the policy container. If true, indicates that the policy container is not under the endpoint type, but under the CommonObjects endpoint type.

Type: Boolean

Default: False

groupclass

Specifies the LDAP objectClass of the group objects.

Type: String

implementationBundle

Should match exactly one of the getName() values of one of the com.ca.jcs.ImplBundle beans created from the conf/connector.xml files read from all the jcs-connector-*.jar files (each of which is a connector implementation) on the classpath. This value is of critical importance when the same LDAP schema is used for multiple connector implementations (for example, eTDYNNamespace which is used for both the JDBC and JNDI connector implementations).

Java Constant: MD_IMPL_BUNDLE

Type: String

Note: For more information, see com.ca.jcs.ImplBundle in the *Java CS Javadoc*.

isHierarchical

Specifies whether the connector supports arbitrary levels of nesting objects within containers, instead of being flat and having only fixed Accounts and Groups containers.

Java Constant: MD_IS_HIERARCHICAL

Type: Boolean

Default: False

policyContainerClass

Specifies the LDAP objectClass of the policy container.

Type: String

policyClass

Specifies the LDAP objectClass of the policy objects.

Type: String

quoteString

Specifies the quote string (for connector) to be replaced around objectClasses/attribute IDs.

Java Constant: MD_QUOTE_STRING

Type: String

Metadata for Connectors

connector

Connector (that is, Provisioning Directory) has connectorMapTo= this value, so it can be identified.

Java Constant: MAP_TO_CONNECTOR_VALUE

Type: String

isConnection

Distinguishes connection-related attributes on the connector, which when changed require the connector to be deactivated and then reactivated by the Java CS framework.

Java Constant: MD_IS_CONNECTION_ATTR

Type: Boolean

Default: False

isLogging

Distinguishes logging-related attributes on the Connector, to assist UI clients like the Provisioning Manager plug-in that need to be able to distinguish them.

Java Constant: MD_IS_LOGGING_ATTR

Type: Boolean

Default: False

Metadata for Classes

connectorMapToLax

Specifies that MD_CONN_MAP_TO mappings are *lax*. This Boolean can appear on any objectclass which has MD_CONN_MAP_TO specified.

This means any attributes on classes that are not explicitly mapped will be treated as strings and keep their original names.

When connectorMapToLax is false, unmapped attributes passed in by clients for mapped object classes are discarded. When connectorMapToLax is true, unmapped attributes are passed through with their original names. Setting connectorMapToLax to true saves having to exhaustively map every property, which is particularly useful for the connector object class.

Java Constant: MD_CONN_MAP_TO_LAX

Type: Boolean

Default: False

isCreatable

Specifies whether the object class may be created. Otherwise, the object class is not fully managed by the connector, but simply read from it.

Type: Boolean

Default: **True****isContainer**

Distinguishes containers from other object classes.

Java Constant: MD_IS_CONTAINER

Type: Boolean

Default: False

isDeletable

Specifies whether the object class may be deleted.

Type: Boolean

Default: True

isMovable

Specifies whether the object class may be moved.

Type: Boolean

Default: True

isRenameable

Specifies whether the object class may be renamed.

Type: Boolean

Default: True

isSingleNamingValue

Handle cases like multiple *cn=* values for JNDI-based connectors. This is acceptable as far as the Java CS is concerned, but may cause problems for the Provisioning Server during explore and correlations (setting this metadata property to true causes all values except the one mentioned in the object's RDN to be deleted).

Java Constant: MD_IS_SINGLE_NAMING_VALUE

Type: Boolean

Default: False

rdnAttribute

Specifies the name of the attribute that forms the RDN in LDAP.

Type: String

Metadata for Attributes

assocRefObjectClass

Defines a relationship from an obj type to a referenced obj type, where direct association through an attribute and indirect association through a table external to both object types are supported.

For direct association, an attribute in the *referencing* objectclass, (for example, Account.groups), defines its association with another *referenced* objectclass, (for example, Group), by specifying the multi-valued attribute in the referenced objectclass that is used to store the names of contained items (for example, Group.accounts). A concrete connector then needs to perform the logical reverse lookup required to populate the virtual associative attribute. In this example, the virtual Account.groups attribute is populated by looking for Groups that contain an account's name in its Group.accounts attribute. Two metadata settings are required:

- MD_ASSOC_REF_OBJECT_CLASS (=Group in the example above)
- MD_ASSOC_REVERSE_ATTR (=accounts in the example above)

For the indirect case, you need to specify the following:

- The referenced objectclass must be specified through MD_ASSOC_REF_OBJECT_CLASS), as for the direct case.
- An association table (specified by MD_ASSOC_TABLE) with two columns:
 - The column named in MD_ASSOC_TABLE_OBJ_COLUMN references the key for the containing objectclass
 - The column named by MD_ASSOC_TABLE_REF_COLUMN) references the key for the referenced objectclass.
- An optional MD_ASSOC_OBJ_KEY_ATTR can be used to flag that an alternative key for the containing objectclass is to be stored in the MD_ASSOC_TABLE_OBJ_COLUMN column (if it is not specified then the containing object's naming attribute is assumed).
- An optional MD_ASSOC_REF_KEY_ATTR can be used to flag that an alternative key for the referenced objectclass is to be stored in the MD_ASSOC_TABLE_REF_COLUMN column (if it is not specified then the containing object's naming attribute is assumed).

Except for the last two, all of these metadata settings are mandatory to specify an indirect association.

Java Constant: MD_ASSOC_REF_OBJECT_CLASS

Type: String

beanPropertyName

Specifies the name of the Java Bean property corresponding to this attribute.

Type: String

boolValues

Specifies, in order, the alternate symbolic values that you can use to represent false (0) and true (1) for target <boolVal> properties stored on the endpoint.

For example:

```
<metadata name="boolValues">
  <value>
    <sequenceValue>
      <!-- "false" value -->
      <val><strValue>XXX</strValue></val>
      <!-- "true" value -->
      <val><strValue>YYY</strValue></val>
    </sequenceValue>
  </value>
</metadata>
```

Java Constant: MD_BOOL_VALUES

Type: Sequence

connectorConverter

Allows explicit converter plug-in configuration to be chosen for a property by reference to its name, where contextual information is not sufficient to allow the choice to be made automatically (for example, if there are multiple flattening converters for a multi-valued field which needs to be flattened).

Java Constant: MD_CONN_CONV_NAME

Type: String

connectorMapToStrict

Causes the framework to augment objectclass filters to include an explicit check for the existence of a particular connector-speak naming attribute. This is used in cases where the naming attribute is potentially ambiguous, but you have chosen to map only one choice.

For instance, in the JNDI case, map account to *person* and map account naming attribute to *uid*. If this metadata setting is *true* on the naming attribute, then the LDAP filter:

```
<code>(objectclass=eTDYNAccount)</code>
```

will become

```
<code>(&(objectclass=person)(uid=*))</code>
```

Without this metadata setting, the filter will be

```
<code>(objectclass=person)</code>
```

, and the framework will deduce the eTDYNAccountNames of pre-existing native accounts that only define

```
<code>cn=</code>
```

 and have no value for

```
<code>uid=</code>
```

.

Note: Naming property only.

Java Constant: MD_CONN_MAP_TO_STRICT

Type: Boolean

Default: False

converterRegex

Specifies regular expression (of style specified by MD_CONVERT_REGEX_STYLE) to be used to convert values stored in the target property. Where the same regular expression is used in a number of places, a better approach would be to define a

```
<code>FLEXI_STR:myType</code>
```

 type and tie this type to a `com.ca.jcs.converter.attr.RegexAttributeConverter` through your connector's `conf/connector.xml` file.

Java Constant: MD_CONVERT_REGEX

Type: String

Note: For more information, see `com.ca.jcs.converter.attr.RegexAttributeConverter` in the *Java CS Javadoc*.

converterRegexStyle

Specifies style of regular expression used to validate the target property.

Java Constant: MD_CONVERT_REGEX_STYLE

Type: String

Default: perl5

Note: For more information, see MD_CONVERT_REGEX and `com.ca.jcs.cfg.RegexPluginConfig#PERL5` in the *Java CS Javadoc* for possible styles.

connectorValidator

Allows explicit validator plug-in configuration to be chosen for a property by reference to its name, where contextual information is not sufficient to allow the choice to be made automatically (for example, if there are multiple flattening converters for a multi-valued field which needs to be flattened).

Java: Constant: MD_CONN_VALID_NAME

Type: String

defaultPolicyValue

Contains the value to use for this attribute in the default policy. Only present on account attributes.

Type: String

(Mandatory) DNLDAPObjectClass

Specifies name of objectclass referenced by the distinguished name value(s) for the target property.

Note: Any property with *FLEXI_STR:DN* setValue

Java Constant: MD_DN_LDAP_OBJECTCLASS

Type: Boolean

DNNameOnly

Specifies that the connector, (which must have *isHierarchical* = *false*), wants the DNs converted to a single name for storage.

For example, `eTDYNGroupName=grp1` will be converted to `grp1`, before passing the value to the connector. Also, the reverse transformation from name to DN occurs when the connector passes back the query results.

Note: Applies to Any property with *FLEXI_STR:DN* setValue

Java Constant: MD_DN_NAME_ONLY

Type: Boolean

Default: False

DNTestExists

Specifies that each contained DN should be tested for existence by doing a `doLookup()` on each DN requesting an empty array of attributes.

Note: Applies to any property with *FLEXI_STR:DN* setValue.

Java Constant: MD_DN_TEST_EXISTS

Type: Boolean

Default: False

Note: For more information, see `com.ca.jcs.LdapUtil#TEST_EXISTENCE_ATTRS` in the *Java CS Javadoc*.

encryptAlgorithm

Specifies the algorithm used for encryption. For more information, see. `MD_IS_OBFUSCATED`.

Note: Any encrypted property.

Java Constant: MD_ENCRYPT_ALG

Type: String

flattenStyle

Specifies that the values for the multi-valued target property should be flattened into a single string literal using the named *style* (for example, XML, CSV, and SQL).

Note: Multi-valued properties only.

Java Constant:MD_FLATTEN_STYLE

Type: String

Note: For more information, refer to classes deriving from `com.ca.jcs.converter.meta.MultiValuePropertyFlattener` in the *Java CS Javadoc*.

flattenStyle

Type: String

Specifies the way Java CS flattens a multi-valued attribute in the Provisioning Server to a single valued endpoint attribute. This metadata item can be set to XML, CSV, or SQL. No flattening will be done if none is provided.

forceModificationMode

Specifies that there is only one mode of modification item for the target property, regardless of the modification operation originally passed in by the client.

This can be set to REPLACE or DELTA.

This is useful when the API used by the connector to interact with the managed system mandates a particular style of access for the property.

DELTA

Specifies that a REPLACE_ATTRIBUTE modification item is replaced by one or both equivalent ADD_ATTRIBUTE and REMOVE_ATTRIBUTE deltas.

A REPLACE_ATTRIBUTE with no values (that is, completely remove the attribute) is replaced by a REPLACE_ATTRIBUTE with all current values mentioned explicitly.

REPLACE

Specifies that an ADD_ATTRIBUTE or REMOVE_ATTRIBUTE modification item is replaced by equivalent item with operation REPLACE_ATTRIBUTE.

Java Constant: MD_FORCE_MOD_MODE

Type: String

isConnectorFilterable

Indicates that the connector is unable to filter this attribute. Filters supplied to the connector will have references to this attribute removed. The results returned for searches filtering on this attribute will have the complete filter reevaluated against each of them by the Java CS framework to determine if they really should be returned, at an additional runtime cost.

Java Constant: MD_IS_CONNECTOR_FILTERABLE

Type: Boolean

Default: False

isEncrypted

Triggers the Provisioning Manager plug-in to display the attribute in a "password" type control so that characters are echoed as *. The user must retype their choice in a second confirmation test field.

Java Constant: MD_IS_ENCRYPTED

Type: Boolean

Default: False

isExpensive

Designates that the target property to which it is attached is expensive to retrieve and is therefore only returned in SEARCH requests when explicitly listed amongst the attributes to be returned (and therefore not by the default null list). For example, a photograph stored on a user / account object.

Java Constant: MD_IS_EXPENSIVE

Type: Boolean

Default: False

isInternal

Specifies whether this attribute is available through public APIs such as JIAM. If false, attribute will be available.

Type: Boolean

Default: False

isJavaObject

Triggers `com.ca.jcs.converter.meta.JavaObjectPropertyConverter` converter from LDAP string values to the Java Object matching `DataModel` value defined in the metadata.

For example, the string `true` stored in a `boolValue` field will be replaced with `<code>Boolean.TRUE</code>`.

Java Constant: MD_IS_JAVA_OBJECT

Type: Boolean

Default: False

Note: For more information, see `DataModelValue#getValue(int, String)` and `com.ca.jcs.converter.meta.JavaObjectPropertyConverter` in the *Java CS Javadoc*.

isMultiValued

Specifies whether the attribute can hold multiple values.

Java Constant: MD_IS_MULTI_VALUED

Type: Boolean

Default: False

isNaming

Specifies whether the attribute is the LDAP naming attribute.

Java Constant: MD_IS_NAMING

Type: Boolean

Default: False

isObfuscated

Triggers encryption converter using the algorithm specified using MD_ENCRYPT_ALG. This means that values are persisted in obfuscated form on the managed system.

Note: Obfuscating fields means that only the connector can read them and can no longer be accessed by direct access to the managed system.

Java Constant: MD_IS_OBFUSCATED

Type: Boolean

Default: False

isReadOnly

Specifies whether the attribute is read-only.

Java Constant: MD_IS_READ_ONLY

Type: Boolean

Default: False

isRequired

Designates that the target property to which it is attached is required, meaning that a value must be provided for it when objects are created and that it must have a least one value at all times.

Note: Using `<value default=true>...</value>` implies that `isRequired=true`, as the Java CS will insert the provided default value unless the target property has a value explicitly provided for it in an ADD request.

Java Constant: MD_IS_REQUIRED

Type: Boolean

Default: False

isReadOnly

Specifies whether the attribute is read-only.

Java Constant: MD_IS_READ_ONLY

Type: Boolean

Default: False

isStoreNumeric

Specifies that target properties that have types that can either be stored as strings or numeric types, (for example, <boolVal> or <enumVal>), are stored on the endpoint in numeric form.

See also MD_IS_STORE_SYMBOLIC, which specifies that it is stored in string form.

Java Constant: MD_IS_STORE_NUMERIC

Type: Boolean

Default: False

isStoreSymbolic

Specifies that target properties that have types that can either be stored as strings or numeric types, (for example, <boolVal> or <enumVal>), are stored on the endpoint in string (symbolic form).

See also MD_IS_STORE_Numeric, which specifies that it is stored in numeric form.

Java Constant: MD_IS_STORE_SYMBOLIC

Type: Boolean

Default: True

isStringArray

Triggers com.ca.jcs.converter.meta.StringArrayPropertyConverter converter from multi-valued LDAP attribute to String[].

Note: Multi-valued properties only

Java Constant: MD_IS_STRING_ARRAY

Type: Boolean

Note: For more information, see com.ca.jcs.converter.meta.StringArrayPropertyConverter in the *Java CS Javadoc*.

isTrimRequired

Specifies that that target property needs a whitespace trim operation.

Java Constant: MD_IS_TRIM_REQUIRED

Type: Boolean

Default: False

isVirtual

Flags the target property as *computed* rather than *persisted*.

Note: This is currently used for associations but may see wider use in the future.

Java Constant: MD_IS_VIRTUAL

Type: Boolean

Default: False

isWriteOnly

Designates that the target property to which it is attached is write-only and therefore can be modified but not read. You can use this for security-sensitive fields like passwords.

Java Constant: MD_IS_WRITE_ONLY

Type: Boolean

Default: False

minLength

Specifies minimum length for value(s) of target property.

Java Constant: MD_MIN_LENGTH

Type: Integer

maxLength

Specifies maximum length for value(s) of target property.

Java Constant: MD_MAX_LENGTH

Type: Integer

minValue

Specifies minimum bound on value(s) of target property.

Java Constant: MD_MIN_VALUE

Type: Number (Integer, long, float or double)

maxValue

Specifies maximum bound on value(s) of target property.

Java Constant: MD_MAX_VALUE

Type: Number (integer, long, float or double)

regexDescription

Describes the regular expression used to validate, or validate and convert the target property, which appears in validation failure messages.

Java Constant: MD_REGEX_DESCR

Type: String

validatingRegex

Specifies regular expression (of style specified by MD_VALID_REGEX_STYLE) to be used to validate values stored in the target property. Where the same regular expression is used in a number of places, a better approach would be to define a `<code>FLEXI_STR:myType</code>` type and tie this type to a `com.ca.jcs.validator.attr.RegexAttributeValidator` using your connector's `conf/connector.xml` file.

Java Constant: MD_VALID_REGEX

Type: String

validatingRegexStyle

Specifies style of regular expression used to validate the target property. For more information, see MD_VALID_REGEX, and `com.ca.jcs.cfg.RegexPluginConfig#PERL5` for possible styles.

Java Constant: MD_VALID_REGEX_STYLE

Type: String

Value: java

writableBy

Restricts where a property's value, or all of a class' property, can be set.

Note: Setting this value to ADD is similar to MD_IS_READ_ONLY, except that the latter throws an error whereas this setting simply throws away attempts to modify the property using the prohibited ADD or MODIFY. The default is null, where both ADD and MODIFY can set property's value.

Note: Applies to any writable property.

Java Constant: MD_WRITABLE_BY

Type: String

Value: Null

Metadata for Associations

assocReverseAttr

See MD_ASSOC_REF_OBJECT_CLASS

Note: Direct association property only.

Type: String

assocRefKeyAttr

See MD_ASSOC_REF_OBJECT_CLASS

Java Constant: MD_ASSOC_REF_KEY_ATTR

Type: String

assocTable

See MD_ASSOC_REF_OBJECT_CLASS

Note: Indirect association property only.

Java Constant: MD_ASSOC_TABLE

Type: String

assocTableObjColumn

See MD_ASSOC_REF_OBJECT_CLASS

Note: Indirect association property only.

Java Constant: MD_ASSOC_TABLE_OBJ_COLUMN

Type: String

assocObjKeyAttr

See MD_ASSOC_REF_OBJECT_CLASS

Note: Indirect association property only.

Java Constant: MD_ASSOC_OBJ_KEY_ATTR

Type: String

assocTableRefColumn

See MD_ASSOC_REF_OBJECT_CLASS

Note: Indirect association property only.

Java Constant: MD_ASSOC_TABLE_REF_COLUMN

Type: String

Metadata for Classes or Attributes

connectorMapTo

Specifies which name to map an object class (including the connector itself) or attribute to in connector-speak. For a dynamic connector, this specifies the name of the native system item to map the attribute to. For example, a JDBC based connector would probably map an object class to the name of a database table, and each property within it to the name of one of its columns.

Java Constant: MD_CONN_MAP_TO

Type: String

connectorMapToAlias

Specifies which name to map an objectclass (including the connector itself) or attribute to in connector-speak, in circumstances where MD_CONN_MAP_TO needs to be mapped to a complicated expression and therefore does not act as a useful reference name in the connector's code.

For more information, see LDA which has ambiguous mappings for many objectclasses/attributes or ORA and SQL JDBC-based connectors which set:

```
<code>MD_CONN_MAP_TO=[some complicated SQL SELECT statement]</code>
```

and

```
<code>MD_CONN_MAP_ALIAS=account</code>
```

Java Constant: MD_CONN_MAP_ALIAS

Type: String

connectorMapToAmbiguous

Specifies the rare cases where a single LDAP attribute ambiguously maps to more than one connector objectclass / attribute. The order of the choices is important, as the first choice is the default when no explicit choice is made and a new object is created. For example, the LDA connector has a number of such ambiguities:

- `eTLDAAccount` is mapped to any one of the objectclasses `person`, `residentialPerson`, `organizationalPerson`, `inetOrgPerson`.
- `eTLDAContainer` is mapped to any one of the objectclasses, `organizationalUnit`, or `organization`, which each have different naming attributes. Consequently, `eTLDAContainerName` makes use of a special syntax which ties its ambiguous choice to the type of the parent object:

```
<metadata name="connectorMapToAmbiguous">
  <value>
    <sequenceValue>
      <val><strValue>organizationalUnit:ou</strValue></val>
      <val><strValue>organization:o</strValue></val>
    </sequenceValue>
  </value>
</metadata>
```

Java Constant: MD_CONN_MAP_TO_AMBIGUOUS

Type: Sequence

connectorMapToAmbiguousChoiceAttr

Specifies that an attribute with ambiguous mappings can optionally name another LDAP attribute which will be used to pass in an explicit choice when new instances of the objectclass are add()ed. For example, in the LDA connector, `eTLDAAccountName` can be either `cn=` or `uid=`, therefore, its metadata specifies:

```
<metadata name="connectorMapToAmbiguousChoiceAttr">
<value>
<strValue>eTLDANamingAttribute</strValue>
</value>
</metadata>
```

This means that an account creation would specify `eTLDANamingAttribute=uid` to explicitly state the desired choice of connector-speak naming attribute.

Java Constant: MD_CONN_MAP_TO_AMBIGUOUS_CHOICE_ATTR

Type: String

connectorMapToScriptLang

Where MD_CONN_MAP_TO is a script, this field specifies its language (for example, JavaScript).

Specifies the MD_CONN_MAP_TO language (for example, JavaScript), where MD_CONN_MAP_TO is a script.

Java Constant: MD_CONN_MAP_TO_LANG

Type: String

Note: Not currently supported.

displayName

Displays a friendly name for entity.

Java Constant: MD_DISPLAY_NAME

Type: String

description

Describes the class or attribute, for programmer use.

Language: Always in English

Type: String

Stored Procedures

Stored procedure arguments that do not have a common SQL type reported by their drivers (that is, the driver returns `java.sql.Type.OTHER = 1111`) are treated as `Type.VARCHAR (16)`, assuming that the driver will convert from string to the argument's desired native type. For example, this is known to work for Oracle `NVARCHAR2` arguments. Where this does not work, your stored procedure will not be invoked and a failure message ending with "Invalid column type" appears.

Therefore, we recommend that you use basic types (like those related to `VARCHAR`) for stored procedure arguments where possible, and verify other argument types against a single stored procedure against your desired vendor/version before proceeding to wide spread usage.

Make sure that the native type of account/group table column you select as the key during the Map Membership Table wizard step in Connector Xpress matches the type of corresponding columns in your chosen membership table. If they do not match, then membership information will not be retrieved successfully. We recommend that you use strict constraints as much as possible. For example, if the column you select as the group naming attribute is of type `NVARCHAR2` and the matching column is of type `VARCHAR2` in the membership table, then looking up the groups an account belongs to will return an empty list (or least certainly be missing groups which have multi-byte characters in their names).

Avoid using the percentage (%) character in stored procedure names, as it acts as a wildcard in queries and is known to cause a problem for Connector Xpress when it appears in Oracle stored procedure names. However, you can use the underscore character (_) as wildcard against all vendors.

Important! Connector Xpress and the Java CS and read and write data from stored procedures through their arguments, but cannot verify the validity of the code of any stored procedures you choose to bind to. We recommend that you verify the validity of the code of any stored procedures you choose to bind to.

View Stored Procedures

To view the stored procedures you have configured for an endpoint type in the Operations Bindings Editor, or edit them using the Connector Xpress Wizard, we recommend that you keep the Connector Xpress project file (.con) you used to create them.

After you deploy an endpoint type, you need to open the Connector Xpress project file where your bindings are stored to view the stored procedure bindings.

To view stored procedure bindings

1. Open the Connector Xpress project file where your bindings are stored.
2. Do *one* of the following:
 - a. Click Bindings in the upper left pane of the Connector Xpress window.
The bindings are displayed in the bindings tree in the rightmost pane of the Connector Xpress window.
 - b. Click Wizard in the upper left pane of the Connector Xpress window.
Connector Xpress displays the bindings on the Stored Procedures screen.

Note: You can also use the JXplorer interface to view the raw XML for your bindings in the *eTOPBindingsMetaData* attribute of the endpoint type. You can see the raw metadata in the *eTMetaData* attribute of the endpoint type.

More information:

[Projects](#) (see page 35)

[Operations Bindings Editor](#) (see page 18)

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

Stored Procedure Column Type Considerations

Three JDBC vendors, Microsoft, Sybase, and Ingres, have peculiarities regarding the following Column Types of their Stored Procedure parameters:

- **In**—Values can only be passed in
- **InOut**—Values can both be passed in and back
- **Out**—Value passed back only

As MSSQL and Sybase do not distinguish between InOut and Out, the Stored procedures screen displays a check box called Output Only for MSSQL and Sybase Stored Procedure parameters. Select this check box to specify that the parameter displayed on a row is an Out parameter, rather than an InOut parameter.

The Stored Procedures screen also displays this column for all Ingres Stored Procedures parameters that are all natively treated as being of type InOut, allowing the Out case to be specified.

Note: It is not currently possible to clarify that a parameter is of type In.

Stored Procedure Bindings

Information about stored procedures is captured in operation bindings and is displayed in the Operations Bindings Editor when a project is reloaded.

You can use the Operation Binding Editor to change some of the settings that you cannot change with the wizard.

Important! Use the Operations Binding editor at your own risk. Removal or alteration of important objects in the metadata may render it inoperable.

The diagram below displays the most important parts of the JCS HOME/conf/xsd/opbindings.xsd XML schema that you can change:



Each XML fragment can consist of multiple **OPBindingType** elements as outlined above, each of which has:

- A guard element specifying the operation (ADD / MODIFY and so forth) and target object class(es) (account / group) the binding is to be executed for.
- A payload that contains information about the stored procedure to be executed.
- (JDBC Only) A timing that was specified previously in the Stored Procedures screen.

- When set to true, the strict completion Boolean signifies that a failure while executing the payload (whether an exception or a non-null textual error status return) should cause the framework to treat the whole LDAP operation as having failed. For Instead Of bindings, strict completion is always true.

More information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[Projects](#) (see page 35)

[Operations Bindings Editor](#) (see page 18)

JDBC Wizard Does Not Display All Steps for Large Stored Procedure Mappings

If you map a large number of Stored Procedures using the JDBC wizard, there may be more steps in the wizard than are displayed on the screen. As you step through the wizard, Connector Xpress may not highlight the current step in the wizard on the left-most side of the screen.

Chapter 5: Importing and Exporting Data

You can edit previously exported XML data, export the metadata to an XML file, or merge XML data into the currently loaded metadata, or create a Mapping Summary file to help you debug and read logs.

This section contains the following topics:

- [Import XML Data Model](#) (see page 75)
- [Export Data Model to an XML File](#) (see page 75)
- [Merge a Modified Data Model](#) (see page 76)
- [Mapping Summary File](#) (see page 76)
- [Export a Mapping Summary File](#) (see page 77)

Import XML Data Model

To edit previously exported XML data, you can import an XML data model file.

To import an XML data model file

1. Click File, Import Data Model XML.
The Import Data Model from XML file dialog appears.
2. Navigate to the location of the XML file you want to import, and then click OK.

The Data Model XML file is loaded, and the XML tree appears in the Edit pane of the Connector Xpress window.

Export Data Model to an XML File

You can export the metadata for the currently open project into an XML file.

To export a Data Model to an XML File

1. [Open the project](#) (see page 35) you want to export the metadata for.
2. Click File, Export Data Model XML.
The Export Data Model to XML file dialog appears.
3. Specify the location where you want to save the file, and then click OK.
Connector Xpress exports and saves the metadata in the file you specified.

Merge a Modified Data Model

You can select a data model XML file to load and merge into the currently loaded metadata.

To merge a modified data model

1. [Open the existing project](#) (see page 35) you want to merge the metadata for.
2. Click Tools, Merge Modified Data Model.

The Merge Data Model from dialog appears.

3. Navigate to the location of the file you want to merge the metadata with, and then click OK.

Connector Xpress overwrites or appends the data model file of the file you opened to the currently loaded data model.

Mapping Summary File

You can use the Mapping Summary file to help you debug and read logs. For example, you can use the Mapping Summary file if the attribute names in the generic schema are not descriptive, for example, eTDYN-str-i-01.

Generating a mapping summary file is also important when you want to write batch scripts for automation.

The Mapping Summary file is a HTML file that displays the following information for a selected endpoint type, or a currently open project:

- Admin LDAP attribute name
- Logical name
- Native name
- Display name
- Attribute values for each class defined in the dynamic name space

Export a Mapping Summary File

You can export a Mapping Summary file to help you debug and read logs. You can export a mapping file for the currently loaded project, or a selected endpoint type in the Provisioning Servers Tree.

To export a mapping summary file

1. Do *one* of the following:
 - a. In the Provisioning Servers tree, right-click the endpoint type you want to export a Mapping Summary file for, then click Export Mapping Summary File.
 - b. [Open the project](#) (see page 35) you want to export a Mapping Summary file for, and then click File, Export Summary File.

The Export Mapping Summary File dialog appears.

2. Specify the location where you want to save the file, and then click OK.

Connector Xpress saves the Mapping Summary file to the location you specified.

Chapter 6: Managing Endpoints

To manager the accounts, users, groups, and other resource objects in your environment, you must acquire endpoints. Once you acquire an endpoint, you must explore the endpoint for your objects to be managed, and correlate the objects to global users. You can specify how an endpoint is going to be explored and correlated. For example you can specify how many levels in the container are explored, one level or the just the current sub-tree.

You may choose whether the correlate function creates any users that are not present or whether it associates accounts with no matching users to the [default user] global user.

You can also create, clean, and delete endpoint types.

This section contains the following topics:

[How You Acquire and Manage Endpoints](#) (see page 80)

[Acquire, Explore and Correlate an Endpoint](#) (see page 81)

[Create a New Endpoint Type](#) (see page 82)

[Delete an Endpoint Type](#) (see page 82)

[Clean an Endpoint Type](#) (see page 83)

[Delete an Endpoint](#) (see page 83)

How You Acquire and Manage Endpoints

Acquiring an endpoint automatically populates the Provisioning Server with accounts and other objects found in the acquired endpoint. An acquired endpoint is any application or computer managed by the Provisioning Server.

Acquiring an endpoint consists of creating it, exploring it, and correlating it. Correlating an endpoint is the process of examining accounts on it, and potentially creating users automatically and setting global user attributes from account attributes.

To manage an endpoint using Connector Xpress, do the following:

1. Create the endpoint in the Provisioning Server. When you register an endpoint, you are declaring it as an endpoint managed by the Provisioning Server.
2. Explore its contents. When you explore an endpoint, Connector Xpress finds the objects in the endpoint and stores a record of them in the Provisioning Server.
3. Correlate the explored accounts. When you correlate accounts on an endpoint, the Provisioning Server associates them with a global user in the Provisioning directory. You may choose whether the correlate function creates any global users that are not present or whether it associates accounts with no matching global user to the [default user] global user.
4. Update global user fields from accounts. Whether or not you chose to let correlate function create missing users, you can optionally run an additional function that sets (or refreshes) selected user attributes using account's attribute values that you select.

More Information:

[Explore/Correlate Endpoint Dialog](#) (see page 90)

[Create New Endpoint Dialog](#) (see page 91)

Acquire, Explore and Correlate an Endpoint

To acquire an endpoint, you must first register it in the Provisioning Server, that is, specify the connection details and credentials of the endpoint so that the connector can communicate with it, then you must explore the endpoint for your objects to be managed, and correlate the objects to global users.

To acquire, explore and correlate an endpoint

1. In the Provisioning Servers tree, right click the Endpoint node that you want to explore or correlate, then click Acquire Endpoint.

The Create New Endpoint dialog appears.

2. Complete the fields on the Create New Endpoint dialog, then click OK.

You have specified the connection details and credentials of the endpoint.

3. In the Provisioning Servers tree, right click the endpoint node that you created and then click Explore/ correlate endpoint.

The Explore/Correlate endpoint dialog appears.

4. Complete the fields on the Explore/Correlate Endpoint dialog, then click OK.

You have specified how the how Connector Xpress explores and correlate the endpoint.

More Information:

[Create New Endpoint Dialog](#) (see page 91)

[Explore/Correlate Endpoint Dialog](#) (see page 90)

Create a New Endpoint Type

You can create a new endpoint type using the loaded metadata.

Note: You can only create a new endpoint type when a Connector Xpress project is open.

To create a new endpoint type

1. [Open an existing project](#) (see page 35).
2. In the Provisioning Servers tree, right-click the Endpoint Types node, then click Create new Endpoint Type.

The Create New Endpoint Type dialog appears.

3. Complete the fields on the dialog and then click OK.

You have defined the name of your new endpoint type and added the endpoint type to the Endpoint Types node of the Provisioning Server you specified.

More information:

[Create New Endpoint Type Dialog](#) (see page 91)

Delete an Endpoint Type

You can delete an endpoint type using Connector Xpress when you no longer need to manage that endpoint type and the endpoints under it in the Provisioning Server.

Important! We strongly recommend that you use only Connector Xpress to remove endpoint types associated with dynamic connectors, to ensure successful deletion.

To delete an endpoint type, expand the Endpoints Type node in the Provisioning Servers tree, right-click the endpoint type you want to delete, and then click Delete Endpoint Type

Connector Xpress deletes the endpoint type, the associated account templates, and any child endpoints, however data on the endpoint is unchanged.

Clean an Endpoint Type

When you need to delete all account templates and endpoints from a selected endpoint type, you can clean the endpoint type. Cleaning an endpoint type removes all endpoint objects from the Provisioning Server, and recreates the default account template object.

To clean an endpoint type

1. In the Provisioning Servers tree, right-click the endpoint type you want to clean, then click Clean Endpoint Type.

The Clean Endpoint Type Objects dialog appears.

2. Confirm that you want to delete the endpoint type.

Connector Xpress deletes all account templates and endpoints of the selected endpoint type from the Provisioning Directory. Data on the endpoints is not affected.

Delete an Endpoint

You can remove a connector using Connector Xpress when you no longer need to manage that connector in the Provisioning Manager.

Important! We strongly recommend that you use only Connector Xpress to remove endpoints associated with dynamic connectors, to ensure successful deletion.

To delete an endpoint, expand the Endpoints node in the Provisioning Servers tree, right-click the endpoint you want to delete, and then click Delete Endpoint.

Connector Xpress deletes the connector, and removes the endpoint and associated account templates, but the data on the endpoint data is unchanged.

More information:

[Managing Endpoints](#) (see page 79)

Chapter 7: Screens and Dialogs

The following section describes the dialogs and screens in Connector Xpress.

This section contains the following topics:

- [Set Preferences Dialog](#) (see page 85)
- [Select Data Source for new project Dialog](#) (see page 86)
- [Source Types Dialog](#) (see page 87)
- [Edit Source Dialog](#) (see page 87)
- [Explore/Correlate Endpoint Dialog](#) (see page 90)
- [Enter Password for Data Source Dialog](#) (see page 90)
- [Create New Endpoint Type Dialog](#) (see page 91)
- [Create New Endpoint Dialog](#) (see page 91)
- [Provisioning Server Password Required Dialog](#) (see page 92)
- [Edit Connector Server Configuration Dialog](#) (see page 93)
- [Provisioning Server Details Dialog](#) (see page 94)
- [Edit Metadata for Endpoint Type Dialog](#) (see page 95)

Set Preferences Dialog

You can use the Set Preferences Dialog to change various aspects of the way Connector Xpress looks and behaves, such as its look and feel, search limits, and command parameters.

This dialog contains the following fields:

Look and Feel

Changes the way Connector Xpress looks. For example, Windows or CDE (Common Desktop Environment) Motif.

Search Size Limit

Specifies the maximum number of search results the Provisioning Servers tree will attempt to retrieve for a particular node. Connector Xpress uses this value to prevent a large drain on system resources, for example if a large number of policies exist.

Command to launch web browser

Specifies the command used to launch the web browser.

Default: (Windows) "rundll32" "url.dll,FileProtocolHandler"

Default: (Solaris)"/usr/sfw/bin/mozilla"

More Information:

[Set Preferences](#) (see page 21)

Select Data Source for new project Dialog

The Select Data Source for new project dialog lets you select a previously configured data source for a new project, add or edit an existing data source, or remove a data source.

This dialog contains the following fields:

Name

Displays the name of a configured data source you can use to create a new project.

Type

Displays the type of configured sources you can use to create a new project.

Add

Displays the Source Types dialog where you can choose from available source types and then configure your new data source.

Edit

Displays the Edit Source dialog where you can edit the connection details for the specified data source.

Remove

Removes the data source from the list box.

More information:

[Create a Project](#) (see page 36)

[Set Up Data Sources](#) (see page 13)

Source Types Dialog

The Source Types dialog lets you select an available source type for the data source.

This dialog contains the following fields:

Available Source Types

Displays the available source types you can use as the basis for your connector definition.

JDBC

Lets you map to database tables and stored procedures.

JNDI

Lets you map to object classes and attributes contained in a directory server.

More Information:

[Create a Project](#) (see page 36)

[Set Up Data Sources](#) (see page 13)

Edit Source Dialog

The Edit Source dialog lets you specify the name and connection details for your data source. For JDBC data sources, this dialog lets you construct an appropriate JDBC URL for the database you select. For JNDI data sources, the dialog lets you specify the details of an LDAP connection.

Note: The fields in this dialog will vary, depending on the type of data source selected.

This dialog contains the following fields:

Name

(JNDI, JDBC) Specifies the name of the data source.

Note: We recommend that you chose a name that clearly describes what the data source is for.

Server Name

(JNDI and JDBC only) Specifies the host name of the server you want to access.

Database Type

(JDBC only) Specifies the type of database connection you want to make:

- Oracle
- Microsoft SQL Server
- DB2
- Ingres
- Sybase

Username

(JDBC only) Specifies the username the connector uses to log onto the target database endpoint.

Defaults

Oracle: System

Microsoft SQL Server: sa

DB2: db2admin

Native

(JDBC and Ms SQL Server only) Specifies that MSSQL Native Authentication on Windows is activated. The following is added to the JDBC URL used for the connection:

`integratedSecurity=true`

Server name

(JDBC only) Specifies the hostname of the database server.

SID

(JDBC and Oracle only) Specifies the Oracle SID.

Port

(JNDI and JDBC only) Specifies the port number the connector uses for communication with the endpoint.

Defaults

JNDI: 389

Oracle: 1521

Microsoft SQL Server: 1433

DB2: 6789

Ingres: 117

Database

(JDBC only) Specifies the database name (most vendors including MS SQL) or its SID (Oracle).

Defaults

Microsoft SQL Server: master

Oracle SID: ORCL

JDBC URL

(JDBC only) Specifies the JDBC URL. Connector Xpress automatically forms the URL as you complete the previous fields.

Note: To edit this field, select the Edit check box.

Bind DN

(JNDI only) Specifies the distinguished name of the user used to bind to the directory endpoint.

Anonymous

(JNDI only) Signifies that no bind DN will be used and instead, an anonymous bind to the directory endpoint will be used.

Use TLS

Controls whether TLS connections are used between Connector Xpress and the endpoint, for example, when you are discovering the schema of the endpoint while doing the mappings.

Base DN

(JNDI only) Specifies the top level of the DIT (Directory Information Tree) you want to manage within the directory endpoint.

Edit

Lets you edit the JDBC URL field.

Test**JNDI and JDBC**

Attempts to connect to the specified directory or database using the information provided (including prompting the user for a password where necessary).

More Information:

[Create a Project](#) (see page 36)

[Set Up Data Sources](#) (see page 13)

Explore/Correlate Endpoint Dialog

The Explore/Correlate Endpoint dialog lets you specify how you want to explore and correlate the endpoint.

This dialog contains the following fields:

Explore Endpoint: Sub Tree

Specifies that all the containers in the sub-tree should be included in the endpoint exploration.

Explore Endpoint: One Level

Specifies that only one level of containers are included in the endpoint exploration.

Correlate Accounts with Users: Use existing Users

Specifies that each account is linked to the user with the same name. The correlation links only existing users; none are created.

Correlate Accounts with Users: Create Users as needed

Specifies that each account is linked to the user sharing the same name. If the user does not exist, the correlation process creates the user.

More Information:

[Deploy the Connector](#) (see page 40)

[Acquire, Explore and Correlate an Endpoint](#) (see page 81)

[How You Acquire and Manage Endpoints](#) (see page 80)

Enter Password for Data Source Dialog

Use the Enter Password for Data Source dialog to specify the password for the data source used for the basis of the connection.

This dialog contains the following fields:

Source name

Displays the name of the data source specified when the data source is created.

Source URL

Displays the connection URL of the data source.

Source type

Displays the type of the data source, (for example, JNDI, JDBC).

User name

Displays the name of the user used to connect to the data source.

Password

Specifies the password of the user Connector Xpress uses to connect to the data source.

More Information:

[Edit a Project](#) (see page 37)

[Create a Project](#) (see page 36)

Create New Endpoint Type Dialog

The Create New Endpoint Type dialog lets you define the name of the endpoint type and optionally, the connector server you want to use to manage the endpoint type.

This dialog contains the following fields:

New Endpoint Type

Defines the name of the endpoint type you want to create. Connector Xpress automatically provides the endpoint type based on the value entered in the wizard. You can accept the default or enter another name.

Connector Servers

Specifies the connector server you want to manage the new endpoint type.

More Information:

[Create a New Endpoint Type](#) (see page 82)

Create New Endpoint Dialog

The Create New Endpoint dialog lets you create a new endpoint for JDBC and JNDI connectors.

This dialog contains the following fields:

Endpoint Name

Specifies the endpoint name. This is an arbitrary name designated by the user and should be very descriptive.

Username

(JDBC only) Specifies the username used to log onto the target database.

Password

Specifies the password that used to log onto the target endpoint.

Connection URI

(JDBC only) Specifies the secure connection URI.

Bind User DN

(JNDI only) Specifies the distinguished name of the user used to bind to the directory endpoint.

LDAP URL

(JNDI only) Specifies the LDAP URL.

Base DN

(JNDI only) Specifies the top level of the DIT (Directory Information Tree) you want to manage within the directory endpoint.

Use TLS

Specifies whether TLS connections are used between CS and directory endpoint.

More Information:

[Deploy the Connector](#) (see page 40)

[How You Acquire and Manage Endpoints](#) (see page 80)

[Acquire, Explore and Correlate an Endpoint](#) (see page 81)

Provisioning Server Password Required Dialog

The Provisioning Server Password Required dialog lets you specify the password for the selected Provisioning Server.

This dialog contains the following field:

Password

Specifies the password needed to access the selected Provisioning Server.

More Information:

[Deploy the Connector](#) (see page 40)

Edit Connector Server Configuration Dialog

The Edit Connector Server Configuration dialog lets you set the hosting, routing, and managed connector information for the server.

This dialog contains the following fields:

Connector Server Host Name

Defines the host name of the machine running the connector server.

Port

Defines the TCP port on the connector server used for non-TLS connections.

TLS Port

Defines the TCP port on the connector server used for TLS connections.

Provisioning Server

(Optional) Lets you specify the Provisioning Server you want to operate on, as opposed to the Provisioning Server that selected in the Provisioning Servers tree.

Use TLS

Specifies that the connections between the Provisioning Server and the connector server are encrypted TLS connections. If not selected, connections between the Provisioning Server and the connector server are in clear text.

Connector Server Password

Defines the password used to access the connector server. This is usually configured when you install the server.

Note: If you edit the Connector Server Host Name, Ports, or Use TLS fields, you must re-enter the password.

Verify Password

Lets you verify the password you specified used to access to the connector server.

Make this the default Connector Server for Provisioning Server

Specifies that this Connector Server should be used as the default for the Provisioning Server.

Note: Any branch (endpoint type or endpoint) that is not explicitly associated with a CS Config is routed through the default CS. As a result, a mismatch could happen when an endpoint type or endpoint is created without explicitly specifying the managing Connector Server. For example, when the default CS is the C++ Connector Server and the user has created a dynamic endpoint type without explicitly specifying a Java CS as the managing Connector Server.

Important! This check box can be selected for multiple connector servers on a single Provisioning Server. If you select this check box a random CS is chosen when the request is routed.

Object Handle

Specifies the object handles of the endpoint types being managed on the selected Provisioning Server.

Note: We recommend that you manage this list using the Set Managing Connector Server procedure.

More Information:

[Set a Managing Connector Server](#) (see page 28)

[Edit a Connector Server Configuration](#) (see page 27)

[Create a Connector Server Configuration](#) (see page 26)

Provisioning Server Details Dialog

The Provisioning Server Details dialog lets you configure a connection to a Provisioning Server to which you can deploy a connector.

This dialog contains the following fields:

Host Name

Defines the name of the host machine on which the Provisioning Server is running.

User Domain

Defines the domain containing the user whose credentials the connector uses to log onto the Provisioning Server.

User Name

Defines the username that Connector Xpress uses to log onto the server.

Use TLS

Specifies that a secure TLS connection to the server is established. This means that the connector does not transmit the password in clear text. If this is not selected the password is transmitted in clear text.

Key

Defines the name of the configuration option that the connector uses to connect to the Provisioning Server.

Value

Defines the value that the configuration option is set to.

Key column

Displays the full list of configuration options that the connector uses to connect to the Provisioning Server.

Value column

Displays the corresponding values for the configuration options displayed in the Key column.

More Information:

[Add and Configure a Provisioning Server](#) (see page 24)

[Edit Provisioning Server Details](#) (see page 24)

Edit Metadata for Endpoint Type Dialog

The Edit Metadata for Endpoint Type dialog lets you manually edit metadata.

This dialog contains the following fields:

Node Type

Displays a list of possible node types that you can add using the Add button. The contents of the list will depend on the type of node currently selected in the XML tree.

Add

Lets you adds a new node to the XML tree of the type specified in the Node Type drop-down list.

Edit

Lets you edit the currently selected node in the XML tree.

Remove

Lets you remove the currently selected node from the XML tree.

XML Tree

Displays an editable XML tree view of the metadata.

Save

Saves your changes.

Export

Lets you export a data model to an XML file.

More Information:

[Edit Metadata](#) (see page 43)

Chapter 8: Wizard Screens

The following sections describe each screen in the Connector Xpress Wizard.

This section contains the following topics:

[General Screen](#) (see page 97)

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[Map Account or Group Table Screen](#) (see page 115)

[Map Account or Map Group Confirm Screen](#) (see page 118)

[Map Group Table Screen](#) (see page 121)

[Map Membership Table Screen \(JDBC Only\)](#) (see page 125)

[Map Containers Screen \(JNDI Only\)](#) (see page 126)

General Screen

The General screen lets you specify the name and details of a new endpoint type.

This screen contains the following fields:

Endpoint Type Name

Specifies the name of the endpoint type that will be managed in the Provisioning Manager.

Note: You can add endpoints under this endpoint type that all share the same metadata mappings using the Provisioning Manager after completing the wizard.

Version

Specifies a version that will be included in the XML metadata generated for the connector.

Description

(Optional) Specifies a description of the endpoint type.

More Information:

[How You Use the Connector Xpress Wizard](#) (see page 38)

Stored Procedures Screen (JDBC Only)

The Stored Procedures screen lets you register pre-existing database stored procedures against selected operations on accounts and groups.

Note: For each account or group operation you choose, the wizard displays different stored procedure screens. For example, if you specify that you want to call a stored procedure before an account is created, the wizard displays the Map Account Create pre screen, where you can map the stored procedures parameters to the required account provisioning attributes.

This screen contains the following fields:

Invoke Stored Procedure

Specifies that you want to trigger a stored procedure on account creation, update, deletion, or modification.

Note: This operation and any registered stored procedures are all executed in the same database transaction.

Execution Order

Specifies when the stored procedure will be executed.

Before

Specifies that the stored procedure will be executed before the selected operation is performed.

Instead Of

Specifies that the stored procedure will be executed in place of the selected operation.

After

Specifies that the stored procedure will be executed after the selected operation is performed.

More Information

[How You Use the Connector Xpress Wizard](#) (see page 38)

[Stored Procedure Bindings](#) (see page 72)

[Stored Procedure Column Type Considerations](#) (see page 71)

Map Account Create pre, op and post screens

The Map Account Create pre, op and post screens let you select a stored procedure that you want to run before, instead of, or after an account is created.

These screens contain the following fields:

Schema

Specifies the schema (for example, table space or database) containing the stored procedures that are displayed in the Select Procedure field.

Select Procedure

Specifies the stored procedure you want to run before, instead of, or after a selected operation is performed.

Nullable

Indicates whether the parameter accepts (or outputs) a null value.

Parameter Name

Specifies the name of the parameter as specified in the store procedure's definition.

SQL Data Type

Displays the SQL type for the named parameter.

Length

Displays the length of the parameter value, if known.

Column Type

Indicates the type of parameter.

Value: In—The value is passed into the stored procedure through the parameter or type.

Value: InOut—The value is passed both in and out of the stored procedure.

Value: Out—The value is passed out from the stored procedure.

Column

Lets you map each parameter of the selected stored procedure, to the target Provisioning Manager LDAP attribute (or an additional runtime context attribute) in this column.

The values in the drop-down menu with asterisks on either side are runtime context values (for example, the Distinguished Name for the account targeted by an operation or operation specific value such as the "new name" in a MODIFY RN rename operation). The remaining values are the Admin account attribute values you mapped in the previous wizard step.

Note: The *ErrorStatus* value can be used to pass a descriptive error string back from a stored procedure and can cause a failure if "strict completion" is true for the binding.

Note: *AddModfiy_AttrsAsXML* can be useful for stored procedures bound to ADD or MODIFY operations where the endpoint has good in-built XML support. The entire request can be passed in a single XML string that can then be broken down within the stored procedure, if desired.

Multivalued

Specifies that the parameter can have multiple values and will need a flattening style.

Flattening Style

Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

Flattening Mode

Displays the flattening mode for multivalued parameters where changes can be expressed in terms of values ADDED or REMOVED, or alternatively, by passing in the complete new list of values (REPLACE).

More Information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Account Modify pre, op and post screens

The Map Account Modify pre, op and post screens let you select a stored procedure that you want to run before, instead of, or after an account is modified.

These screens contain the following fields:

Schema

Specifies the schema (for example, table space or database) containing the stored procedures that are displayed in the Select Procedure field.

Select Procedure

Specifies the stored procedure you want to run before, instead of, or after a selected operation is performed.

Nullable

Indicates whether the parameter accepts (or outputs) a null value.

Parameter Name

Specifies the name of the parameter as specified in the store procedure's definition.

SQL Data Type

Displays the SQL type for the named parameter.

Length

Displays the length of the parameter value, if known.

Column Type

Indicates the type of parameter.

Value: In—The value is passed into the stored procedure through the parameter or type.

Value: InOut—The value is passed both in and out of the stored procedure.

Value: Out—The value is passed out from the stored procedure.

Column

Lets you map each parameter of the selected stored procedure, to the target Provisioning Manager LDAP attribute (or an additional runtime context attribute) in this column.

The values in the drop-down menu with asterisks on either side are runtime context values (for example, the Distinguished Name for the account targeted by an operation or operation specific value such as the "new name" in a MODIFY RN rename operation). The remaining values are the Admin account attribute values you mapped in the previous wizard step.

Note: The *ErrorStatus* value can be used to pass a descriptive error string back from a stored procedure and can cause a failure if "strict completion" is true for the binding.

Note: *AddModfiy_AttrsAsXML* can be useful for stored procedures bound to ADD or MODIFY operations where the endpoint has good in-built XML support. The entire request can be passed in a single XML string that can then be broken down within the stored procedure, if desired.

Multivalued

Specifies that the parameter can have multiple values and will need a flattening style.

Flattening Style

Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

Flattening Mode

Displays the flattening mode for multivalued parameters where changes can be expressed in terms of values ADDED or REMOVED, or alternatively, by passing in the complete new list of values (REPLACE).

More Information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Account Delete pre, op and post screens

The Map Account Delete pre, op and post screens let you select a stored procedure that you want to run before, instead of, or after an account is deleted.

These screens contain the following fields:

Schema

Specifies the schema (for example, table space or database) containing the stored procedures that are displayed in the Select Procedure field.

Select Procedure

Specifies the stored procedure you want to run before, instead of, or after a selected operation is performed.

Nullable

Indicates whether the parameter accepts (or outputs) a null value.

Parameter Name

Specifies the name of the parameter as specified in the store procedure's definition.

SQL Data Type

Displays the SQL type for the named parameter.

Length

Displays the length of the parameter value, if known.

Column Type

Indicates the type of parameter.

Value: In—The value is passed into the stored procedure through the parameter or type.

Value: InOut—The value is passed both in and out of the stored procedure.

Value: Out—The value is passed out from the stored procedure.

Column

Lets you map each parameter of the selected stored procedure, to the target Provisioning Manager LDAP attribute (or an additional runtime context attribute) in this column.

The values in the drop-down menu with asterisks on either side are runtime context values (for example, the Distinguished Name for the account targeted by an operation or operation specific value such as the "new name" in a MODIFY RN rename operation). The remaining values are the Admin account attribute values you mapped in the previous wizard step.

Note: The *ErrorStatus* value can be used to pass a descriptive error string back from a stored procedure and can cause a failure if "strict completion" is true for the binding.

Note: *AddModfiy_AttrsAsXML* can be useful for stored procedures bound to ADD or MODIFY operations where the endpoint has good in-built XML support. The entire request can be passed in a single XML string that can then be broken down within the stored procedure, if desired.

Multivalued

Specifies that the parameter can have multiple values and will need a flattening style.

Flattening Style

Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

Flattening Mode

Displays the flattening mode for multivalued parameters where changes can be expressed in terms of values ADDED or REMOVED, or alternatively, by passing in the complete new list of values (REPLACE).

More Information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Account Modify RN pre, op and post screens

The Map Account Modify RN pre, op and post screens let you select a stored procedure that you want to run before, instead of, or after an account is renamed.

These screens contain the following fields:

Schema

Specifies the schema (for example, table space or database) containing the stored procedures that are displayed in the Select Procedure field.

Select Procedure

Specifies the stored procedure you want to run before, instead of, or after a selected operation is performed.

Nullable

Indicates whether the parameter accepts (or outputs) a null value.

Parameter Name

Specifies the name of the parameter as specified in the store procedure's definition.

SQL Data Type

Displays the SQL type for the named parameter.

Length

Displays the length of the parameter value, if known.

Column Type

Indicates the type of parameter.

Value: In—The value is passed into the stored procedure through the parameter or type.

Value: InOut—The value is passed both in and out of the stored procedure.

Value: Out—The value is passed out from the stored procedure.

Column

Lets you map each parameter of the selected stored procedure, to the target Provisioning Manager LDAP attribute (or an additional runtime context attribute) in this column.

The values in the drop-down menu with asterisks on either side are runtime context values (for example, the Distinguished Name for the account targeted by an operation or operation specific value such as the "new name" in a MODIFY RN rename operation). The remaining values are the Admin account attribute values you mapped in the previous wizard step.

Note: The *ErrorStatus* value can be used to pass a descriptive error string back from a stored procedure and can cause a failure if "strict completion" is true for the binding.

Note: *AddModfiy_AttrsAsXML* can be useful for stored procedures bound to ADD or MODIFY operations where the endpoint has good in-built XML support. The entire request can be passed in a single XML string that can then be broken down within the stored procedure, if desired.

Multivalued

Specifies that the parameter can have multiple values and will need a flattening style.

Flattening Style

Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

Flattening Mode

Displays the flattening mode for multivalued parameters where changes can be expressed in terms of values ADDED or REMOVED, or alternatively, by passing in the complete new list of values (REPLACE).

More Information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Group Modify pre, op and post screens

The Map Group Modify pre, op and post screens let you select a stored procedure that you want to run before, instead of, or after a group is modified.

These screens contain the following fields:

Schema

Specifies the schema (for example, table space or database) containing the stored procedures that are displayed in the Select Procedure field.

Select Procedure

Specifies the stored procedure you want to run before, instead of, or after a selected operation is performed.

Nullable

Indicates whether the parameter accepts (or outputs) a null value.

Parameter Name

Specifies the name of the parameter as specified in the store procedure's definition.

SQL Data Type

Displays the SQL type for the named parameter.

Length

Displays the length of the parameter value, if known.

Column Type

Indicates the type of parameter.

Value: In—The value is passed into the stored procedure through the parameter or type.

Value: InOut—The value is passed both in and out of the stored procedure.

Value: Out—The value is passed out from the stored procedure.

Column

Lets you map each parameter of the selected stored procedure, to the target Provisioning Manager LDAP attribute (or an additional runtime context attribute) in this column.

The values in the drop-down menu with asterisks on either side are runtime context values (for example, the Distinguished Name for the account targeted by an operation or operation specific value such as the "new name" in a MODIFY RN rename operation). The remaining values are the Admin account attribute values you mapped in the previous wizard step.

Note: The *ErrorStatus* value can be used to pass a descriptive error string back from a stored procedure and can cause a failure if "strict completion" is true for the binding.

Note: *AddModfiy_AttrsAsXML* can be useful for stored procedures bound to ADD or MODIFY operations where the endpoint has good in-built XML support. The entire request can be passed in a single XML string that can then be broken down within the stored procedure, if desired.

Multivalued

Specifies that the parameter can have multiple values and will need a flattening style.

Flattening Style

Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

Flattening Mode

Displays the flattening mode for multivalued parameters where changes can be expressed in terms of values ADDED or REMOVED, or alternatively, by passing in the complete new list of values (REPLACE).

More Information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Group Create pre, op and post screens

The Map Group Create pre, op and post screens let you select a stored procedure that you want to run before, instead of, or after a group is created.

These screens contain the following fields:

Schema

Specifies the schema (for example, table space or database) containing the stored procedures that are displayed in the Select Procedure field.

Select Procedure

Specifies the stored procedure you want to run before, instead of, or after a selected operation is performed.

Nullable

Indicates whether the parameter accepts (or outputs) a null value.

Parameter Name

Specifies the name of the parameter as specified in the store procedure's definition.

SQL Data Type

Displays the SQL type for the named parameter.

Length

Displays the length of the parameter value, if known.

Column Type

Indicates the type of parameter.

Value: In—The value is passed into the stored procedure through the parameter or type.

Value: InOut—The value is passed both in and out of the stored procedure.

Value: Out—The value is passed out from the stored procedure.

Column

Lets you map each parameter of the selected stored procedure, to the target Provisioning Manager LDAP attribute (or an additional runtime context attribute) in this column.

The values in the drop-down menu with asterisks on either side are runtime context values (for example, the Distinguished Name for the account targeted by an operation or operation specific value such as the "new name" in a MODIFY RN rename operation). The remaining values are the Admin account attribute values you mapped in the previous wizard step.

Note: The *ErrorStatus* value can be used to pass a descriptive error string back from a stored procedure and can cause a failure if "strict completion" is true for the binding.

Note: *AddModfiy_AttrsAsXML* can be useful for stored procedures bound to ADD or MODIFY operations where the endpoint has good in-built XML support. The entire request can be passed in a single XML string that can then be broken down within the stored procedure, if desired.

Multivalued

Specifies that the parameter can have multiple values and will need a flattening style.

Flattening Style

Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

Flattening Mode

Displays the flattening mode for multivalued parameters where changes can be expressed in terms of values ADDED or REMOVED, or alternatively, by passing in the complete new list of values (REPLACE).

More Information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Group Delete pre, op and post screens

The Map Group Delete pre, op and post screens let you select a stored procedure that you want to run before, instead of, or after a group is deleted.

These screens contain the following fields:

Schema

Specifies the schema (for example, table space or database) containing the stored procedures that are displayed in the Select Procedure field.

Select Procedure

Specifies the stored procedure you want to run before, instead of, or after a selected operation is performed.

Nullable

Indicates whether the parameter accepts (or outputs) a null value.

Parameter Name

Specifies the name of the parameter as specified in the store procedure's definition.

SQL Data Type

Displays the SQL type for the named parameter.

Length

Displays the length of the parameter value, if known.

Column Type

Indicates the type of parameter.

Value: In—The value is passed into the stored procedure through the parameter or type.

Value: InOut—The value is passed both in and out of the stored procedure.

Value: Out—The value is passed out from the stored procedure.

Column

Lets you map each parameter of the selected stored procedure, to the target Provisioning Manager LDAP attribute (or an additional runtime context attribute) in this column.

The values in the drop-down menu with asterisks on either side are runtime context values (for example, the Distinguished Name for the account targeted by an operation or operation specific value such as the "new name" in a MODIFY RN rename operation). The remaining values are the Admin account attribute values you mapped in the previous wizard step.

Note: The *ErrorStatus* value can be used to pass a descriptive error string back from a stored procedure and can cause a failure if "strict completion" is true for the binding.

Note: *AddModfiy_AttrsAsXML* can be useful for stored procedures bound to ADD or MODIFY operations where the endpoint has good in-built XML support. The entire request can be passed in a single XML string that can then be broken down within the stored procedure, if desired.

Multivalued

Specifies that the parameter can have multiple values and will need a flattening style.

Flattening Style

Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

Flattening Mode

Displays the flattening mode for multivalued parameters where changes can be expressed in terms of values ADDED or REMOVED, or alternatively, by passing in the complete new list of values (REPLACE).

More Information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Group Modify RN pre, op and post screens

The Map Group Modify RN pre, op and post screens let you select a stored procedure that you want to run before, instead of, or after a group is renamed.

These screens contain the following fields:

Schema

Specifies the schema (for example, table space or database) containing the stored procedures that are displayed in the Select Procedure field.

Select Procedure

Specifies the stored procedure you want to run before, instead of, or after a selected operation is performed.

Nullable

Indicates whether the parameter accepts (or outputs) a null value.

Parameter Name

Specifies the name of the parameter as specified in the store procedure's definition.

SQL Data Type

Displays the SQL type for the named parameter.

Length

Displays the length of the parameter value, if known.

Column Type

Indicates the type of parameter.

Value: In—The value is passed into the stored procedure through the parameter or type.

Value: InOut—The value is passed both in and out of the stored procedure.

Value: Out—The value is passed out from the stored procedure.

Column

Lets you map each parameter of the selected stored procedure, to the target Provisioning Manager LDAP attribute (or an additional runtime context attribute) in this column.

The values in the drop-down menu with asterisks on either side are runtime context values (for example, the Distinguished Name for the account targeted by an operation or operation specific value such as the "new name" in a MODIFY RN rename operation). The remaining values are the Admin account attribute values you mapped in the previous wizard step.

Note: The *ErrorStatus* value can be used to pass a descriptive error string back from a stored procedure and can cause a failure if "strict completion" is true for the binding.

Note: *AddModfiy_AttrsAsXML* can be useful for stored procedures bound to ADD or MODIFY operations where the endpoint has good in-built XML support. The entire request can be passed in a single XML string that can then be broken down within the stored procedure, if desired.

Multivalued

Specifies that the parameter can have multiple values and will need a flattening style.

Flattening Style

Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

Flattening Mode

Displays the flattening mode for multivalued parameters where changes can be expressed in terms of values ADDED or REMOVED, or alternatively, by passing in the complete new list of values (REPLACE).

More Information:

[Stored Procedures Screen \(JDBC Only\)](#) (see page 98)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Account or Group Table Screen

The Map Accounts or Group Table screen lets you select the schema and account or group table, or account and group object class, and then map the selected columns from the specified account or group table, or from the account or group Objectclass, to Provisioning Manager LDAP attributes. Rows in this screen are sorted as follows:

Attributes specified as required by the endpoint appear first in ascending alphabetical order (for JDBC this means not-nullable columns in SQL schema for a table, and for JNDI means attributes listed as "MUST" in an object class' schema definition). The remaining attributes in ascending alphabetical order.

Note: In all mapping tables in Connector Xpress, the leftmost four or five columns display read-only information from the data source. The remaining columns to the right accept mapping information from the user.

This screen contains the following fields:

Schema

(JDBC only) Specifies the schema (for example, table space, or database) name for the tables displayed in the Select Account/Group Table field.

Select Account / Group Objectclass

(JDBC only) Specifies the table that represents account/group information in the selected schema.

Select Account Objectclass

(JNDI only) Specifies the object class or group object class (within the endpoints schema) containing the attribute to which account/group information will be mapped.

Is Required

Specifies that the endpoint requires that this column must have a value.

Column Name / (External Attribute JNDI)

Displays the name of a column or attribute in an Account/Group Table or Account/Group Object class respectively.

SQL Data Type (JDBC only)

Displays the SQL data type for the Account/Group Table Column where the type names displayed may be vendor specific (For example, Oracle "VARCHAR2" or MSSQL "DATETIME").

Size (JDBC only)

Displays the size of the Account/Group Table column in characters.

Note: In some cases this value is a "best guess", such as for NVARCHARS or storage of time-related values in vendor specific formats.

Attribute

Lets you map each column of the selected database/group table (or each attribute of the JNDI object class), to the target Provisioning ServerLDAP attribute in the associated Attribute.

Note: If you select a Generic Attribute, you must also enter a display name in the Display Name column. If you do not want to map a column, do not select an attribute.

Display Name

Specifies the name used as a label for the field as displayed by the Provisioning Manager and must be unique across all mapped attributes.

Note: You must enter a display name for the Generic Attribute type.

Data Type

Displays the data-model type depending on the SQL type for the JDBC case.

Default: String for JNDI. You can change the default if required.

Important! The data model type is very important because it drives data validation and conversion in clients and the Java CS. The following types match syntaxes defined as part of the XML schema (XSD) specification. You can find information on the XML schema definition at the following website:

<http://www.w3.org>

String case sensitive

Specifies that the case of the string is considered

String case insensitive

Specifies that the case of the characters in the string is ignored.

Important! When representing strings it is important to choose the best match from the String Case and String Case Insensitive fields, especially for naming attributes where you need to consider whether you wish “val1” to be treated as equal to “Val1” in lookups.

Boolean

Specifies logically true or false in XML, but is represented by the Provisioning Server and JIAM APIs as 1 or 0 in LDAP attribute values.

Integer

Specifies a 32- bit value between -2147483648 and 2147483647.

Long

Specifies a 64- bit value between 9223372036854775808 and 9223372036854775807.

Double

Specifies a double-precision 64-bit floating point value.

Date

Specifies a date.

Example: 1999-05-31

Note: The Dynamic Namespace plug-in to Provisioning Manager supports the years between 1800 and 9999, but other components of the solution impose no such restrictions and can represent virtually any year in recorded history.

DateTime

Specifies a particular time on a particular day.

Example: 1999-05-31T13:20:00

Note: The Dynamic Namespace plug-in to Provisioning Manager supports the years between 1970 and 2036, so you must use Date to represent days falling outside of this range.

Time

Specifies an offset of between 0 seconds and 23:59:59.

Example: 13:20:00

Float

Specifies a single-precision 32-bit floating point number.

Note: The handling of time related columns is complicated by vendor differences. For example, MSSQL "DATETIME" signifies a DateTime value whereas other vendors use the standard "TIMESTAMP", and MSSQL TIMESTAMPS are actually automatically generated binary values. Also, Oracle does not support a "TIME" type and its "DATE" type is effectively a TIMESTAMP as well. Therefore, to remain vendor-neutral, you are allowed to map to any of Date/DateTime/Time whenever it makes sense for you to map to one of them.

Sync Type

Keeps account/group attributes synchronized with their policies within the Provisioning Server.

Note: Selection of Sync Type is a strong statement. Strict validation is performed on the values of the associated attribute and should only be used where synchronization is required. It implies use of capability attributes within the Provisioning Server.

The valid Sync Type values are:

<blank>

Skips the Provisioning Manager account template synchronization.

Boolean Sync

Specifies a Boolean-account template-synchronization.

Note: Boolean Sync only works on values 1 and 0 at runtime although the values true and false can be entered in the user interface for consistency.

Integer Sync

Specifies an integer-account template-synchronization.

String Sync

Specifies a string-account template-synchronization.

Multivalued column

Specifies that the column can hold multiple values, which means you must choose a flattening style.

Flattening Style

(JDBC only) Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multivalued, you must use the Flattening Style column.

More Information:

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Account or Map Group Confirm Screen

The Map Account or Map Group Confirm screen lets you review and correct, if necessary, the columns that have been created.

The order of attributes in this screen dictate the order of fields displayed in the Provisioning Manager GUI. To reorder the rows, drag and drop the rows to their new location. Click within a read-only column when you start to drag and drop so that you do not accidentally edit the data.

Note: You cannot move the first row, as it is always the naming attribute.

This screen contains the following fields:

Is Required

Specifies that the column is required to have a value. You can only edit this setting if the endpoint does not mark the attribute as *required*. In this case, you can choose to override the endpoint by selecting this column to specify that the attribute is required.

Attributes that are not read only can be altered to is required, or from is required, to not is required.

Column Name (JDBC only)

Displays the name of a database column from the selected Group Table.

Attribute (JDBC Only)

Specifies the name of the Admin Account/Group LDAP attribute that is mapped to the Account/Group Attribute. Only power users that want to override the wizard's recommended attribute names should change this value. For example, to exactly match the attribute name in a previous mapping.

Native Attribute (JNDI only)

Specifies the name of an attribute for the selected Group Objectclass.

Datatype

Displays the type of data you chose in the previous wizard step.

Logical Name

Specifies the logical name used in the mapping and defines the associated Java method name used in JIAM.

These names are checked for uniqueness and for clashes against reserved logical names such as "comments" and "description".

(Optional) String Format

Specifies an additional syntax restriction that can be applied to string formats. You can choose from the following:

DN

Specifies that the string constraint must be a distinguished name.

email

Specifies that the string constraint must be in email format.

Example: xx@xx.xx

quoteless

Specifies that the string constraint must not contain single or double quotes.

Maximum Length

Specifies the maximum length of the attribute values.

Note: In rare cases, for JDBC databases, you may want to set this to be less than the endpoint's maximum for the matching column. For JNDI directories, there is no maximum length defined for the attribute unless:

- The endpoint publishes maximum lengths in its schema, as per Novel Directories, which use *{nnn}* syntax for attribute types.
- You specify one here (As with JDBC, you can also set a length which is stricter than the native length if you want).

Use Default

Specifies whether the value in the Default Value column is to be used as a default by the Provisioning Manager GUI and the Java CS. For example, if no value is provided for the attribute in an ADD request, the default value is implicitly filled in.

Default Value

Specifies the default value that is to be assigned to the attribute when Use Default is selected.

Note: For multi-valued attributes only a single default value can be specified. If this is not sufficient then you will need to use the account template editing functionality in the Provisioning Manager to add additional default values. We recommend that you use this alternative for defaults for attributes which have a string Format of *DN* to ensure the validity of the default value chosen.

Trim Required

Specifies that the leading and trailing white space are stripped from values returned from attributes that are represented as strings on the endpoint.

Note: This is useful primarily for fixed-size CHAR database columns in which shorter strings can be stored. When mapping Date/DateTime or Time attributes to CHAR columns, the Java CS forces trimming regardless of this setting, as validation would fail without it. In all other cases, if this check box is not selected for a CHAR column, it means that it will always need to be referred to by padding values out to the size specified for the column.

Default Account Template (Map Account Only)

Specifies the value (rules string) to be assigned to an attribute in the default account template. Required attributes must have a value in this column, otherwise an account cannot be created based on the default account template.

More Information:

[How You Use the Connector Xpress Wizard](#) (see page 38)

[Maximum Length of LDAP String Used to Represent an Attribute's Value](#) (see page 121)

Maximum Length of LDAP String Used to Represent an Attribute's Value

The Maximum Length field on the [Map Account or Group Confirm](#) (see page 118) screen specifies the maximum length of LDAP string used to represent an attribute's value, including all characters such as optional sign, decimal point and such. Therefore, to represent -32, you need to specify a maximum length of 3. This has no direct relationship to concepts such as significant digits on the native endpoint.

There is currently some asymmetry in the handling of signed attributes with a data type of integer and long. For example, setting a maximum length of 3 for an integer will allow value in the range 999 - 99 inclusive to be entered, whereas ideally the range would be only 99 - 99.

If you map a numeric attribute (integer, long, float, double) to a database column that has a native precision, the native system may change the format of the value provided. In particular, characters may be added and must be taken into account when specifying the maximum length on the Map Account or Map Group Confirm screen.

For example, an attribute with a data type of integer may be mapped to a native floating point column with a precision of 10. If you attempt to store 1234, then the value 1234.00000 is actually stored. This is what will be returned in queries. Therefore, you will need to specify the maximum length for the attribute as 10, even if you only intend to stored integers consisting of four digits.

Also, when you display the property sheet for an object which has this attribute value, you will need to strip off the trailing decimal point and zeros before you will be able to submit, as 1234.000000 is not a syntactically valid integer value.

More information:

[Map Account or Map Group Confirm Screen](#) (see page 118)

Map Group Table Screen

Use this screen to select the schema and group table or group object class to map table columns to managed attributes. This screen is the same as the Map Account Table screen except that groups are targeted rather than accounts.

This screen contains the following fields:

Schema

(JDBC Only) Specifies the schema (for example, table space or database) name for the tables displayed in the Select Account/Group Table field.

Select Account /Group Table

(JDBC Only) Specifies the table that represents account/group information in the selected schema.

Select Account Objectclass

(JNDI Only) Specifies the object class or group object class (within the endpoints schema) containing the attribute to which account/group information will be mapped.

Is Required

Specifies that the endpoint requires that this column must have a value.

Column Name / (Native Attribute JNDI)

Displays the name of a column or attribute in an Account/Group Table or Account/Group Object class respectively.

SQL Data Type

(JDBC Only) Displays the SQL data type for the Account/Group Table Column where the type names displayed may be vendor specific (For example, Oracle "VARCHAR2" or MSSQL "DATETIME").

Size

(JDBC Only) Displays the size of the Account/Group Table column in characters.

Note: In some cases this value is a "best guess", such as for NVARCHARS or storage of time-related values in vendor specific formats.

Attribute

Lets you map each column of the selected database/group table (or each attribute of the JNDI object class), to the target Provisioning Manager LDAP attribute in the associated Attribute column.

Note: If a Generic Attribute is selected, you must also enter a display name in the Display Name column. If you do not want to map a column, do not select an attribute.

Display Name

Specifies the name used as a label for the field as displayed by the Provisioning Manager and must be unique across all mapped attributes.

Note: A display name must be entered for the Generic Attribute type, Provisioning Attribute only.

Data Type

Displays the data-model type depending on the SQL type for the JDBC case.

Default: String for JNDI. You can change the default if required.

Important! The choices you make in the Data Type column are very important as they drive data validation and conversion in clients and the Java CS. The following types match syntaxes defined as part of the XML schema (XSD) specification. You can find information on the XML schema definition at the following website:

<http://www.w3.org>

String case sensitive

Specifies that the case of the string is considered

String case insensitive

Specifies that the case of the characters in the string is ignored.

Important! When representing strings it is important to choose the best match from the String Case and String Case Insensitive fields, especially for naming attributes where you need to consider whether you wish “val1” to be treated as equal to “Val1” in lookups.

Boolean

Specifies logically true or false in XML but which is represented by the Provisioning Server and JIAM APIs as 1 or 0 in LDAP attribute values. In graphical clients should be rendered as a check box.

Integer

Specifies a 32 bit value between -2147483648 and 2147483647.

Long

Specifies a 64 bit value between 9223372036854775808 and 9223372036854775807.

Double

Specifies a double-precision 64-bit floating point value.

Date

Specifies a date.

Example: 1999-05-31

Note: The Dynamic Namespace plug-in to Provisioning Manager supports the years between 1800 and 9999, other components of the solution impose no such restrictions and can represent virtually any year in recorded history.

DateTime

Specifies a particular time on a particular day.

Example: 1999-05-31T13:20:00

Note: The Dynamic Namespace plug-in to Provisioning Manager supports the years between 1970 and 2036, so you must use date to represent days falling outside of this range.

Time

Specifies an offset of between 0 seconds and 23:59:59.

Example: 13:20:00

Float

Specifies a single-precision 32-bit floating point number.

Note: The handling of time related columns is complicated by vendor differences. For example, MSSQL "DATETIME" signifies a DateTime value whereas other vendors use the standard "TIMESTAMP", and MSSQL TIMESTAMPS are actually automatically generated binary values. Also Oracle does not support a "TIME" type and its "DATE" type is effectively a TIMESTAMP as well. Therefore, to remain vendor-neutral, you are allowed to map to any of Date/DateTime/Time whenever it makes sense for you to map to one of them.

Sync Type

Keeps keep account/group attributes synchronized with their policies within the Provisioning Server (see the Provisioning Manager documentation). The valid Sync Type values are:

<blank>

Skips Provisioning Manager account template synchronization.

Boolean Sync

Specifies a Boolean-account template-synchronization.

Note: Boolean Sync only works on values 1 and 0 at runtime although the values true and false can be entered in the user interface for consistency.

Integer Sync

Specifies an integer-account template-synchronization.

String Sync

Specifies a string-account template-synchronization.

Note: Selection of Sync Type is a strong statement. Strict validation is performed on the values of the associated attribute and should only be used where synchronization is required. It implies use of "capability" attributes within Provisioning Manager.

Multivalued

Specifies that the column can hold multiple values, mandating a flattening style be chosen.

Flattening Style

(JDBC Only) Specifies a flattening method used to represent multiple values in a single string literal. If the attribute is multi-valued, you must use the Flattening Style column.

More Information

[Map Account or Group Table Screen](#) (see page 115)

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Membership Table Screen (JDBC Only)

The Map Membership Table screen lets you select a separate lookup table for associations between account and group objects and map the relevant attributes.

We recommended that you use foreign keys in the membership table to relate to the account table and group table, and then map the foreign keys in the membership table to account attribute and group attribute. This will eliminate the possibility of a mismatch of the SQL data type/size between the membership table column and the associated account/group column.

Schema

Specifies the schema (for example, table space or database) name for which the tables will be displayed in the Membership Table field.

Membership Table

Specifies the tables that define the members of each group.

Note: The database table that you select needs to have at least two columns and can not have more than two required columns. You must map all the required columns to provisioning attribute.

Is Required

Specifies that the column is required to have a value.

Column Name

Displays the name of a column or attribute in a Membership Table.

SQL Data Type

Displays the SQL data type for the Membership Table column.

Size

Displays the size of the Membership Table column.

Attribute

Lets you assign the name of the account attribute you want to use in the association and the other to the name of the group attribute. Normally, the account and group name is used to define the association, but power users may want to select alternate keys (for example, an account's compact user-id attribute can be used instead of the account name). If you do not want to map a column, do not select an attribute.

More Information:

[How You Use the Connector Xpress Wizard](#) (see page 38)

Map Containers Screen (JNDI Only)

The Map Containers screen lets you specify the object classes which you want to use as containers in your connector, and their naming attributes.

This screen contains the following fields:

Container Objectclass

Specifies a candidate object class. To add to the list of containers, click Add.

Naming Attribute

Specifies the naming attribute for the object class selected in the Container Objectclass field.

Note: The list on the right displays the currently selected container object classes and their naming attributes. To add or remove items from the list, use the Add and Remove buttons.

More Information:

[How You Use the Connector Xpress Wizard](#) (see page 38)

Chapter 9: Examples

This section contains the following topics:

[How Acme Corporation Promotes a Connector from a Test to a Production Environment](#) (see page 127)

How Acme Corporation Promotes a Connector from a Test to a Production Environment

Acme Corp has completed creating and testing a new Oracle connector that maps Oracle database tables to provisioning accounts and groups. Acme has deployed the connector to the Acme test/staging provisioning server, Acme_Test_PS. Acme Corp now wants to deploy the connector to a new production provisioning server, Acme_Prod_PS. After deploying the connector, Acme Corp wants to remove the connector from the test environment.

The following example shows you the process the Administrator of Acme Corp's Identity Manager installation follows to promote the connector from the test environment to the new production environment. The Administrator:

- Saves the Oracle connector as a Connector Xpress project. For example, ORA_Connector.con.
- Configures the new production provisioning server, Acme_Prod_PS.
This specifies the connection details of the Acme_Prod_PS production provisioning server to which the Acme Corp wants to deploy the connector.
- Opens the Connector Xpress project ORA_Connector.con.
- Creates a new endpoint type on the Acme_Prod_PS provisioning server.
- Specifies the managing connector server that will manage the new endpoint type.
This ensures that requests for the endpoint type are routed to it.
- Explores and correlates the new endpoint type.
This populates the Acme_Prod_PS provisioning server with data found in the endpoint.
- Deletes the old endpoint type from the test provisioning server, Acme_Test_PS.
This deletes the endpoint type, the associated account templates, and any child endpoints, however data on the endpoint is unchanged.

More Information:

[Projects](#) (see page 35)

[Add and Configure a Provisioning Server](#) (see page 24)

[Create a Connector Server Configuration](#) (see page 26)

[Create a New Endpoint Type](#) (see page 82)

[Acquire, Explore and Correlate an Endpoint](#) (see page 81)

[Clean an Endpoint Type](#) (see page 83)