NetIQ® Identity Manager
Driver for Delimited Text Implementation Guide

October 2014

Contents

About this Book and the Library

The *Identity Manager Driver for Delimited Text Implementation Guide* explains how to install and configure the Identity Manager Driver for Delimited Text.

Intended Audience

This guide is intended for eDirectory and Identity Manager administrators who are using the Delimited Text driver.

Other Information in the Library

The library provides the following information resources:

Identity Manager Setup Guide

Provides overview of Identity Manager and its components. This book also provides detailed planning and installation information for Identity Manager.

Designer Administration Guide

Provides information about designing, testing, documenting, and deploying Identity Manager solutions in a highly productive environment.

User Application: Administration Guide

Describes how to administer the Identity Manager User Application.

User Application: User Guide

Describes the user interface of the Identity Manager User Application and how you can use the features it offers, including identity self-service, the Work Dashboard, role and resource management, and compliance management.

User Application: Design Guide

Describes how to use the Designer to create User Application components, including how to work with the Provisioning view, the directory abstraction layer editor, the provisioning request definition editor, the provisioning team editor, and the role catalog.

Identity Reporting Module Guide

Describes the Identity Reporting Module for Identity Manager and how you can use the features it offers, including the Reporting Module user interface and custom report definitions, as well as providing installation instructions.

Analyzer Administration Guide

Describes how to administer Analyzer for Identity Manager.

Identity Manager Common Driver Administration Guide

Provides information about administration tasks that are common to all Identity Manager drivers.

Identity Manager Driver Guides

Provides implementation information about Identity Manager drivers.

About NetIQ Corporation

We are a global, enterprise software company, with a focus on the three persistent challenges in your environment: Change, complexity and risk—and how we can help you control them.

Our Viewpoint

Adapting to change and managing complexity and risk are nothing new

In fact, of all the challenges you face, these are perhaps the most prominent variables that deny you the control you need to securely measure, monitor, and manage your physical, virtual, and cloud computing environments.

Enabling critical business services, better and faster

We believe that providing as much control as possible to IT organizations is the only way to enable timelier and cost effective delivery of services. Persistent pressures like change and complexity will only continue to increase as organizations continue to change and the technologies needed to manage them become inherently more complex.

Our Philosophy

Selling intelligent solutions, not just software

In order to provide reliable control, we first make sure we understand the real-world scenarios in which IT organizations like yours operate—day in and day out. That's the only way we can develop practical, intelligent IT solutions that successfully yield proven, measurable results. And that's so much more rewarding than simply selling software.

Driving your success is our passion

We place your success at the heart of how we do business. From product inception to deployment, we understand that you need IT solutions that work well and integrate seamlessly with your existing investments; you need ongoing support and training post-deployment; and you need someone that is truly easy to work with—for a change. Ultimately, when you succeed, we all succeed.

Our Solutions

- Identity & Access Governance
- Access Management
- Security Management
- Systems & Application Management
- Workload Management
- Service Management

Contacting Sales Support

For questions about products, pricing, and capabilities, contact your local partner. If you cannot contact your partner, contact our Sales Support team.

Worldwide:	www.netiq.com/about_netiq/officelocations.asp
United States and Canada:	1-888-323-6768
Email:	info@netiq.com
Web Site:	www.netiq.com

Contacting Technical Support

For specific product issues, contact our Technical Support team.

Worldwide:	www.netiq.com/support/contactinfo.asp
North and South America:	1-713-418-5555
Europe, Middle East, and Africa:	+353 (0) 91-782 677
Email:	support@netiq.com
Web Site:	www.netiq.com/support

Contacting Documentation Support

Our goal is to provide documentation that meets your needs. The documentation for this product is available on the NetIQ Web site in HTML and PDF formats on a page that does not require you to log in. If you have suggestions for documentation improvements, click **Add Comment** at the bottom of any page in the HTML version of the documentation posted at www.netiq.com/documentation. You can also email Documentation-Feedback@netiq.com. We value your input and look forward to hearing from you.

Contacting the Online User Community

NetIQ Communities, the NetIQ online community, is a collaborative network connecting you to your peers and NetIQ experts. By providing more immediate information, useful links to helpful resources, and access to NetIQ experts, NetIQ Communities helps ensure you are mastering the knowledge you need to realize the full potential of IT investments upon which you rely. For more information, visit community.netiq.com.

1 Understanding the Delimited Text Driver

The Delimited Text driver lets you use delimited text files (CSV) to synchronize data between the Identity Vault and applications. Essentially, you can transfer data from the Identity Vault to any system that can consume delimited text files, or transfer data to the Identity Vault from any system that can generate delimited text files.

The Delimited Text driver can also be modified to consume an XML files. The following sections provide information to help you understand the Delimited Text driver.

- Section 1.1, "How the Delimited Text Driver Works," on page 9
- Section 1.2, "Java Interfaces to the Driver," on page 12
- Section 1.3, "Key Driver Features," on page 12

1.1 How the Delimited Text Driver Works

The Delimited Text driver uses the Publisher channel, the Subscriber channel, and policies to control data flow between the Identity Vault and the delimited text files, as explained in the following sections:

- Section 1.1.1, "Publisher and Subscriber Channels," on page 9
- Section 1.1.2, "Policies," on page 10
- Section 1.1.3, "Supported File Types," on page 11

1.1.1 Publisher and Subscriber Channels

The Delimited Text driver provides data flow along the Publisher and Subscriber channels as shown in Figure 1-1.

Figure 1-1 *Data Flow*
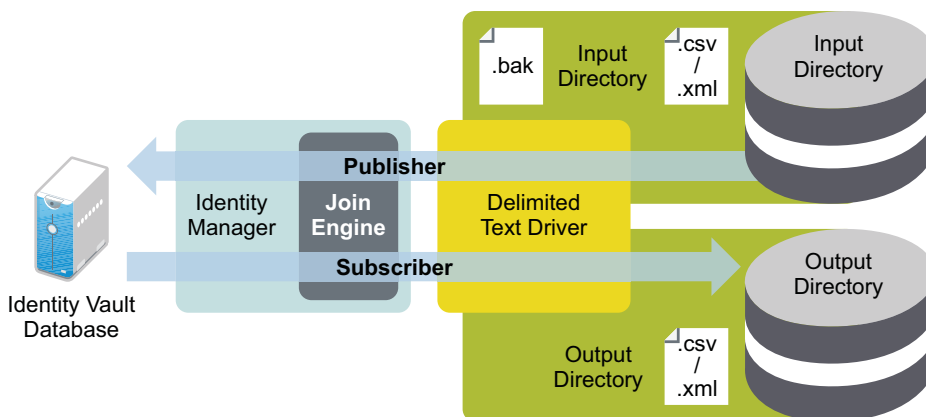
The example configuration that ships with this driver includes both Subscriber and Publisher channels. However, in many configurations, only one-way data flow is required. In those configurations, only a Publisher or Subscriber channel is used. The other channel is disabled. For more information, see Chapter 5, "Setting Up One-Way Synchronization," on page 29.

Publisher Channel

The Publisher channel reads information from input text files on your local file system and submits that information to the Identity Vault via the Identity Manager engine.

By default, the Publisher channel does the following:

- Checks the input directory every 10 seconds.
- Processes any files that have a `.csv` extension.
- Changes `.csv` extensions of processed files to `.bak`.
- Cycles through this process until you stop the driver.

Subscriber Channel

The Subscriber channel watches for additions and modifications to Identity Vault objects and creates output files on your local file system that reflect those changes.

By default, the Subscriber channel keeps an output file open until either 200 transactions have been logged or 30 seconds have elapsed. When either of these thresholds is reached, the output file is saved with a `number.csv` filename and a new output file is opened.

1.1.2 Policies

Policies control data synchronization between the driver and the Identity Vault. The following table provides information on the set of preconfigured policies that come with the Delimited Text driver. For information about modifying policies, see *NetIQ Identity Manager Policies in iManager Guide* or *NetIQ Identity Manager Policies in Designer*.

Table 1-1   Preconfigured Policies

Policy	Description
Schema Map	Configured on the driver object. Maps Identity Vault User properties to application attributes as follows: Surname > LastName Given Name > FirstName Title > Title Internet EMail Address > Email Telephone Number > WorkPhone Facsimile Telephone Number > Fax mobile > WirelessPhone Description > Description The application attributes correspond to the sequence of values in the file or, if present, to the attributes associated with unnamed XDS `<field>` elements.

Policy	Description
Input Transform	Configured on the driver object.
	If the input document is an XML document, no transformations are made. If the document is a delimited text file, each record is transformed into an XDS Add element for User objects with attributes defined by the schema map.
	Associations are defined by the value of user's e-mail attribute.
Output Transform	Configured on the driver object.
	Specifies that a comma is used as the delimiter character for output files and that the file format is comma-separated value (CSV) format. It transforms the XDS document into CSV format and submits it to the driver shim.
Create	Configured on the Publisher channel.
	Specifies that in order for a User to be created in an Identity Vault, the Given Name and Internet EMail Address attributes must be defined.
	The user CN is formed by concatenating the values of first name and last name.
Matching	Configured on the Publisher channel.
	Specifies that a user in an Identity Vault is the same user specified in the input file when the value of Internet Email Address is the same in both places.
	If there is a match, only changed attributes are updated in the Identity Vault.
Placement	Configured on the Publisher channel.
	Specifies that a new user is placed in the Users or Active container and named with the CN created by the Input Transform rule.
	You need to create a Users\Active container at the root of your tree before you start the driver.
Event Transform	Configured on the Subscriber channel.
	If an Identity Vault reports a Modify or Sync event, those events are changed to an instance element that can be used to create a complete output record.

NOTE: The SYN_INTEGER64 attribute type is supported on both Subscriber and Publisher channels. Use this attribute type to represent the dates after 2037.

1.1.3 Supported File Types

The driver currently supports two types of files:

- "Comma-Separated Values Files" on page 12
- "XML Files in XDS Format" on page 12

Comma-Separated Values Files

Comma-separated value (CSV) files are text files that contain data divided into fields and records. Fields are delimited by commas, and records are delimited by a hard return.

If you need a comma or hard return within the value of a particular field, the entire field value should be enclosed in quotes.

Because the meaning of each field in a CSV file is derived from its position, each record in a CSV file should have the same number of fields. Field values can be left blank, but each record should have the same number of delimiter characters.

XML Files in XDS Format

The XDS format is the defined NetIQ subset of possible XML formats. This is the initial format for data coming from an Identity Vault. By modifying default rules and changing the style sheets, the Delimited Text driver can be configured to work with any XML format.

For detailed information on the XDS format, refer to NDS DTD Commands and Events (http:// developer.novell.com/ndk/doc/dirxml/index.html?dirxmlbk/data/a5323rs.html).

For information on configuring the driver to use XML files in the XDS format, see Chapter 6, "Configuring for XDS XML Files," on page 31.

1.2   Java Interfaces to the Driver

The Delimited Text driver includes four Java interfaces that enable you to add extensions, which are optional. These enhancements to the driver require Java programming. For more information, see Chapter 8, "Using Java Interfaces to Customize File Processing," on page 35.

1.3   Key Driver Features

The sections below contains information about the key driver features.

1.3.1   Local and Remote Platforms

The Delimited Text driver runs on the Identity Manager server or uses the Remote Loader to run on another server.

1.3.2   Entitlements

The Delimited Text driver supports custom and administrator-defined entitlements. The driver content includes a new entitlement package that allows Identity Manager to consume CSV files containing target system permission information, dynamically create an entitlement and dynamic resource for each permission type, and load the permission data as entitlement values into Identity Manager Role-

Based Provisioning Module. The driver can then automatically provision or deprovision these resources to Identity Manager identities based on attribute values consumed during standard Publication channel processing.

The following packages contain the content necessary for Permission Collection and Reconciliation service:

- NOVLDTXTBASE_2.2.0 (Base Package)
- NOVLDTXTENT_2.0.0 (Entitlements Package)

If you want the driver to support custom and administrator-defined entitlements, ensure that these packages are installed on the driver. You can turn Permission Collection and Reconciliation service on or off using GCVs provided with this service.

Prerequisites

Before continuing, ensure that you go through the prerequisites needed for enabling this service. For general prerequisites, see "Prerequisites" in "Understanding Permission Collection and Reconciliation Service " in the *NetIQ Identity Manager Driver Administration Guide*. In addition to the general prerequisites, ensure that the Delimited Text driver version is 4.0.0.1.

Also, you need to set up administrative user accounts and configure a password policy for them. For more information, see "Setting Up Administrative User Accounts" and "Setting Up Administrative Passwords" in the *NetIQ Identity Manager Driver Administration Guide*.

To use the Permission Collection and Reconciliation service included in the Delimited Text driver, you can either create a new driver with the latest packages or upgrade packages on an existing driver. For more information about creating a driver, see Section 3.2, "Creating the Driver Object in Designer," on page 17 or Section 3.4, "Adding Packages to an Existing Driver," on page 24.

CSV File Format

The Delimited Text driver can consume the entitlement information from the CSV file extracted from a target system. The CSV file must contain values for the target system permission information in a correct format so that the driver reads them correctly from the file. A separate CSV file must be maintained for every custom entitlement. For example, a CSV file that holds **ParkingPass** entitlement details represents this information in the following format:

```
North, North Lot, North Parking Lot
```

where **North** is the entitlement value, **North Lot** is the display name in the User Application for the entitlement value **North**, and **North Parking Lot** is the description for the entitlement value. This description is displayed in the User Application.

The CSV file must be located on the same server as the driver. This file contains the values for application entitlements.

1.3.3 Password Synchronization

The Delimited Text driver has policies to handle password synchronization. However, no automatic password synchronization exists for the Delimited Text driver. You must decide which attribute you want to hold the password and then synchronize that attribute. Any password synchronization for the driver is just an attribute synchronization.

2 Installing Driver Files

By default, the Delimited Text driver files are installed on the Identity Manager server at the same time as the Identity Manager engine. The installation program extends the Identity Vault's schema and installs both the driver shim and the driver configuration files. It does not create the driver in the Identity Vault (see Chapter 3, "Creating a New Driver Object," on page 17) or upgrade an existing driver's configuration (see Chapter 4, "Upgrading an Existing Driver," on page 27).

Unless you extend the driver's functionality along with the Java interfaces (see Chapter 8, "Using Java Interfaces to Customize File Processing," on page 35), the driver is only capable of reading input text files from the local file system of the server where the driver is running. This means that your input directory must be located on the same server as the driver. You have three options:

- If your input directory can be located on the Identity Manager server and the Delimited Text driver files are already installed on the server, skip this section and continue with Chapter 3, "Creating a New Driver Object," on page 17.

- If your input directory can be located on the Identity Manager server but the driver files are not already installed on the server, install the files by using the instructions in "Preparing to Install the Engine, Drivers, and Plug-ins" in the *NetIQ Identity Manager Setup Guide*.

- If your input directory cannot be located on the Identity Manager server, install the Remote Loader (required to run the driver on a non-Identity Manager server) and the driver files on the server where the input directory resides. See "Preparing to Install the Engine, Drivers, and Plug-ins" in the *NetIQ Identity Manager Setup Guide*.

3 Creating a New Driver Object

You can create the driver object in the Identity Vault by installing the driver packages and then modifying the driver configuration to suit your environment. The following sections provide instructions:

3.1 Preparing Data Locations

You need to make sure that the driver's input and output directories exist. The input directory is where the driver looks for files to be processed into the Identity Vault. The output directory is where the driver places files to be processed by the target application.

The input and output directories must be located on the same server as the driver. The directories can have any names supported by the server operating system.

3.2 Creating the Driver Object in Designer

To create a Delimited Text driver object, install the driver packages and then modify the configuration to suit your environment. After you create and configure the driver object, you need to deploy it to the Identity Vault and start it.

3.2.1 Importing the Current Driver Packages

The driver packages contain the items required to create a driver, such as policies, entitlements, filters, and Schema Mapping policies. These packages are only available in Designer. You can upgrade any package that is installed if there is a newer version of the package available. It is recommended to have the latest packages in the Package Catalog before creating a new driver

object. Designer prompts you for importing the required packages when it creates the driver object. For more information on upgrading packages, see "Upgrading Installed Packages" in the *NetIQ Designer for Identity Manager Administration Guide*.

To verify you have the latest packages in the Package Catalog:

1 Open Designer.

2 In the toolbar, click **Help** > **Check for Package Updates**.

3 Click **OK** if there are no package updates

or

Click **OK** to import the package updates.

4 In the Outline view, right-click the Package Catalog.

5 Click **Import Package**.

6 Select the Delimited Text packages.

or

Click **Select All** to import all of the packages displayed, then click **OK**.

By default, only the base packages are displayed. Deselect **Show Base Packages Only** to display all packages.

IMPORTANT: If you want the driver to support the Permission Collection and Reconciliation Service functionality, ensure you import the following packages to the driver:

- ◆ NOVLDTXTBASE_2.2.0 (Base Package)
- ◆ NOVLDTXTENT_2.0.0 (Entitlements Package)
- ◆ NOVLACOMSET_2.0.0 (Common Settings Advanced Edition Package)

For information about the Permission Collection and Reconciliation service, see "Understanding Permission Collection and Reconciliation Service " in the *NetIQ Identity Manager Driver Administration Guide*.

---

7 Click **OK** to import the selected packages, then click **OK** in the successfully imported packages message.

8 After the current packages are imported, continue with Section 3.2.2, "Installing the Driver Packages," on page 18.

3.2.2 Installing the Driver Packages

To install the driver packages:

1 Start Designer and open your project.

2 In the Modeler, right-click the driver set where you want to create your new driver, then select **New** > **Driver**.

3 Select **Delimited Text Base** from the list of base packages, then click **Next**.

4 Select the optional features to install for the Delimited Text driver.

All options are selected by default. The options are:

- ◆ **Delimited Text Entitlements:** This package contains policies for quick onboarding of custom entitlements and dynamic resource creation. This package also contains GCVs to control the resource mapping. Select this package if you want to enable the entitlement

onboarding feature for this driver. For more information, see "Understanding Permission Collection and Reconciliation Service " in the *NetIQ Identity Manager Driver Administration Guide*.

NOTE: If you are enabling quick onboarding of custom entitlements functionality, ensure that you upgrade the Managed System Gateway driver version to 4.0.0.6.

- ◆ **Delimited Text Password Synchronization:** This package contains GCVs and sample policies for synchronizing passwords.
- ◆ **Delimited Text Managed System Information:** This package contains the policies that enable the driver to collect data for reports.

Note that if the Delimited Text Managed System Information and Password Synchronization packages are not imported into the Package Catalog, only the package ID is displayed for those packages in the list of optional features.

5 Click **Next**.

6 (Optional) If you want the driver to synchronize passwords, select the Delimited Text Password Synchronization package, then click **Next**.

7 (Conditional) If not already configured, fill in the following fields on the Common Settings page, then click **Next**:

- ◆ **User Container:** Select the Identity Vault container where users are added if they don't already exist in the Identity Vault. This value becomes the default value for all drivers in the driver set.
- ◆ **Group Container:** Select the Identity Vault container where groups are added if they don't already exist in the Identity Vault. This value becomes the default value for all drivers in the driver set.

NOTE: The Common Settings page is only displayed if the Common Settings package is a dependency.

8 (Conditional) If not already configured, fill in the following fields on the Common Settings Advanced Edition page, then click **Next**:

- ◆ **User Application Provisioning Services URL:** Specify the User Application Identity Manager Provisioning URL.
- ◆ **User Application Provisioning Services Administrator:** Specify the DN of the User Application Administrator user. This user should have the rights for creating and assigning resources. For more information, see "Setting Up Administrative User Accounts" in the *NetIQ Identity Manager Driver Administration Guide*.

NOTE: This page is only displayed if you installed the Common Settings Advanced Edition package.

9 On the Install Delimited Text Base page, specify a name for the driver, then click **Next**.

10 Fill in the following fields, then click **Next**:

- ◆ **Input File Path:** Specify the path for the input file.
- ◆ **Output File Path:** Specify the path for the output file.

11 (Conditional) If you want to use the Remote Loader with this driver, fill in the following fields to configure the driver to connect through the Remote Loader, then click **Next**. Otherwise, click **No**, then click **Next**.

* **Connect to Remote Loader:** By default, the driver is configured to connect through the Remote Loader. If you want to run the driver locally, select **no**, then click **Next**. Otherwise, fill in the remaining fields to configure the driver to connect through the Remote Loader.

* **Host Name:** Specify the host name or IP address of the server where the driver's Remote Loader service is running.

* **Port:** Specify the port number where the Remote Loader is installed and is running for this driver. The default port number is 8090.

* **Remote Password:** Specify the Remote Loader's password, as defined on the Remote Loader. The Identity Manager server or the Remote Loader shim requires this password to authenticate to the Remote Loader

* **Driver Password:** Specify the driver object password that is defined in the Remote Loader service. The Remote Loader requires this password to authenticate to the Identity Manager server.

12 On the Driver Parameters page, fill in the following fields, then click **Next**:

* **Field Delimiter:** Specify the character to use to delimit field values in the input and output files. It must be one character. You can also use the tab as the delimiter field value. Tab is represented as {tab}. The default is a comma.

* **Field Names:** Specify a comma-separated list of attribute names that can be referred to in the Schema Mapping rule. The fields of the records included in your input CSV files must correspond to the order and positioning of the names in this list.

13 (Conditional) On the Entitlements Name to CSV File Mappings page, click the **Add Name to File Mapping** + icon to populate the page with the entitlement configuration options.

NOTE: This page is only displayed if you installed the Entitlements package.

The information that you specify in this page is used for creating the permission catalog. Fill in the following fields:

* **Entitlement Name:** Specify a descriptive name for the entitlement to map it to the CSV file that contains the target system entitlement details.

  Entitlement Name is the name of the entitlement. This parameter corresponds to the Entitlement Assignment Attribute on the target system. For example, you could define an entitlement called **ParkingPass**.

  This parameter is used to create a resource in the User Application.

* **Entitlement Assignment Attribute:** Specify a descriptive name for the assignment attribute for an entitlement.

  Entitlement Assignment Attribute holds the entitlement values on the target system. For example, you could have an attribute called **Parking**.

  You must add this parameter to **Field Names** in the Driver Parameters page or modify it in driver settings after creating the driver.

* **CSV File:** Specify the location of the CSV file. This file must be located on the same server as the driver. This file contains the values for application entitlements.

* **Multi-valued?:** Set the value of this parameter to **True** if you want to assign resources and entitlements multiple times with different values to the same user. Otherwise, set it to **False**.

NOTE: After creating the driver, you can modify `Entitlement Name to CSV File Mapping` from `PermissionNameToFile` mapping.

14 (Conditional) On the Install Delimited Text Managed System Information page, fill in the following fields to define your Delimited Text system, then click **Next**:

NOTE: This page is only displayed if you installed the Managed System Information package.

- ◆ **Name:** Specify a descriptive name for this Delimited Text system.
- ◆ **Description:** Specify a brief description for this Delimited Text system.
- ◆ **Location:** Specify the physical location of this Delimited Text system.
- ◆ **Vendor:** Leave the setting unchanged.
- ◆ **Version:** Specify the version of this Delimited Text system.

15 (Conditional) On the Install Delimited Text Managed System Information page, fill in the following fields to define your Delimited Text system, then click **Next:**

NOTE: This page is only displayed if you selected to install the Managed System Information package.

- ◆ **Business Owner:** Select a user object in the Identity Vault that is the business owner of the Delimited Text system. This can only be a user object, not a role, group, or container.
- ◆ **Application Owner:** Select a user object in the Identity Vault that is the application owner of the Delimited Text system. This can only be a user object, not a role, group, or container.

16 (Conditional) On the Install Delimited Text Managed System Information page, fill in the following fields to define your Delimited Text system, then click **Next**:

- ◆ **Classification:** Select the classification of the Delimited Text system. This information is displayed in the reports. Your options are:
  - ◆ Mission-Critical
  - ◆ Vital
  - ◆ Not-Critical
  - ◆ Other

    If you select **Other**, you must specify a custom classification for the Delimited Text driver system.
- ◆ **Environment:** Select the type of environment the Delimited Text system provides. The options are:
  - ◆ Development
  - ◆ Test
  - ◆ Staging
  - ◆ Production
  - ◆ Other

    If you select **Other**, you must specify a custom classification for the Delimited Text driver system.

17 Review the settings and click **Finish** to create the driver.

18 Modify the driver settings. Proceed to Section 3.2.3, "Modifying the Driver Settings," on page 22.

3.2.3 Modifying the Driver Settings

There are many settings that can help you customize and optimize the driver. The settings are divided into categories such as Driver Configuration, Engine Control Values, and Global Configuration Values (GCVs). Although it is important for you to understand all of the settings, your first priority should be to review the Driver Parameters located on the Driver Configuration page. These settings let you control the format and content of the input and output files.

The driver configuration settings are explained in Appendix A, "Driver Properties," on page 43.

If you do not have the Driver Properties page displayed in Designer:

1 Open your project.

2 In the Modeler, right-click the driver icon ![icon] or the driver line, then select **Properties**.

3 (Conditional) Click **GCVs > Entitlements** and review the following settings:

NOTE: These settings are only displayed if you installed the Entitlements package.

- ◆ **Enable Permission Collection and Reconciliation:** Select the value of this parameter to **True** for allowing permission collection and entitlement assignment. By default it is set to **False**, which allows the driver to override any other conditions to reconcile custom entitlements.

- ◆ **Enable Permission Reconciliation for all Custom Entitlements:** If the value of this parameter is set to **No**, it allows you to select specific custom entitlements for reconciling them. By default, it is set to **Yes**, which allows reconciling of all custom entitlements.

- ◆ **Add Custom Entitlements for Reconciliation:** This parameter is presented if the value of **Enable Permission Reconciliation for all Custom Entitlements** is set to **No**.

  Click the **Add** ![icon] icon to add custom entitlements you want to selectively onboard and specify **Assignment Attribute Name** for them.

NOTE: Ensure that **Entitlement Assignment Attributes** values are added to the **Field Names** parameter in the driver configuration if they are not added initially during driver creation.

---

4 Click **OK** when finished.

5 Deploy the driver to the Identity Vault. Proceed to Section 3.2.4, "Deploying the Driver Object," on page 22.

3.2.4 Deploying the Driver Object

To deploy the driver into the Identity Vault,

1 In the Modeler, right-click the driver icon ![icon] or the driver line, then select **Live > Deploy**.

2 If you are authenticated to the Identity Vault, skip to Step 3, otherwise, specify the following information, then click **OK**.:

- ◆ **Host:** Specify the IP address or DNS name of the server hosting the Identity Vault.

- ◆ **Username:** Specify the DN of the user object used to authenticate to the Identity Vault.

- ◆ **Password:** Specify the user's password.

3 Read through the deployment summary, then click **Deploy**.

4 Read the successful message, then click **OK**.

5 Click **Define Security Equivalence** to assign rights to the driver.

The driver requires rights to objects within the Identity Vault and to the input and output directories on the server. The Admin user object is most often used to supply these rights. However, you might want to create a DriversUser, for example, and assign security equivalence to that user. For more information about defining a Security Equivalent User in objects for drivers in the Identity Vault, see "Establishing a Security Equivalent User" in the Identity Manager Security Guide.

For receiving events from the Identity Vault, ensure that the driver's Security Equals DN has the following rights in the Identity Vault:

- **Entry:** Browse rights.
- **Attributes:** Read rights.

5a Click **Add**, browse to and select the object with the correct rights.

5b Click **OK** twice.

6 Click **Exclude Administrative Roles** to exclude users that should not be synchronized.

You should exclude any administrative User objects (for example, Admin and DriversUser) from synchronization.

6a Click **Add**, browse to and select the user object you want to exclude, then click **OK**.

6b Repeat Step 6a for each object you want to exclude, then click **OK**.

7 Click **OK**.

After you have customized the driver for your environment, you must deploy it to the Identity Vault. Proceed to Section 3.2.4, "Deploying the Driver Object," on page 22.

3.2.5 Starting the Driver Object

If you configured the driver with the Permission Collection and Reconciliation service, ensure the driver meets the following requirements before it is started for the first time:

- The Entitlement value CSV files are available in the locations specified during driver configuration. You can check the location you specified by examining the **PermissionNameToFile** mapping table under the driver in the Outline View of Designer.
- The driver administrator and the User Application Resource Administrator are added to a Password Policy.

To start the driver, in the Modeler, right-click the driver icon or the driver line, then select **Live > Start Driver**.

For information about management tasks with the driver, see Chapter 9, "Managing the Driver," on page 39.

3.3 Activating the Driver Object

If you create the Delimited Text driver in a driver set where you have already activated a driver that comes with the Integration Module for Tools, the driver inherits the activation. If you created the Delimited Text driver in a driver set that has not been activated, you must activate the driver within 90 days by using the Integration Module for Tools activation. Otherwise, the driver stops working.

The drivers that are included in the Integration Module for Tools are:

- Driver for Delimited Text
- Driver for SOAP

For information on activation, refer to "Activating Identity Manager" in the *NetIQ Identity Manager Setup Guide*.

If driver activation has expired, the following error message is displayed in the ndstrace window:

```
DirXML Log Event -------------------
     Driver:   \META-RHEL6\system\DriverSet\eDirDriver-BulkOperations
     Channel:  Subscriber
     Status:   Error
     Message:  Code(-9075) Shutting down because DirXML engine evaluation period
has expired. Activation is required for further use.
```

To use the driver, you must reactivate it.

3.4 Adding Packages to an Existing Driver

You can add new functionality to an existing driver by adding new packages to it.

1 Right-click the driver, then click **Properties**.

2 Click **Packages**, then upgrade the already installed Delimited Text Base package.

    2a Select the package from the list of packages, then click the **Select Operation** cell.

    2b Click **Upgrade** from the drop-down list, then click **Apply**.

    2c Click **OK** to close the Package Management page.

    You can upgrade the Password Synchronization package in a similar way.

3 Click the **Add Packages** icon ➕.

4 Select the packages to install.

> **NOTE:** The Delimited Text Entitlements package contains the content for Permission Collection and Reconciliation service. Select it to enable this service.

5 (Optional) If you want to see all available packages for the driver, clear the **Show only applicable package versions** option, if you want to see all available packages for the driver, then click **OK**.

This option is only displayed on drivers. By default, only the packages that can be installed on the selected driver are displayed.

6 Click **Apply** to install all of the packages listed with the Install operation.

7 (Conditional) Fill in the fields with appropriate information to install the package you selected for the driver, then click **Next**.

8 Read the summary of the installation, then click **Finish**.

9 Click **OK** to close the Package Management page after you have reviewed the installed packages.

10 Modify the driver configuration settings. See Section 3.2.3, "Modifying the Driver Settings," on page 22.

11 Deploy the driver. See Section 3.2.4, "Deploying the Driver Object," on page 22.

12 Start the driver. See Section 3.2.5, "Starting the Driver Object," on page 23.

13 (Conditional) Review the newly created or modified configuration objects. See Section 3.5, "Viewing Permission Collection and Reconciliation Service Configuration Objects," on page 25.

14 Repeat Step 1 through Step 9 for each driver where you want to add the new packages.

3.5 Viewing Permission Collection and Reconciliation Service Configuration Objects

NOTE: This section contains information about verifying the objects that are either newly created or modified as part of enabling the Permission Collection and Reconciliation service. If this service is not enabled for the driver, skip this section.

After the driver is deployed and configured with the new Permission Collection and Reconciliation service, verify that the driver correctly creates and updates the entitlements information in the Identity Vault.

Complete the following steps:

1 In iManager, click 🌐 to display the Identity Manager Administration page.

2 In the **Administration** list, click **Identity Manager Overview**.

   2a (Conditional) If the driver set is not listed on the **Driver Sets** tab, use the **Search In** field to search for and display the driver set.

   2b Click the driver set to open the Driver Set Overview page.

3 Click the Delimited Text driver icon.

4 Click the **Jobs** tab. The **PermissionOnboarding** job is displayed in the Jobs page. For more information, see "PermissionOnboarding Job" in the *NetIQ Identity Manager Driver Administration Guide*.

5 Click **Advanced > Mapping Tables**. The DNs of the Entitlement objects are displayed in the Mapping Table page based on the InitEntitlementResourceObjects policy and data from the configuration objects. For more information, see "Mapping Tables" in the *NetIQ Identity Manager Driver Administration Guide*.

6 Click **Global Config Values** to display the driver set GCV page.

This page contains two sets of GCVs that are consumed by the drivers in the driver set. Ensure that you configure them for the driver set containing the drivers for quick onboarding of identity, resources, and permission assignments.

   ◆ **NOVLCOMSET:** This GCV object contains the following:

      ◆ **User Container:** Specifies the Identity Vault container where the users are added, if they don't already exist in the Identity Vault. This value is the default value for all drivers in the driver set.

      ◆ **Group Container:** Specifies the Identity Vault container where the groups are added, if they don't already exist in the Identity Vault. This value is the default value for all drivers in the driver set.

   ◆ **NOVLACOMSET:** This GCV object contains the following:

      ◆ **User Application Provisioning Services URL:** Specifies the User Application Identity Manager Provisioning URL.

      ◆ **User Application Provisioning Administrator:** Specifies the DN of the provisioning administrator. This user should have the rights for creating and assigning resources.

4 Upgrading an Existing Driver

If you are running the driver on the Identity Manager server, the driver shim files are updated when you update the server unless they were not selected during a custom installation. If you are running the driver on another server, the driver shim files are updated when you update the Remote Loader on the server.

The 4.5 version of the driver shim supports drivers created by using any 3.*x* version of the driver configuration file. You can continue to use these driver configurations until you want to upgrade the driver to packages.

The following sections provide information to help you upgrade an existing driver to version 4.5:

- Section 4.1, "Supported Upgrade Paths," on page 27
- Section 4.2, "What's New in Version 4.5," on page 27
- Section 4.3, "Upgrade Procedure," on page 27

4.1 Supported Upgrade Paths

You can upgrade from any 3.*x* version of the Delimited Text driver. Upgrading a pre-3.*x* version of the driver directly to version 4.5 is not supported.

4.2 What's New in Version 4.5

Version 4.5 of the driver now provides support for quick onboarding of custom entitlements. For more information, see "Entitlements" on page 12. For more information about how quick onboarding of custom entitlements works, see "Understanding Permission Collection and Reconciliation Service " in the *NetIQ Identity Manager Driver Administration Guide*.

From Identity Manager 4.0 onwards, driver content is delivered in packages instead of through a driver configuration file.

4.3 Upgrade Procedure

The process for upgrading the Delimited Text driver is the same as for other Identity Manager drivers. For detailed instructions, see "Upgrading the Identity Manager Drivers" in the *NetIQ Identity Manager Setup Guide*.

5 Setting Up One-Way Synchronization

If your data synchronization goes only one way, you need to disable the channel that you don't use. To disable one of the channels, clear the filters on the channel you don't need and remove the path for the input or output directory, depending on the channel.

For example, if you only need a Publisher channel:

1 In the Filter editor in iManager, clear the filters on the Subscriber channel.

    1a For example, select the **User** class.
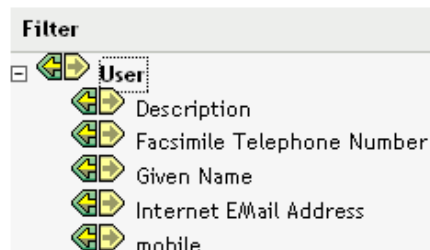
    1b Select **Ignore** in the Subscribe section.

    The Subscriber channel icon for the User class changes to show that the class is no longer being synchronized.
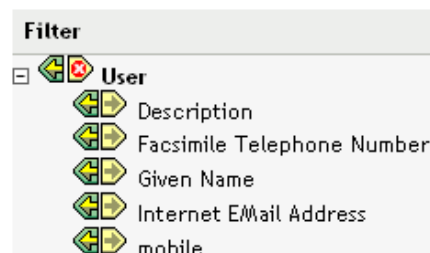
2 Click **OK** to save the changes.

3 Edit the driver properties to remove the path specified for the **Output File Path**. This setting is located under the **Subscriber Settings** for the **Driver Parameters** on the **Driver Configuration** page. For details, see Section A.1.5, "Driver Parameters," on page 45.

If you only need a Subscriber channel, clear the filters on the Publisher object and remove the path specified for the **Input File Path** in the **Driver Parameters** section.

6 Configuring for XDS XML Files

You can use XML files in XDS format instead of comma-separated value (CSV) files with the driver.

Because you generally use this driver only with a Publisher or Subscriber channel, perform only the steps from the section that you need.

6.1 Using the Publisher Channel

To have the driver accept input in XML format, change the input file extension to `.xml`.

6.2 Using the Subscriber Channel

To have the driver send output in XDS format, remove the Event Transform and Output Transform style sheets from the Subscriber channel.

We recommend you to use Designer for modifying or deleting policies, rules, or objects.

1 Launch Designer, then go to the Outline view.
2 Browse to the driver's Subscriber object, then select the SubscriberEventTransformSS object and click **Delete**.
3 Browse to and select the driver's OutputTransformSS object, then click **Delete**.

7 Using Style Sheets to Configure Data Synchronization

The real power of Identity Manager is in managing the shared data itself. This section covers some common customizations for the Delimited Text driver.

The example configuration available with the driver uses comma-separated value files. However, you can use the driver in many ways. It is designed to be as flexible as possible. The driver passes the text-based files largely unchanged to the style sheets. The style sheets do most of the work. You can write new style sheets to allow the driver to work with almost any text-based file that contains predictably repeatable patterns.

The basis for this exchange is the `<delimited-text>` XML element. For example, to design a Publisher channel that reads information from a text file, create an Input Transform style sheet that receives the contents of the file and converts it into a `<delimited-text>` element.

The following is an example of a `<delimited-text>` element:

```
<delimited-text>
     <record>
          <field name="FirstName">John</field>
          <field name="LastName">Maxfield</field>
          <field name="Phone">555-1212</field>
     </record>
     <record>
          <field name="FirstName">Sarah</field>
          <field name="LastName">Lopez</field>
          <field name="Phone">555-3434</field>
     </record>
</delimited-text>
```

For detailed information on writing style sheets to handle other document types, refer to the sample style sheets that come with this driver. If you create the driver by using the example configuration, you can use Input Transform, Output Transform, and Event Transformation style sheets as a starting point.

8 Using Java Interfaces to Customize File Processing

Java interfaces enable you to customize file processing by using Java classes that you write:

- InputSorter
- InputSource
- PreProcessor
- PostProcessor

These interfaces enable you to add extensions, which are optional. The driver continues to function as before without extensions. However, if you want to directly modify the behavior of the driver, but you have been unable to make these modifications from a style sheet or DirXML Script, extending the Delimited Text driver can be useful.

By using Java classes that you write, you can use the interfaces to customize the publish and subscribe processes in the following ways:

Table 8-1 *Customizing the Publish and Subscribe Processes*

Process	Interface	Description
Publish	InputSorter	Defines the processing order of multiple input files.
		The system where your driver is installed determines the default processing order. For example, files on an NT system are processed in alphabetical order. You can use the InputSorter to impose the processing order that you require.
Publish	InputSource	Provides data other than the files in the default location for the driver to process.
		For example, you can check an FTP server for input files and then transfer the files to the local file system for processing.
Publish	PreProcessor	Ties data manipulation required to prepare input files for driver processing directly to the driver.
		Before this interface was available, preprocessing was independent of the driver. You could create a separate application that monitored another directory for input files, modify the files in some way, and then copy the files to the input directory of the driver. By creating a class that implements the PreProcessor, you can do this type of preprocessing more directly.
Subscribe	PostProcessor	Ties data manipulation required by the application, consuming Identity Vault output directly to the driver.

These enhancements to the driver require Java programming. To implement this functionality, complete the following processes:

8.1 Creating a Java Class

JavaDoc and an example class are included with the driver to help you implement this functionality. Find the JavaDoc at `./products/IDM/windows/setup/drivers/delimitedtext/javadocs` and the sample code at `./products/IDM/windows/setup/drivers/delimitedtext/extensions/sample code`.

8.2 Creating a Java .jar File

After you have implemented your class file, create a Java `.jar` file (Java archive) by using the jar tool. The `.jar` file must contain the class that you have created. Put the `.jar` file into the `novell/nds/lib` directory. The path might differ, depending on the platform you're on, but it should be the same location as `DelimitedTextShim.jar` and `DelimitedTextUtil.jar`.

8.3 Configuring the Driver to Use the New Class

After you have placed the new `.jar` file in the correct location, configure the driver to use your new class by modifying the driver's properties.

1 In iManager, open the driver's property page:

    1a In iManager, click 🔵 to display the Identity Manager Administration page.

    1b In the **Administration** list, click **Identity Manager Overview**.

    1c Open the driver set that contains the driver.

    1d Click the driver icon to open the **Identity Manager Driver Overview** page.

    1e Click the upper right corner of the driver icon to open the **Actions** menu, then click **Edit properties**.

2 Select **Driver Configuration**.

3 Scroll to **Driver Parameters**, then click **Edit XML**.

4 Locate the `<publisher-options>` section of the file.

This file defines which parameters and values appear in the Driver Parameters section of the **D**river Configuration page.

For each class you created that works on the Publisher channel, you enter an additional option in the `<publisher-options>` section. After you update this file, you see your new options in the interface.

5 For each new class you created on the Publisher channel, add an entry corresponding to the interface type. Use the following table as a guide:

Interface	New Entry
InputSorter	`<definition display-name="InputSorter Class" name=" input-sorter" type="string"> <value>com.acme.MyNewClass</value> </definition>`
	`<definition display-name="InputSorter init string" name=" input-sorter-params" type="string"> <value>MY CONFIG PARAMS</value> </definition>`
InputSource	`<definition display-name="InputSource Class" name=" input-source" type="string"> <value>com.acme.MyNewClass</value> </definition>`
	`<definition display-name="InputSource init string" name=" input-source-params" type="string"> <value>MY CONFIG PARAMS</value> </definition>`
PreProcessor	`<definition display-name="PreProcessor Class" name="pre-processor" type="string"> <value>com.acme.MyNewClass</value> </definition>`
	`<definition display-name="PreProcessor init string" name="pre-processor-params" type="string"> <value>MY CONFIG PARAMS</value> </definition>`

5a Replace `com.acme.MyNewClass` with the name of the class that you defined, along with a full package identifier.

5b Replace `MY CONFIG PARAMS` with any information that you want to pass to the init method of your class.

The init method of your class is then responsible for parsing the information contained in this string. If your class doesn't require a configuration string to be passed to the init method, you can leave off the whole element, in which case null is passed to the init method.

6 If you created a PostProcessor rule, locate the `<subscriber-options>` section of the file and add the following lines:

```
<post-processor display-name="PostProcessor Class:">com.acme.MyNewClass</post-processor>
```

```
<post-processor-params display-name="PostProcessor init string:">MY CONFIG PARAMS</post-processor-params>
```

6a Replace `com.acme.MyNewClass` with the name of the class that you have defined along with full package information.

6b Replace `MY CONFIG PARAMS` with any information that you want to pass to the init method of your class.

The init method of your class is then responsible for parsing the information contained in this string. If your class doesn't require a configuration string to be passed to the init method, you can leave off the entire element, in which case null would be passed to the init method.

7 Click **OK**.

9 Managing the Driver

As you work with the Delimited Text driver, there are a variety of management tasks you might need to perform, including the following:

- Starting and stopping the driver
- Viewing driver version information
- Using Named Passwords to securely store passwords associated with the driver
- Monitoring the driver's health status
- Backing up the driver
- Inspecting the driver's cache files
- Viewing the driver's statistics
- Using the DirXML Command Line utility to perform management tasks through scripts
- Securing the driver and its information

Because these tasks, as well as several others, are common to all Identity Manager drivers, they are included in one reference, the *NetIQ Identity Manager Driver Administration Guide*.

10 Troubleshooting

Viewing driver processes is necessary to analyze unexpected behavior. To view the driver processing events, use DSTrace. You should only use it during testing and troubleshooting the driver. Running DSTrace while the drivers are in production increases the utilization on the Identity Manager server and can cause events to process very slowly. For more information, see "Viewing Identity Manager Processes" in the *NetIQ Identity Manager Driver Administration Guide*.

10.1 Stop the driver before doing a bulk assignment or removal of resource assignments

If you don't stop the driver before doing a bulk removal of resource assignments from the User Application, the driver creates multiple user entries in the output CSV file. It might display a warning message when you are deploying the driver for the first time.

To workaround this issue, stop the driver before doing such bulk revocations from the User Application, then restart the driver. This creates the output CSV file with one entry instead of multiple intermediate entries in it.

A Driver Properties

A Driver Properties

This section provides information about the Driver Configuration and Global Configuration Values properties for the Delimited Text driver. These are the only unique properties for the Delimited Text driver. All other driver properties (Named Password, Engine Control Values, Log Level, and so forth) are common to all drivers. Refer to "Driver Properties" in the *NetIQ Identity Manager Driver Administration Guide* for information about the common properties.

The properties information is presented from the viewpoint of iManager. If a field is different in Designer, it is marked with a Designer icon.

A.1 Driver Configuration

In iManager:

1 Click ![icon] to display the Identity Manager Administration page.

2 Open the driver set that contains the driver whose properties you want to edit:

    2a In the **Administration** list, click **Identity Manager Overview**.

    2b If the driver set is not listed on the **Driver Sets** tab, use the **Search In** field to search for and display the driver set.

    2c Click the driver set to open the Driver Set Overview page.

3 Locate the Delimited Text driver icon, then click the upper right corner of the driver icon to display the **Actions** menu.

4 Click **Edit Properties** to display the driver's properties page.

In Designer:

1 Open a project in the Modeler.

2 Right-click the driver icon ![icon] or line, then select click **Properties > Driver Configuration.**

The Driver Configuration options are divided into the following sections:

A.1.1　Driver Module

The Driver Module section lets you change the driver from running locally to running remotely or the reverse.

Java: Used to specify the name of the Java class that is instantiated for the shim component of the driver. This class can be located in the `classes` directory as a class file, or in the `lib` directory as a `.jar` file. If this option is selected, the driver is running locally.

The name of the Java class is:

`com.novell.nds.dirxml.driver.delimitedtext.DelimitedTextDriver`

Native: Used to specify the name of the `.dll` file that is instantiated for the application shim component of the driver. If this option is selected, the driver is running locally.

Connect to Remote Loader: Used when the driver is connecting remotely to the connected system. Designer includes two suboptions:

- **Remote Loader Client Configuration for Documentation**: Includes information on the Remote Loader client configuration when Designer generates documentation for the Delimited Text driver.
- **Driver Object Password**: Specifies a password for the Driver object. If you are using the Remote Loader, you must enter a password on this page. Otherwise, the remote driver does not run. The Remote Loader uses this password to authenticate itself to the remote driver shim.

A.1.2　Driver Object Password

Driver Object Password: Specifies a password for the Driver object. If you are using the Remote Loader, you must enter a password on this page. Otherwise, the remote driver does not run. The Remote Loader uses this password to authenticate itself to the remote driver shim.

A.1.3　Authentication

The Authentication section stores the information required to authenticate to the connected system.

Authentication information for server: Displays or specifies the server that the driver is associated with.

Authentication ID: Specify a user application ID. This ID is used to pass Identity Vault subscription information to the application.

Example: `Administrator`

Authentication Context: Specify the IP address or name of the server that the application shim should communicate with.

Remote Loader Connection Parameter: Used only if the driver is connecting to the application through the Remote Loader.

In iManager, the parameter to enter is `hostname=xxx.xxx.xxx.xxx port=xxxx kmo=certificatename`, where the host name is the IP address of the Remote Loader server and the port is the port the Remote Loader is listening on. The default port for the Remote Loader is 8090.

The `kmo` entry is optional. It is used only when an SSL connection exists between the Remote Loader and the Identity Manager engine.

Example: `hostname=10.0.0.1 port=8090 kmo=IDMCertificate`

Application Password: Specify the password for the user object listed in the **Authentication ID** field.

Remote Loader Password: Used only if the driver is connecting to the application through the Remote Loader. The password is used to control access to the Remote Loader instance. It must be the same password specified during the configuration of the Remote Loader on the connected system.

Cache limit (KB): Specify the maximum event cache file size (in KB). If it is set to zero, the file size is unlimited.

Click **Unlimited** to set the file size to unlimited in Designer.

A.1.4   Startup Option

The Startup Option section enables you to set the driver state when the Identity Manager server is started.

Auto start: The driver starts every time the Identity Manager server is started.

Manual The driver does not start when the Identity Manager server is started. The driver must be started through Designer or iManager.

Disabled: The driver has a cache file that stores all of the events. When the driver is set to **Disabled**, this file is deleted and no new events are stored in the file until the driver state is changed to **Manual** or **Auto Start**.

Do not automatically synchronize the driver: This option applies only if the driver is deployed and was previously disabled. If this is not selected, the driver re-synchronizes the next time it is started.

A.1.5   Driver Parameters

The Driver Parameters section lets you configure the driver-specific parameters. When you change driver parameters, you tune driver behavior to align with your network environment. For example, you might find the default Publisher polling interval to be shorter than your synchronization requires. Making the interval longer could improve network performance while still maintaining appropriate synchronization.

The driver parameters are presented by categories:

Driver Options

Driver parameters for server: Displays or specifies the server name or IP address of the server whose driver parameters you want to modify.

Edit XML: Opens an editor so that you can edit the driver's configuration file.

Field Delimiter: Specifies the character that is used to delimit field values in the input files. It must be one character. You can also use the tab as the delimiter field value. Tab is represented as {tab}. The default is a comma.

If the values of any of the input fields contain this character, enclose the entire value in quotes to prevent it from being seen as a delimiter.

Changing this delimiter parameter to something other than a comma does not automatically change the delimiter character used in the output files when a Subscriber is used. To change the delimiter character in the output files, edit the Output Transform style sheet. The delimiter character is assigned to a variable at the beginning of that style sheet. For example, to change the delimiter, locate the `<xsl:variable name="delimiter" select="','"/>` line in the default style sheet and modify it appropriately.

Field Names: Specifies a comma-separated list of attribute names that can be referred to in the Schema Mapping rule. In the input files, the fields of the records must correspond to the order and positioning of the names in this list.

For example, if you list eight field names in this parameter, each record of the input files should have eight fields separated by the field delimiter character. On Windows, see `sample.csv` in the `delimitedtext/samples` directory for an example. On Solaris and Linux, `sample.csv` is located in the `/usr/lib/dirxml/rules/delim` directory.

The default values are LastName, FirstName, Title, Email, WorkPhone, Fax, WirelessPhone, and Description.

Object Class Name: Specifies the NetIQ eDirectory class name that should be used when creating new objects to correspond to input files.

Allow Driver to Consume Its Own Output: Prevents you from inadvertently creating a situation in which the driver writes output files that are immediately read in again as input of the same driver.

The default is No. By default, the driver won't load if all the following conditions occur:

- You have both a Subscriber channel and a Publisher channel.
- The input and output directories are the same.
- The input and output file extensions are the same.

If you want to feed the output of the Subscriber channel into the input of the Subscriber channel as a way to detect Identity Vault events to trigger other changes in the Identity Vault, set this parameter to Yes. For example, to update the Full Name attribute when the Given Name, Surname, or Initials attributes are updated, set this parameter to Yes.

Subscriber Options

Output File Path: Specifies the directory on the local file system where output files will be created. An error occurs if this directory doesn't exist. The default values are:

Windows: `c:\csvsample\output`

Solaris or Linux: `/csvsample/output`

Output File Extension: When this parameter contains no value, the default Java character encoding for your locale is used.

Output files have a unique name that ends with the characters in the Output File Extension parameter. If the output files from a Subscriber channel are used as input files for the Publisher channel of another Delimited Text driver, the destination file extension must match the source file extension parameter of the second driver.

Add Header to Output File: This parameter is set to No by default. When this parameter is set to Yes and the output file is of `.xml` format, the driver adds the xml header to the output file. No header is added if the output file has any other format other than `.xml`.

Destination File Character Encoding (leave blank for default): To use an encoding other than the default for your locale, enter one of the canonical names from the Supported Encoding table.

The Publisher and Subscriber channels can use different character encodings.

Maximum Number of Transactions per Output File: Specifies the maximum number of transactions that are written to a single output file. When the file transaction limit is reached, the file closes, and a new file is created for subsequent transactions. To limit the number of transactions that can be written to a single file, leave this parameter blank or set it to zero.

For more information, see the next item, "Maximum Time in Seconds before Flushing All Transactions:" on page 47.

Enable/Disable Duplicate Records: To allow duplicate records to be written into the output files, change this parameter to **Yes**. By default, the parameter is set to **No**.

Maximum Time in Seconds before Flushing All Transactions: If no new transactions have been written to the output file in the amount of time specified in this parameter, the file is closed. When new transactions need to be written, a new output file is created. If you don't want to limit the time that can pass before the output file is closed, leave this parameter blank or set it to zero.

Time of Day (Local Time) to Flush All Transactions: If a value is supplied for this parameter, the current output file is closed at the specified time each day. Subsequent transactions are written to a new file. This parameter does not prevent the **Maximum Number of Transactions per Output File** or the **Maximum Time in Seconds before Flushing All Transactions** parameters from also acting as output file thresholds. If you use this parameter and only want one file per day, set the other two parameters to zero.

The format of this parameter can be HH:MM:SS (using the 24-hour clock) or H:MM:SS AM/PM. An hour is required, but the minutes and seconds are optional. Because the parameter assumes local time, any time zone information included in the value is ignored.

The previous three parameters (**Maximum Number of Transactions per Output File**, **Maximum Time in Seconds before Flushing All Transactions**, and **Time of Day to Flush All Transactions**) are all capable of acting as a threshold for the transaction size a file is able to grow to, or for the time that it remains open to accept new transactions.

As long as an output file is still open for writing by the Delimited Text driver, it shouldn't be considered as finalized. Avoid opening the file in any other process until the driver closes the file. For this reason, one of the three previous parameters must be set to assure that output files don't remain open indefinitely. To avoid this condition, if the driver detects that all three parameters are blank (or zero), it automatically sets the **Maximum Number of Transactions per Output File** to the value of 1.

Publish Options

Input File Path: The Publisher channel looks for new input files in the Input File Path, which is a directory on the local file system. Example paths:

* On Windows: `c:\csvsample\input`
* On Solaris and Linux: `/usr/lib/dirxml/rules/delim`

Input File Extension: The extension used to designate input files (for example, `csv`).

Source File Character Encoding (leave blank for default): When this parameter contains no value, the default Java character encoding for your locale is used.

To use an encoding other than the default for your locale, enter one of the canonical names from the Supported Encoding table.

If the Input File Extension parameter is `.xml`, the Source File Character Encoding can be indicated in one of two ways.

- If a value is indicated in the **Source File Character Encoding** parameter, it is used.
- If the parameter is blank, and if the XML document specifies an Encoding Declaration as described in the W3C XML Recommendation in paragraph 4.3.3, the Encoding Declaration is handled by the XML parser in the Identity Manager engine.

  The Identity Manager XML parser handles the following character encodings:
  - UTF-8
  - UTF-16
  - ISO-8859-1
  - US-ASCII

Rename File Extension: The Publisher channel uses only files that have the extension specified in the parameter. After the files have been processed, the value of the **Rename File Extension** parameter is appended to the filename, so the Publisher channel won't try to process the same file again. If the value of the Rename File Extension parameter is left blank, the source file is deleted after it is processed.

IMPORTANT: If you change the default, use only characters that are valid in filenames on your platform. Invalid characters cause the rename to fail and the driver to reprocess the same file repeatedly.

Polling Rate (in Seconds): When the Publisher channel has finished processing all source files, it waits the number of seconds specified in this parameter before checking for new source files to process.

Publisher heartbeat interval: Configures the driver shim to send a periodic status message on the Publisher channel when there has been no Publisher traffic for the given number of minutes.

A.1.6 ECMAScript (Designer Only)

Enables you to add ECMAScript resource files. The resources extend the driver's functionality when Identity Manager starts the driver.

A.1.7 Global Configurations

Displays an ordered list of Global Configuration objects. The objects contain extension GCV definitions for the driver that Identity Manager loads when the driver is started. You can add or remove the Global Configuration objects, and you can change the order in which the objects are executed.

A.2 Global Configuration Values

Global configuration values (GCVs) enable you to specify settings for the Identity Manager features such as password synchronization and driver heartbeat, as well as settings that are specific to the function of an individual driver configuration. Some GCVs are provided with the drivers, but you can also add your own.

In iManager:

1 Click to display the Identity Manager Administration page.

2 Open the driver set that contains the driver whose properties you want to edit:

    2a In the **Administration** list, click **Identity Manager Overview**.

    2b If the driver set is not listed on the **Driver Sets** tab, use the **Search In** field to search for and display the driver set.

    2c Click the driver set to open the Driver Set Overview page.

3 Locate the Delimited Text driver icon, then click the upper right corner of the driver icon to display the **Actions** menu.

4 Click **Edit Properties** to display the driver's properties page.

5 Click **Global Config Values** to display the GCV page.

In Designer:

1 Open a project in the Modeler.

2 Right-click the driver icon or line, then select **Properties > Global Configuration Values.**

The Global Configuration Values are divided into the following categories:

- Section A.2.1, "Default Configuration," on page 49
- Section A.2.2, "Entitlements," on page 50
- Section A.2.3, "Password Synchronization," on page 50
- Section A.2.4, "Managed System Information," on page 51

A.2.1 Default Configuration

The following GCVs control the configuration of the Delimited Text driver:

- **Field Delimiter:** Specifies the character that is used to delimit field values in the input files. It must be one character. You can also use the tab as the delimiter field value. Tab is represented as {tab}. The default is a comma.

  If the values of any of the input fields contain this character, enclose the entire value in quotes to prevent it from being seen as a delimiter. Changing this delimiter parameter to something other than a comma does not automatically change the delimiter character used in the output files when a Subscriber is used. To change the delimiter character in the output files, edit the Output Transform style sheet. The delimiter character is assigned to a variable at the beginning of that style sheet. For example, to change the delimiter, locate the <xsl:variable name="delimiter" select="','"/> line in the default style sheet and modify it appropriately.

  This option in the driver configuration synchronizes User, Group, Organization, Country, and Organizational Unit objects. It also mirrors the structure of a subtree in the other tree.

- **Field Names:** Specifies a comma-separated list of attribute names that can be referred to in the Schema Mapping rule. In the input files, the fields of the records must correspond to the order and positioning of the names in this list.

  For example, if you list eight field names in this parameter, each record of the input files should have eight fields separated by the field delimiter character. On Windows, see sample.csv in the delimitedtext/samples directory for an example. On Solaris and Linux, sample.csv is located in the /usr/lib/dirxml/rules/delim directory.

  The default values are LastName, FirstName, Title, Email, WorkPhone, Fax, WirelessPhone, and Description.

- **Association Attribute:** Specifies the attribute value used for association.

- **Suppress Encrypted Attribute in Trace:** This GCV is effective only when any of the attributes is set as Encrypted Attribute or only when Encrypted Attribute policy is active. When this attribute is set to **Yes**, the complete content of the csv file is suppressed. In case you wish to view the content, set the attribute to **No**.

  By default the GCV is set to **Yes**.

A.2.2 Entitlements

The following options are enabled or displayed when the Entitlements package is installed. For more information about entitlements, see Section 1.3.2, "Entitlements," on page 12.

Enable Permission Collection and Reconciliation: Select the value of this parameter to **True** for allowing permission collection and entitlement assignment. By default it is set to **False**, which allows the driver to override any other conditions to reconcile custom entitlements.

Enable Permission Reconciliation for all Custom Entitlements: If the value of this parameter is set to **No**, it allows you to select specific custom entitlements for reconciling them. By default, it is set to **Yes**, which allows reconciling of all custom entitlements.

Add Custom Entitlements for Reconciliation: This parameter is presented if the value of **Enable Permission Reconciliation for all Custom Entitlements** is set to **No**.

Click the **Add** ✚ icon to add custom entitlements you want to selectively onboard and specify **Assignment Attribute Name** for them.

A.2.3 Password Synchronization

The following GCVs control password synchronization for the Delimited Text driver. For more information, see the *NetIQ Identity Manager Password Management Guide*.

In Designer, you must click the 🖣 icon next to a GCV to edit it. This displays the Password Synchronization Options dialog box for a better view of the relationship between the different GCVs.

In iManager, you should edit the Password Management Options on the **Server Variables** tab rather than under the GCVs. The Server Variables page has a better view of the relationship between the different GCVs.

Connected System Name or Driver Name: Specify the name of the driver. The e-mail notification template uses this value to identify the source of the notification message.

Application accepts passwords from Identity Manager: If **True**, allows passwords to flow from the Identity Manager data store to the connected system.

Identity Manager accepts passwords from application: If **True**, allows passwords to flow from the connected system to Identity Manager.

Publish passwords to NDS password: Use the password from the connected system to set the non-reversible NDS password in eDirectory.

Publish passwords to Distribution Password: Use the password from the connected system to set the NMAS Distribution Password used for Identity Manager password synchronization.

Require password policy validation before publishing passwords: If **True**, applies NMAS password policies during publish password operations. The password is not written to the data store if it does not comply.

Reset user's external system password to the Identity Manager password on failure: If **True**, on a publish Distribution Password failure, attempt to reset the password in the connected system by using the Distribution Password from the Identity Manager data store.

Notify the user of password synchronization failure via e-mail: If **True**, notify the user by e-mail of any password synchronization failures.

A.2.4 Managed System Information

These settings help the Identity Reporting Module function to generate reports. There are different sections in the **Managed System Information** tab.

General Information

Name: Specifies a descriptive name for this Identity Vault.

Description: Specifies a brief description of this Identity Vault.

Location: Specifies the physical location of this Identity Vault.

Vendor: Specifies the vendor of the Identity Vault.

Version: Specifies the version of this Identity Vault.

System Ownership

Business Owner: Browse to and select the business owner in the Identity Vault for this Identity Vault. You must select a user object, not a role, group, or container.

Application Owner: Browse to and select the application owner in the Identity Vault for this Identity Vault. You must select a user object, not a role, group, or container.

System Classification

Classification: Specifies the classification of the Identity Vault. This information is displayed in the reports. The options are:

- Mission-Critical
- Vital
- Not-Critical

- Other

  If you select **Other**, you must specify a custom classification for the Identity Vault.

Environment: Specifies the type of environment the Identity Vault provides. The options are:

- Development
- Test
- Staging
- Production
- Other

  If you select **Other**, you must specify a custom classification for the Identity Vault.

Connection and Miscellaneous Information

Connection and miscellaneous information: This options is always set to **hide**, so that you don't make changes to these options.

B Delimited Text Driver Extensions

The Delimited Text driver defines different interfaces that you can implement in Java classes to extend the base functionality of the driver.

Figure B-1   Java Class Interfaces
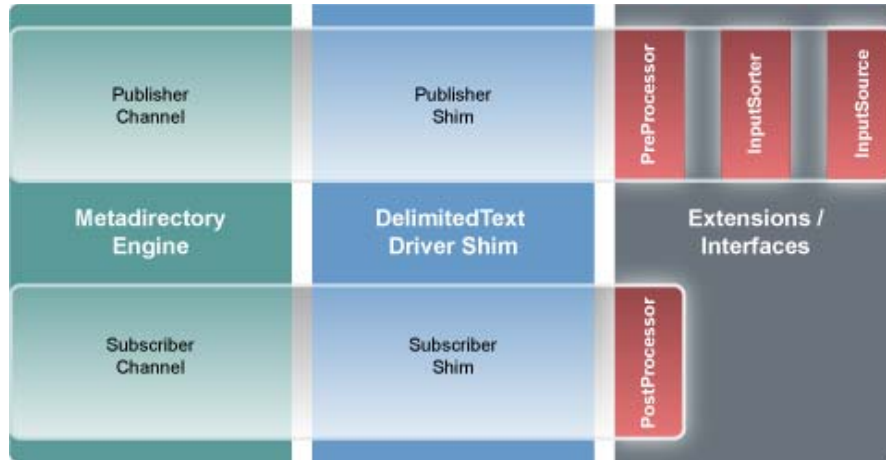
This section includes the following information about using the ImageFile and ImageSource extensions:

B.1   Using the ImageFile InputSource Extension

The ImageFile InputSource extension allows you to easily import GIF, JPG, and PNG images into eDirectory. The InputSource reads the image files from a specified directory, transforms them to a Base64-encoded string, and passes this string to the driver shim as if it were a normal delimited text string, together with additional information about the image, such as file size, file name, and modification date.

Standard rules and style sheets, as used for other implementations of the Delimited Text driver, can be used to further process the image data. The image itself is meant to be stored in an attribute of syntax octet string or stream.

B.1.1   Installing the ImageFile InputSource Extension

The ImageFile InputSource extensions are installed when you install Identity Manager and select the Delimited Text driver. The extensions are configured in the driver parameters. Each extension takes a parameter string that is specified in the driver parameters and passes it to the extension during initialization. Extensions define their own format for the parameter string.

The extensions are included in the `DelimitedTextUtil.jar` file.

B.1.2   Configuring the Driver for the ImageFile InputSource Extension

Use the following table to customize your driver for the ImageFile extension.

Table B-1   *Delimited Text Driver Parameters*

Driver Parameter	XML Name	Sample Values	Purpose
Field Delimiter	field-delimiter	#	Indicates the character that is used to delineate field values in the input files. It must be one character.This must be set to the same character as the delim extension parameter or, if delim is not specified, set it to #.

Driver Parameter	XML Name	Sample Values	Purpose
Field Names (Field1, Field2, Field3…)	field-names	idx name prefix suffix size modified pic64	A comma-separated list of attribute names that can be referred to in the Schema Mapping rule. In the input files, the fields of the records must correspond to the order and positioning of the names in this list. For example, if you list eight field names in this parameter, then each record of the input files should have eight fields separated by the field delimiter character.The extension defines which fields it supports. The fields as listed here are the ones delivered by the extension. Altering the fields can cause malfunctions.
InputSource Class	input-source	com.novell.nds.dirxml.driver.delimitedtext.imagefile.ImageFileInputSource	Class name of the input source.
InputSource init string	input-source-params	srcdir=c:\temp\dirxml;renameto=bak;delblacklist=false;renblacklistto=ignore;minsize=1000;debug=true	InputSource initialization parameters. For more information, see Section B.3.2, "Configuring the Driver for the ImageFile Extension," on page 57.

B.2 Customizing ImageFile InputSource

You can fine-tune the behavior of the ImageFile InputSource by setting the parameters that are discussed in this section. Parameters are passed to the InputSource as a string.

The string must be in the following format:

```
<name1>=<value1>;<name2>=<value2>;<name n>=<value n>
```

Sample:

```
srcdir=c:\temp\dirxml;renameto=bak;delblacklist=false;renblacklistto=ignore;minsize=1000;debug=true
```

Use the following values to configure the ImageFile properties:

Table B-2   *ImageFile InputSource Parameters*

Parameter	Required	Default	Purpose
srcdir	yes		Directory where the image files are stored.
minsize	no	-1	Minimum size of the file to be processed. Number is the number of bytes. Set the value to -1 to disable the filter.
maxsize	no	-1	Maximum size of the file to be processed, in bytes. Set the value to -1 to disable the filter.

Parameter	Required	Default	Purpose
minwidth	no	-1	Minimum width of the image to be processed (number of pixels). Set the value to -1 to disable the filter.
maxwidth	no	-1	Maximum width of the image to be processed (number of pixels). Set the value to -1 to disable the filter.
minheight	no	-1	Minimum height of the image to be processed (number of pixels). Set the value to -1 to disable the filter.
delim	no	#	Delimiter to be used in the created input files. The same delimiter must be configured in the driver parameters for the Delimited Text driver.
consolidate	no	true	Set consolidate to True to produce only one input file for all images. Set consolidate to False to produce an input file for each image.
renameto	no		Specify the suffix you want to be appended to the image files after processing. By default, the renameto parameter is not set and the files are deleted after processing.
debug	no	false	Set to True to turn on debug messages. Debugging is turned off by default.
delblacklist	no	true	Set to False to prevent deletion of the blacklist. The blacklist consists of all the image files that have been filtered out. By default, the blacklist is deleted.
renblacklistto	no		Specify any file suffix you want blacklist image files to be renamed to. By default the renameto parameter is not set and the processed files are handled according to the delblacklist parameter.

B.3 Using the ImageFile PostProcessor

The ImageFile PostProcessor extension allows you to easily export images from the directory to the file system as JPG or PNG files. The PostProcessor further processes the output file generated by the driver. It expects the output file to be in a certain format. The images are contained in the output file as Base64-encoded strings and are then converted into binary blobs and written to files in a specified directory.

Figure B-3   *ImageFile PostProcessor Extension*

Policies and style sheets can be used to control the export process. The images are usually stored in an attribute of syntax octet string or stream for binary data.

B.3.1   Installing the ImageFile PostProcessor Extension

The ImageFile PostProcessor extensions are installed when you install Identity Manager and select the Delimited Text driver. The extensions are configured in the driver parameters. Each extension takes a parameter string that is specified in the driver parameters and passes it to the extension during initialization. Extensions define their own format for the parameter string.

The extensions are included in the `DelimitedTextUtil.jar` file.

B.3.2   Configuring the Driver for the ImageFile Extension

Driver Module

The ImageFile InputSource requires the Delimited Text driver.

1 In Designer, right-click the Delimited Text driver, then click **Properties**.

2 Select **Java**, then specify
   `com.novell.nds.dirxml.driver.delimitedtext.DelimitedTextDriver` if the driver is running locally

   or

   Select **Connect to Remote Loader**,

3 Click **OK** to save the changes.

4 (Conditional) If you are running the Remote Loader, you must add the following class for the Delimited Text driver
   `com.novell.nds.dirxml.driver.delimitedtext.DelimitedTextDriver.`

   For more information, see "Configuring the Remote Loader" in the *Identity Manager 4.5 Remote Loader Guide*.

Driver Parameters

Use the following table to configure the driver. For additional information regarding other driver parameters, refer to the Identity Manager Driver for Delimited Text documentation. (http://www.NetIQ.com/documentation/idm45drivers/delimited/data/bktitle.html)

Table B-3 *ImageFile PostProcessor Parameters*

Parameter	XML Name	Sample Value	Purpose
Field Delimiter	field-delimiter	#	Indicates the character that is used to delineate field values in the input files. It must be one character.This must be set to the same character as the delim extension parameter or, if delim is not specified, set it to #.
Field Names (Field1, Field2, Field3…)	field-names	idx name prefix suffix size modified pic64	A comma-separated list of attribute names that can be referred to in the Schema Mapping rule. In the input files, the fields of the records must correspond to the order and positioning of the names in this list. For example, if you list eight field names in this parameter, then each record of the input files should have eight fields separated by the field delimiter character.The extension defines which fields it supports. The fields as listed here are the ones delivered by the extension. Altering the fields can cause malfunctions.
PostProcessor Class	post-processor	com.novell.nds.dirxml.driver.delimitedtext.imagefile.ImageFileInputSource	Class name of the input source.
PostProcessor init string	post-processor-params	destdir=/var/novell/idm/users/output/images;format=png;debug=true	PostProcessor initialization parameters. For more information, see Section B.1.1, "Installing the ImageFile InputSource Extension," on page 54.

B.4 Customizing the ImageFile Extension

You can fine-tune the behavior of the ImageFile extension by setting parameters. Parameters are passed to the PostProcessor as a string.

The string must be in the following format:

```
<name1>=<value1>;<name2>=<value2>;<name n>=<value n>
```

Sample:

```
destdir=/var/novell/idm/users/output/images;format=png;debug=true
```

Use the following table to customize the ImageFile extension.

Table B-4   *ImageFile Custom Parameters*

Parameter	Required	Default Value	Purpose
destdir	yes		Directory where the image files are stored.
delim	no	#	Used when parsing the output files. The same delimiter should be configured in the Delimited Text driver parameters.
format	no	png	Image format.
debug	no	false	Set the value to True to turn on debugging. Debugging is off by default.