ORACLE
INSURANCE

Oracle® Documaker

Documaker Enterprise

Administration Guide

version 12.0

Part number: E17552-01

December 2011

THIRD PARTY SOFTWARE NOTICES

This product includes software developed by Apache Software Foundation (http://www.apache.org/).

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software distributed via the Berkeley Software Distribution (BSD) and licensed for binary distribution under the Generic BSD license.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by the JDOM Project (http://www.jdom.org/).

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE JDOM AUTHORS OR THE PROJECT CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by the Massachusetts Institute of Technology (MIT).

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This product includes software developed by Jean-loup Gailly and Mark Adler. This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

This software is based in part on the work of the Independent JPEG Group (http://www.ijg.org/).

This product includes software developed by the Dojo Foundation (http://dojotoolkit.org).

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 2005-2009, The Dojo Foundation. All rights reserved.

This product includes software developed by W3C.

Copyright © 2009 World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. (http://www.w3.org/Consortium/Legal/)

This product includes software developed by Mathew R. Miller (http://www.bluecreststudios.com).

Copyright (c) 1999-2002 ComputerSmarts. All rights reserved.

This product includes software developed by Shaun Wilde and distributed via Code Project Open License (http://www.codeproject.com).

THIS WORK IS PROVIDED "AS IS", "WHERE IS" AND "AS AVAILABLE", WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS OR GUARANTEES. YOU, THE USER, ASSUME ALL RISK IN ITS USE, INCLUDING COPYRIGHT INFRINGEMENT, PATENT INFRINGEMENT, SUITABILITY, ETC. AUTHOR EXPRESSLY DISCLAIMS ALL EXPRESS, IMPLIED OR STATUTORY WARRANTIES OR CONDITIONS, INCLUDING WITHOUT LIMITATION, WARRANTIES OR CONDITIONS OF MERCHANTABILITY, MERCHANTABLE QUALITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY WARRANTY OF TITLE OR NON-INFRINGEMENT, OR THAT THE WORK (OR ANY PORTION THEREOF) IS CORRECT, USEFUL, BUG-FREE OR FREE OF VIRUSES. YOU MUST PASS THIS DISCLAIMER ON WHENEVER YOU DISTRIBUTE THE WORK OR DERIVATIVE WORKS.

This product includes software developed by Chris Maunder and distributed via Code Project Open License (http://www.codeproject.com).

THIS WORK IS PROVIDED "AS IS", "WHERE IS" AND "AS AVAILABLE", WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS OR GUARANTEES. YOU, THE USER, ASSUME ALL RISK IN ITS USE, INCLUDING COPYRIGHT INFRINGEMENT, PATENT INFRINGEMENT, SUITABILITY, ETC. AUTHOR EXPRESSLY DISCLAIMS ALL EXPRESS, IMPLIED OR STATUTORY WARRANTIES OR CONDITIONS, INCLUDING WITHOUT LIMITATION, WARRANTIES OR CONDITIONS OF MERCHANTABILITY, MERCHANTABLE QUALITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY WARRANTY OF TITLE OR NON-INFRINGEMENT, OR THAT THE WORK (OR ANY PORTION THEREOF) IS CORRECT, USEFUL, BUG-FREE OR FREE OF VIRUSES. YOU MUST PASS THIS DISCLAIMER ON WHENEVER YOU DISTRIBUTE THE WORK OR DERIVATIVE WORKS.

This product includes software developed by PJ Arends and distributed via Code Project Open License (http://www.codeproject.com).

THIS WORK IS PROVIDED "AS IS", "WHERE IS" AND "AS AVAILABLE", WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS OR GUARANTEES. YOU, THE USER, ASSUME ALL RISK IN ITS USE, INCLUDING COPYRIGHT INFRINGEMENT, PATENT INFRINGEMENT, SUITABILITY, ETC. AUTHOR EXPRESSLY DISCLAIMS ALL EXPRESS, IMPLIED OR STATUTORY WARRANTIES OR CONDITIONS, INCLUDING WITHOUT LIMITATION, WARRANTIES OR CONDITIONS OF MERCHANTABILITY, MERCHANTABLE QUALITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY WARRANTY OF TITLE OR NON-INFRINGEMENT, OR THAT THE WORK (OR ANY PORTION THEREOF) IS CORRECT, USEFUL, BUG-FREE OR FREE OF VIRUSES. YOU MUST PASS THIS DISCLAIMER ON WHENEVER YOU DISTRIBUTE THE WORK OR DERIVATIVE WORKS.

This product includes software developed by Erwin Tratar. This source code and all accompanying material is copyright (c) 1998-1999 Erwin Tratar. All rights reserved.

THIS SOFTWARE IS PROVIDED "AS IS" WITHOUT EXPRESS OR IMPLIED WARRANTY. USE IT AT YOUR OWN RISK! THE AUTHOR ACCEPTS NO LIABILITY FOR ANY DAMAGE/LOSS OF BUSINESS THAT THIS PRODUCT MAY CAUSE.

This product includes software developed by Sam Leffler of Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE

Copyright (c) 1988-1997 Sam Leffler
Copyright (c) 1991-1997 Silicon Graphics, Inc.

This product includes software developed by Guy Eric Schalnat, Andreas Dilger, Glenn Randers-Pehrson (current maintainer), and others. (http://www.libpng.org)

The PNG Reference Library is supplied "AS IS". The Contributing Authors and Group 42, Inc. disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The Contributing Authors and Group 42, Inc. assume no liability for direct, indirect, incidental, special, exemplary, or consequential damages, which may result from the use of the PNG Reference Library, even if advised of the possibility of such damage.

This product includes software components distributed by the Cryptix Foundation.

THIS SOFTWARE IS PROVIDED BY THE CRYPTIX FOUNDATION LIMITED AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE CRYPTIX FOUNDATION LIMITED OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

Copyright © 1995-2005 The Cryptix Foundation Limited. All rights reserved.

This product includes software components distributed by Sun Microsystems.

This software is provided "AS IS," without a warranty of any kind. ALLEXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANYIMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN AND ITS LICENSORS SHALL NOT BELIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THE SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Copyright (c) 1998 Sun Microsystems, Inc. All Rights Reserved.

This product includes software components distributed by Dennis M. Sosnoski.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS'" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2003-2007 Dennis M. Sosnoski. All Rights Reserved

It also includes materials licensed under Apache 1.1 and the following XPP3 license

THIS SOFTWARE IS PROVIDED "AS IS'" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2002 Extreme! Lab, Indiana University. All Rights Reserved

---

This product includes software components distributed by CodeProject. This software contains material that is © 1994-2005 The Ultimate Toolbox, all rights reserved.

This product includes software components distributed by Geir Landro.

Copyright © 2001-2003 Geir Landro (drop@destroydrop.com) JavaScript Tree - www.destroydrop.com/hjavascripts/tree/version 0.96

This product includes software components distributed by the Hypersonic SQL Group.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

This product includes software components distributed by the International Business Machines Corporation and others.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This product includes software components distributed by the University of Coimbra.

University of Coimbra distributes this software in the hope that it will be useful but DISCLAIMS ALL WARRANTIES WITH REGARD TO IT, including all implied warranties of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. In no event shall University of Coimbra be liable for any special, indirect or consequential damages (or any damages whatsoever) resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

This product includes software components distributed by Steve Souza.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "AS IS"' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by the OpenSymphony Group (http://www.opensymphony.com/.)"

CONTENTS

Preface

This document contains information necessary for the configuration of Oracle Documaker Enterprise, including Oracle Documaker Document Factory and Oracle Documaker Interactive.

Note The installation of Oracle Documaker Enterprise is covered in the Documaker Enterprise Installation Guide.

AUDIENCE

This document is intended for users who want to administer Documaker Enterprise. Experience installing Oracle Documaker and experience as a system administrator is necessary.

In addition to this guide, implementation of Document Factory with Documaker requires familiarity with Oracle Documaker configuration and processing. You can find this information in the various Documaker manuals, by taking Documaker training classes, or via hands-on experience.

Once familiar with the material in this guide and other prerequisite background information, an administrator should be able to plan, execute, and manage the day to day operation of a Documaker Enterprise environment.

DOCUMENTATION ACCESSIBILITY

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, refer to the following Oracle resources:

- The Oracle Documaker documentation set, specifically:
  - Documaker Enterprise Installation Guide
  - Documaker Installation Guide
  - Documaker Administration Guide
- To make sure you have the latest documentation, visit the Oracle Technology Network:

  http://www.oracle.com/technetwork/documentation/insurance-097481.html

Conventions

The following text conventions are used in this document:

Convention	Description
bold	Indicates information you enter.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands, URLs, code in examples, and text that appears on the screen.

Chapter 1

Introduction to Document Factory

Oracle Documaker Document Factory is a publishing application that uses an assembly line processing methodology. The Document Factory is a part of Oracle Documaker Enterprise which also includes web based applications that manage and use the Document Factory application.

This chapter provides an introduction to Document Factory and covers these topics:

OVERVIEW

Oracle Documaker Document Factory is a publishing application that uses an assembly line processing methodology. Document Factory supports parallel processing and real-time monitoring and reporting capabilities and through a framework known as the Document Factory Dashboard.

The Document Factory encompasses a processing model referred to as the Automated Document Factory (ADF). The architecture of this model incorporates the vision of document creation and delivery for mission-critical documents. The ADF vision equates the concepts of factory production to document production by integrating the following within a document publishing environment:

- Template design
- Data input and transformation
- Delivery preparation
- Response management activities

All of these were previously part of Oracle Documaker technology, but the introduction of the Document Factory model in Oracle Documaker 12.0 enhances the underlying architecture to provide parallel processing, integrated logging and error handling, as well as a control and reporting layer across the factory.

BENEFITS OF DOCUMENT FACTORY

Documaker's Document Factory application provides many benefits to a publishing environment, including:

- A system architecture that works well with a clustered, load-balanced, multi-server environment, one that supports fail-over and automatically restarts.
- Support for both real-time and batch processing within the same architecture.
- A high level of functionality and ready-to-use capability, based on 20+ years of industry expertise in production document output requirements.
- A single monitoring point over multiple deployments for easier administration and trouble-shooting, as well as data for business reports that help you track resource usage and manage your document production operation.
- An efficient output factory for communications to enable low-cost, high-quality output.

DOCUMENT FACTORY COMPONENTS

The Oracle Documaker Document processing model is a series of processes that are managed by a Supervisor service. The Scheduler is responsible for moving work flow throughout the factory to all of the other processes that transform the input data into published output. Once input data is received, all activity and logging are stored in the backbone of the system, the Document Factory Database Assembly Line processing tables.

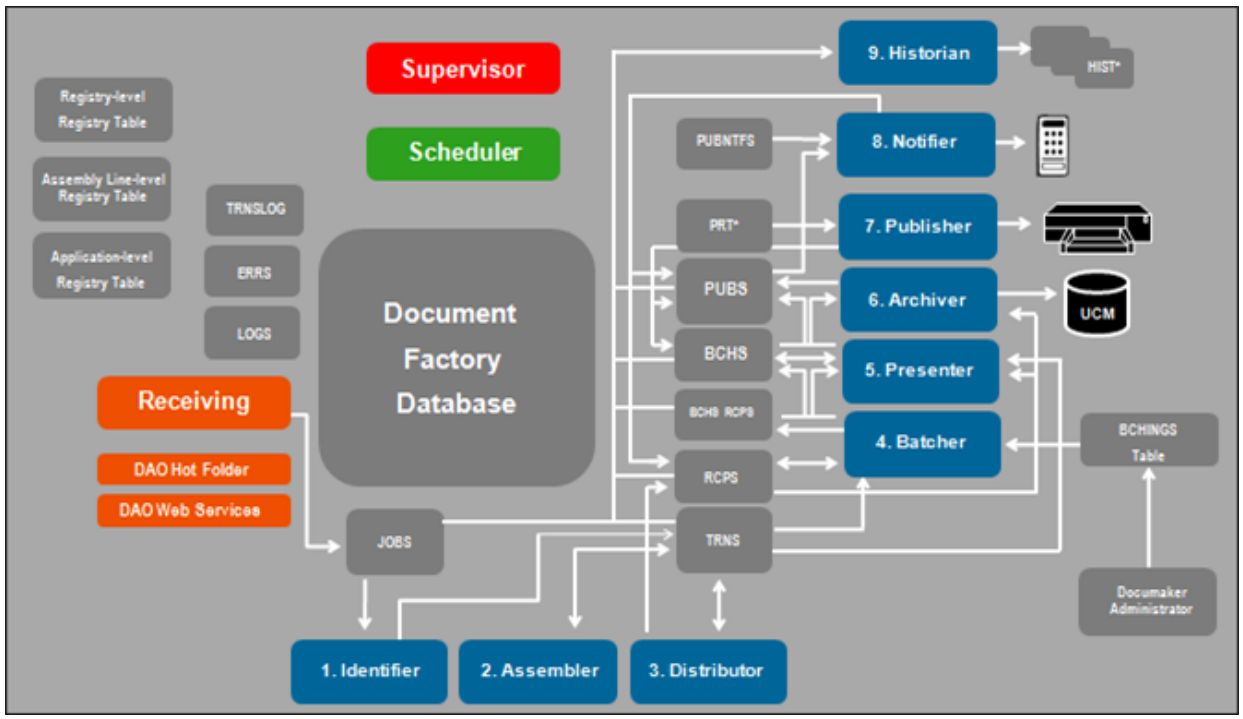


Figure 1: Primary components of the Oracle Documaker Document Factory

There are five primary components of the Document Factory. They are database tables and managers, the receiving process, the Supervisor process, the Scheduler process, and the Workers (Identifier, Assembler, Distributor, Batcher, Presenter, Archiver, Publisher, Notifier, and Historian).

The following is an overview of the primary Document Factory processes. See *Configuring Document Factory* on page 61 for more detailed information on each process.

Database Tables and Managers

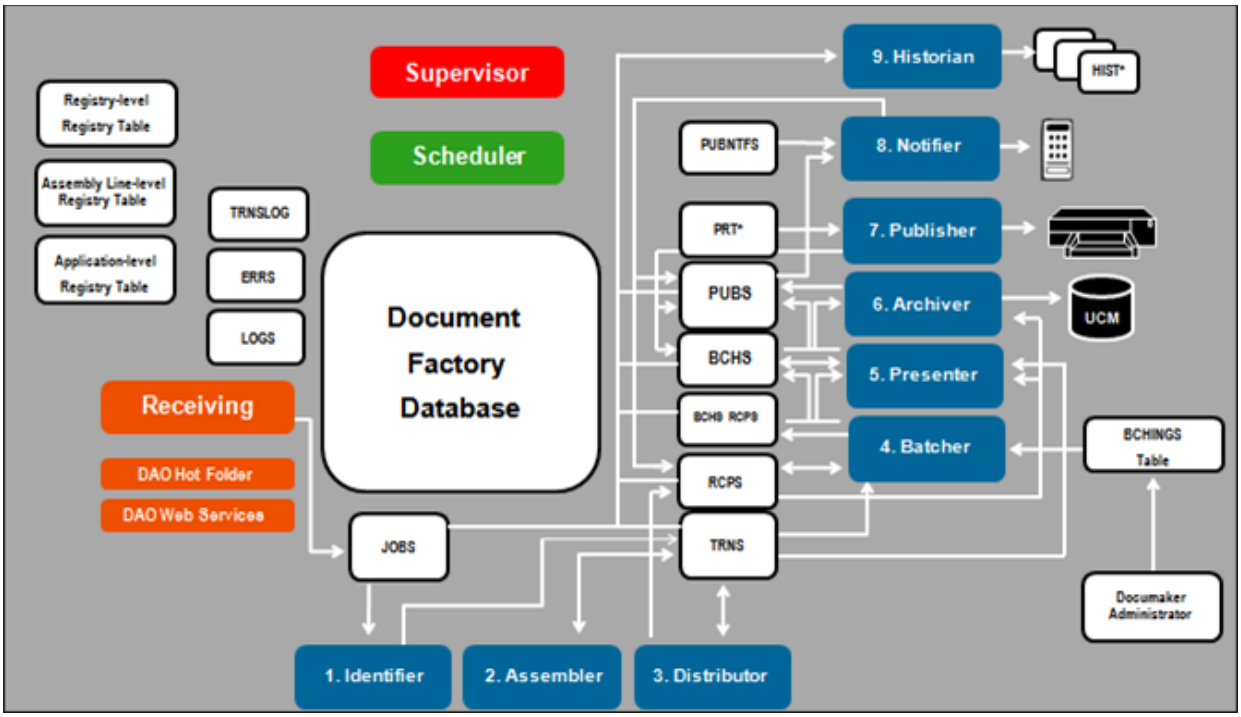


Figure 2: The database tables and managers of Document Factory

The following table describes Document Factory database tables and associated configuration web application.

Table type	Description
Registry Tables	There are three levels of registry configuration tables: • System level • Assembly Line level • Application-level, either a specific worker or a direct application of the Assembly Line - such as Documaker Interactive: Correspondence. Defined by the SQL provided during installation. Set up by a Database Administrator.

Table type	Description
Processing Tables	Each Assembly Line within the Document Factory has its own set of processing tables to store and manage document related activity. • Job table: one record for each submitted job. • Transaction table: one or more per job. • Recipient table: one per recipient or addressee for each transaction. • Batch table: one record for each group (or individual if immediate print) of recipients that have distribution options enabled. Distribution options are archive, publication, and publication notification. • Recipient-batch relationship table: links recipients to a defined batch. • Publication table: where the print stream output is stored. • Historical tables: matches the layout of the active processing tables named above. Used to support smart archive, useful for BI and other reporting analytics. • Transaction log table: an audit record of activities logged by applications. • Errors table: a common place for reporting processing errors occurring inside applications. • Log table: a common place for applications to send multi-level debugging information. Defined by the SQL provided during installation. Set up by a Database Administrator.
Web Application	**Description**
Documaker Administrator	Used to configure database connection information, hot folder locations, and other system assembly line and application configuration options.

Supervisor

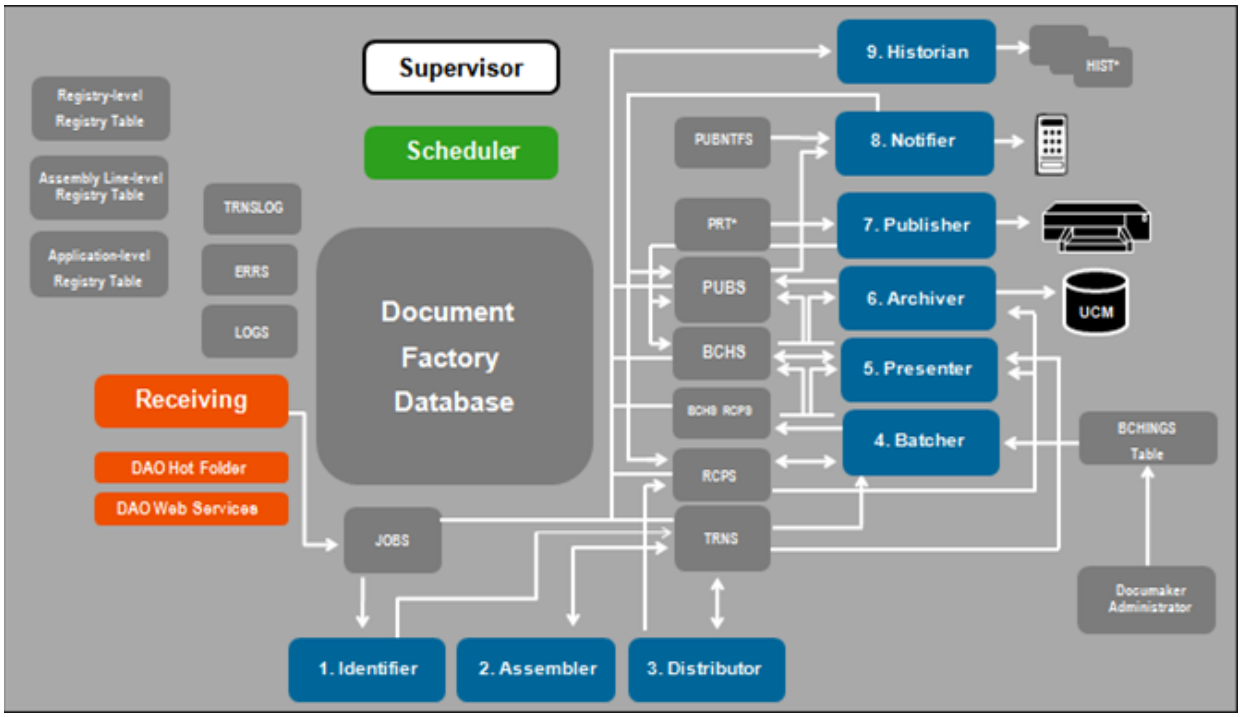


Figure 3: The Supervisor process in Document Factory

The Supervisor process, also known as the watchdog, is a light-weight, multithreaded Java process that acts as the manager of a given assembly line within the factory. There is, by design, one Supervisor per assembly line. Since it is a single point of failure, it is packaged to run as a service under Windows so it will be started up automatically and restarted if it fails.

The Supervisor runs the Scheduler process, the Receiving process, and monitors the presence and operability of the factory workers. It is the central process for running and balancing other processes in the document factory assembly line.

Note For more information, see *Using the Supervisor* on page 69.

Scheduler

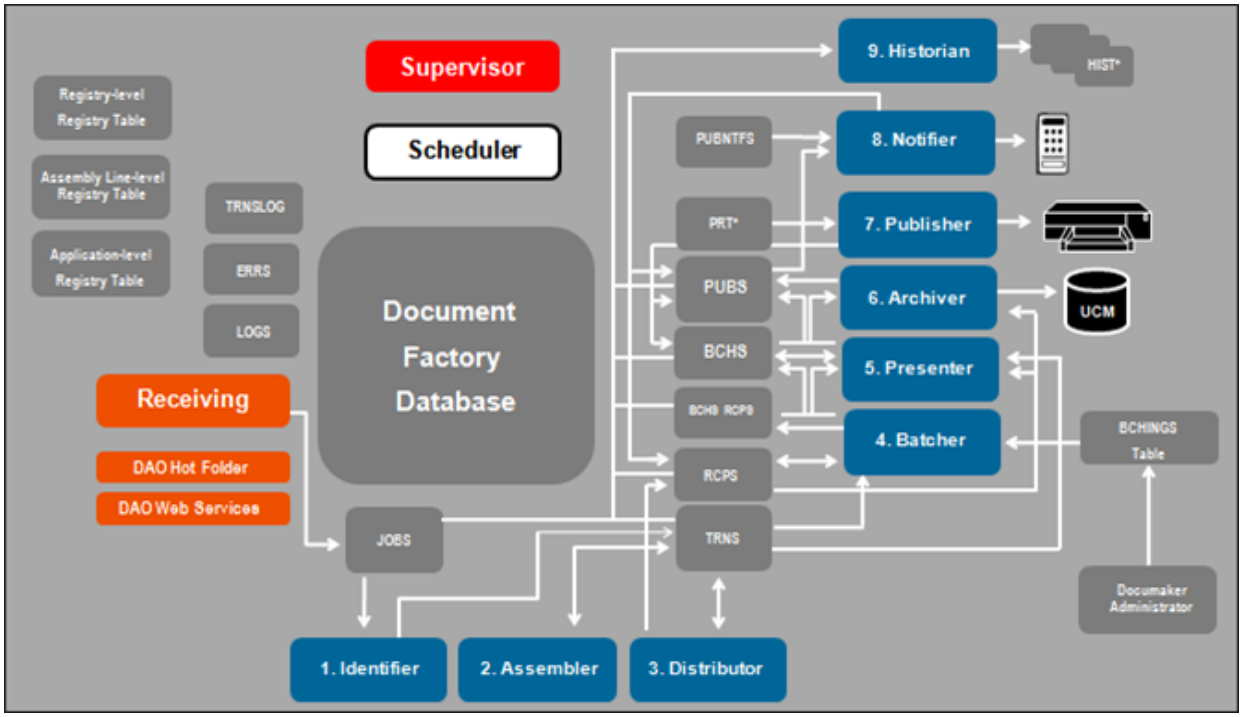


Figure 4: The Schedule process in Document Factory

The Scheduler is a Java process that monitors Document Factory processing tables and routes activity to worker component queues for processing. It watches one or more Documaker Factory tables and notifies different workers via a message bus that there is work ready to be processed.

Note For more information, see *Using the Scheduler* on page 94.

Receiver

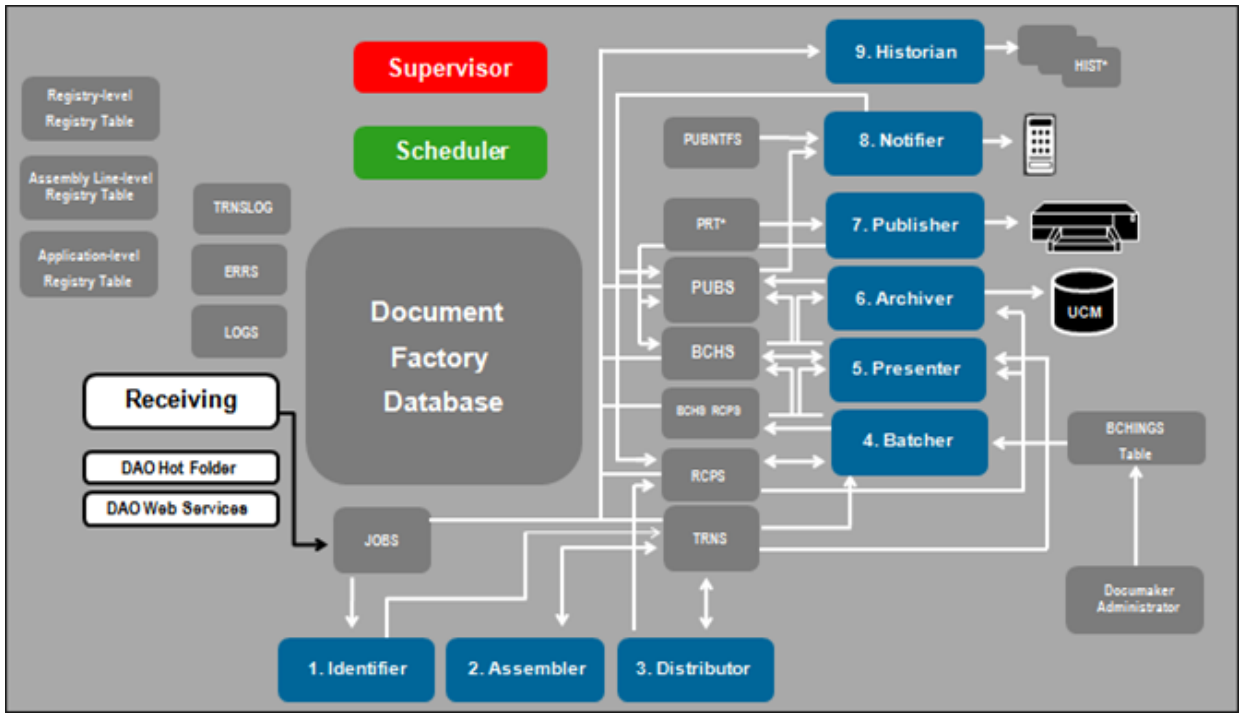


Figure 5: The Receiver component of Document Factory

The Receiver is a Java process responsible for accepting publishing jobs into the Document Factory. There are two methods of input into the Document Factory:

- DAO hot folder is where an input extract file can be manually placed. The hot folder (or multiple hot folders) is polled by the JobImporter which hands off the extract file to the Receiving process.
- DAO Web Services receives the input extract file and hands it off to the Receiving process.

The Receiving process reads the input file and converts it into an XML Job file that contains the extract data for the job. It updates the jobs database table, and the job status code so the Scheduler can process another job.

Note For more information, see *Configuring the Receiver* on page 132.

Workers

The worker components include the Identifier, Assembler, Distributor, Batcher, Presenter, Archiver, Publisher, Notifier, and Historian.

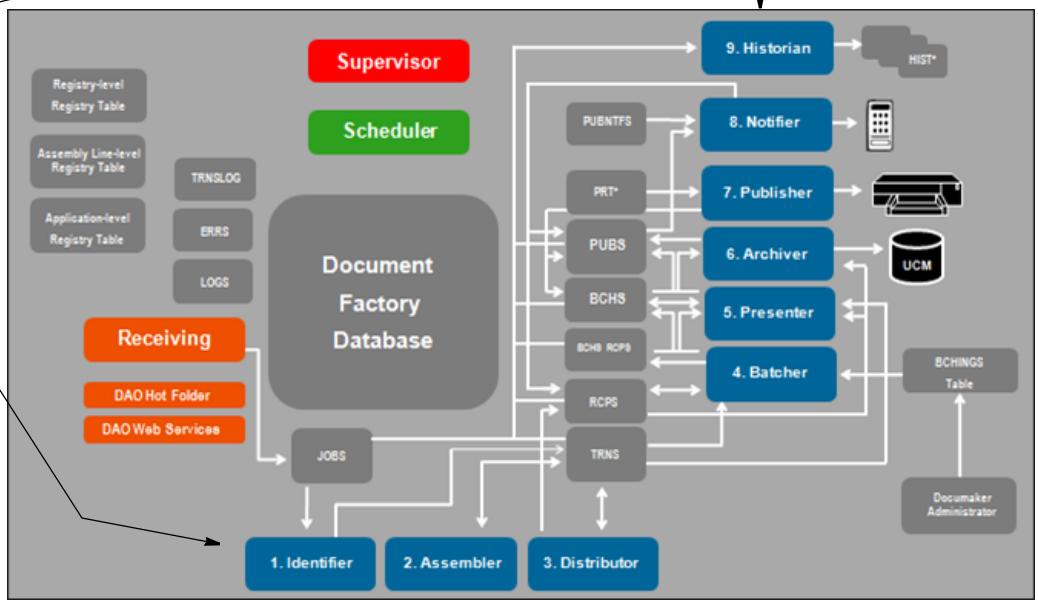


Figure 6: The Worker components of the Document Factory

This table provides an overview of the worker components of Document Factory.

Worker	Function
Identifier	Retrieves a job record from the JOBS table and breaks it into separate transactions. This functionality is similar to that which is performed by the GenTrn program in Documaker server processing. For more information see *Configuring the Identifier* on page 146.
Assembler	Processes extract data per transaction record, and creates an initial form set for the transaction, that includes triggered forms and mapped data.. For more information see *Configuring the Assembler* on page 157.
Distributor	Retrieves data for a transaction and distributes that data to different recipient records. For more information see *Configuring the Distributor* on page 174.
Batcher	Responsible for creating and associating batches with recipients. Notifies Presenter when batches are ready. For more information see *Configuring the Batcher* on page 191.
Presenter	Generates one or more print streams for the Document Factory. For more information see *Configuring the Presenter* on page 209.
Archiver	Submits each print stream for the batch to the configured archive destination (UCM, FTP, or file system) when Archive is enabled.
Publisher	Submits each print stream for the batch to the specified output destination. This includes the print and email distribution methods.
Notifier	Sends alerts to the recipients of a batch. Notifier can send SMS or email alerts in various formats, depending on how you configure it.

Worker	Function
Historian	Moves Assembly Line processing data from active processing tables to a corollary set of tables for retention and reporting.

DOCUMAKER ENTERPRISE EDITION WEB APPLICATIONS

Oracle Documaker Enterprise Edition includes the following web applications:

Documaker Administrator

The Documaker Administrator is the interface for controlling the Document Factory configuration at the system, assembly line, and application — or individual worker, level.

The Documaker Administrator also lets you configure user group permissions, which are used by the web applications, as well user approval levels, which are used within Documaker Interactive (assuming you are using Oracle Business Rules for approval workflow).

Documaker Interactive

Documaker Interactive is the interface you use to create and edit documents for distribution. Updates transactions in the Assembly Line that need further updates or editing and allows end users to prepare these transactions for distribution.

Document Factory Dashboard Overview

The Document Factory Dashboard is the interface for monitoring Document Factory processes. It displays a defined flow of information from job submission to document printing and archival. The Dashboard monitors the publishing system, providing opportunities to identify any issues during processing.

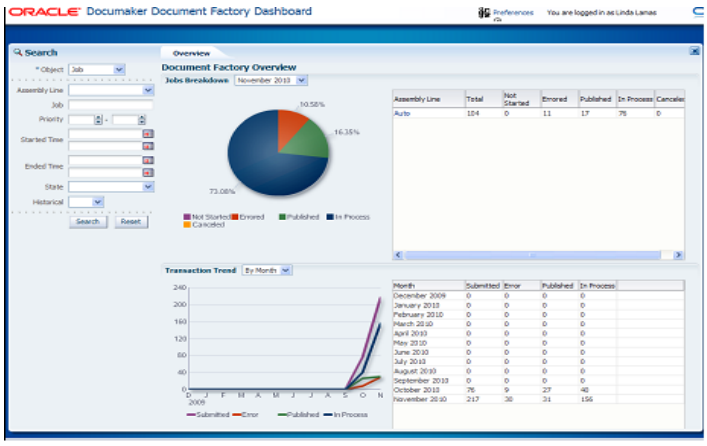


Figure 7: The Document Factory Dashboard main screen

The Dashboard has an object search facility that provides fast access to all objects within the factory, transaction and job metrics and analystics, and drill down views of jobs within the Document Factory.

Chapter 2

Using Documaker Enterprise

This chapter discusses the following topics:

SETTING UP AN MRL

A master resource library (MRL) is a collection of forms, field definitions, form lists, and key values associated with a set of documents to be produced in a publishing system.

Each MRL is unique to the particular implementation so it can support the needs of the business. The Documaker Studio Guide provides information on setting up an MRL for publishing. Keep in mind the following system features when you design an MRL for use with Documaker Enterprise Edition:

- Enterprise Edition lets you distribute document sets by email for a named user with specific contact information. The distribution process for email relies upon the presence of an email address in the ADR_EMAIL column of the RCPS table.

  This column is populated by addressee logic that applies address-specific information to an instance of a recipient identified in the MRL. If you want to use this functionality, apply an address map to a specific recipient within the MRL by updating the recipient within the Business Definition (BDF) file. You access the BDF file using Documaker Studio. For more information, see the Documaker Studio User Guide.

Note If your implementation does not rely upon address-specific information for distribution, and you are not using Documaker Interactive addressee processing, you do not need to apply an address map to a recipient in Documaker Studio.

- Enterprise Edition incorporates the concept of users of a transaction into both Documaker Interactive and Document Factory.

  When viewing or searching for transactions in Document Factory Dashboard, transactions are listed with a user if the resource library has been configured to recognize that user. There are various ways to associate a user with a transaction, a few examples are listed here:

  - If the source extract data has information/knowledge of the associated user, you can map this information into the TRNS table index using the TRNS fields.
  - If the source application has knowledge of the specific user, you can provide this information in the payload of a doPublishfromImport request.
  - You can specify the user ID in the CurrUser option in this FSISYS.INI control group:

    ```
    < AFG2WIP >
       CurrUser =
    ```

  In each case, the user value you provide must be the ENTITY_ID value for the user as stored in the ENTITIES tables within the dmkr_admin schema in the Document Factory database.

  The same user information is referenced in Documaker Interactive and is part of the criteria used when displaying documents on the different tabs of the application.

Documaker Interactive also has *unassigned transactions*, which means the transaction, or document, is available for editing by anyone in the designated group and the first named user to edit the document takes ownership. If no user or group information is associated with a document, the document is only available to users with the administrator ability set (not the Documaker Administrator users).

Users are also recognized as part of the components to evaluate when supporting approval based document distribution. See below for more information.

- Documaker Interactive provides a web-based user interface for creating, editing, and approving documents for distribution. The approval rules and workflow provided in the installed environment, also called the *reference implementation*, evaluate form metatdata against the submitting users approval level, for distribution approval.

  This means that if you want to use the default approval rules, you must assign approval levels to the document templates in Studio and then set up approval levels for the system's groups and users using the Documaker Administrator.

  To assign an approval level to a document template, check out the form in Documaker Studio, open the Metadata properties window, add the metadata name of Approval Level, and assign a numeric value to this metadata element. The reference implementation uses approval levels 1-4, but you can use as many or as few as needed. See the Adding Users topic for more information about assigning approval levels to users and groups.

  If the approval rules for a particular customer or customer MRL are not form or user dependant, but instead depend upon transactional data, there are two ways to approach the solution:

  - Update the Approval Level metadata for a form in the document during Assembly processing via a DAL script or custom rule. Set the approval level based on the transactional data value provided. Then, use the existing rules to check the form approval level with the submitting user approval level as provided in the installation.
  - Update the business rules to evaluate the transactional, or form set, data rather than the Approval Level metadata elements. The users and groups still have approval levels that can be used for evaluation in the updated business rule logic. For more information on modifying these business rules, see *Customizing Approval Business Rules* on page 34.

Note You can find examples of addressee maps and form metadata approval setup with the reference implementation for Correspondence resources which is accessible after the installation.

See also *Migrating to Document Factory* on page 507.

ADDING USERS

Users must be granted the necessary permissions to access and use the web applications associated with Oracle Documaker Enterprise Edition. The installation process creates an administrator user, named *Documaker*, who can add users and configure the application within the Documaker Administrator.

After installation, the *Documaker* user can log into the Documaker Administrator and perform the needed configuration activities, such as adding other users. Understanding these terms will help you understand the process of adding a new user:

Term	Definition
Entities	A user or a group of users that are identified to use an application.
Abilities	Types of tasks that an entity can perform.
Ability Set	A grouping of tasks/actions that an entity can perform. Also known as a role.
Approval Levels	Levels set up by administrators that define what degree of approval is required for the various documents and documents sets. This information is stored in an approval level metadata element that is applied to the forms in the MRL.

Note If a user without the needed ability set logs into Documaker Interactive: Correspondence, that user receives this error after authenticating into the system:

```
User has no permission.
```

Here are the tables where the entities and ability relationships are maintained. These tables are within the administration schema, named *dmkr_admin* by default.

Table	Description
DMKR_ENTITIES	The list of users and groups/roles that have been given access to the web applications.
DMKR_ABILITYSETS	Collections of permissions (abilities) that may be made available to web application users.
DMKR_ABILITIES	Individual permissions that pertain to application components (UI or functional)
DMKR_ENTITY_ABILITYSET	Information about which entities have access to which ability sets.
DMKR_ABILITYSET_ABILITY	Information about which abilities are members of which ability sets as well as the types of permission associated with each ability.
DMKR_ENTITYTYPES	The list of possible entity types: • 1 = User Group Entity • 2 = User Entity
DMKR_ABILITYTYPES	Stores a list of user-defined ability types. These represent an additional dimension for grouping abilities that can be linked to an ability set. This is not presently used by the application.

Setting up users and assigning permissions involves these steps:

1. Define the ability sets available for each web application. Ability sets define the various roles within an application that users may assume. These roles determine what abilities/permissions each user has.

   The system is installed with these ability sets, which apply to Documaker Interactive: Correspondence:

   - Drafter
   - Approver
   - Administrator

   The system is also installed with the *Documaker* administrator ability that defines the role for the user responsible for configuring the system via the Documaker Administrator web application. The ability set definition information is stored in these tables:

Table	Stores the
DMKR_ABILITYSETS	Ability set name
DMKR_ABILITYSET_ABILITY	Associated abilities.
DMKR_ABILITIES	Possible ability options.

   Use the ability set functionality to add or remove an ability set from the system. You can also use the ability set functionality to control the abilities and functions available within Documaker Interactive. This lets you set the functions and tabs available for each user role.

2. Manage Entities. The system uses Oracle Platform Security Services (OPSS) to get a list of possible user groups for the web applications. The Manage Entities function identifies those user groups that should have access to the Documaker Interactive, Documaker Document Factory Dashboard, and the Documaker Administrator web applications, and also associates the user group with one or more pre-defined ability sets.

   You do this using the Manage Entities tab in the Documaker Administrator. Select Add (+) to add a new entity. This action uses OPSS to query the user identity management application, and retrieves a list of user groups.

Note The user group must have a display name attribute in the identity management application to appear in the Add new entity list in the Documaker Administrator.

3. Select the group that you want to include as a known entity to the Documaker web applications.

   Entities known to the Documaker Administrator application are the enterprise users or groups. To link a user group with a set of abilities, first add the user group stored in the DMKR_ENTITY table. Then link the group to an ability set by adding and associated ability set. This association is stored in DMKR_ENTITY_ABILITYSET.

4. Define and Link Approval Levels. If your Documaker Interactive environment uses approval rules based on document approval levels, you must associate an approval level with a user group or individual user within a group.

   For any groups or users linked to the Approver ability set, you can associate a pre-defined approval level. First, create the approval level value and then link this value to a specific group or users in a group. During Documaker Interactive processing, this approval level is compared with document content (form metadata) of items submitted by drafters for distribution to determine the outcome of the submitted document.

   For more information the default approval rules provided with the system see *Understanding Documaker Interactive Validation and Approval Rules* on page 23.

By default, the system has four approval levels (1-4). On the Set Approval Levels tab, associate either a complete group or an individual user to an approval level. You can also define additional approval levels. These should be kept in sync with the document approval levels added to the master resource library (MRL) used for the Documaker Interactive application.

Note For more information on how to add document approval levels to the MRL, see *Setting Up an MRL* on page 14.

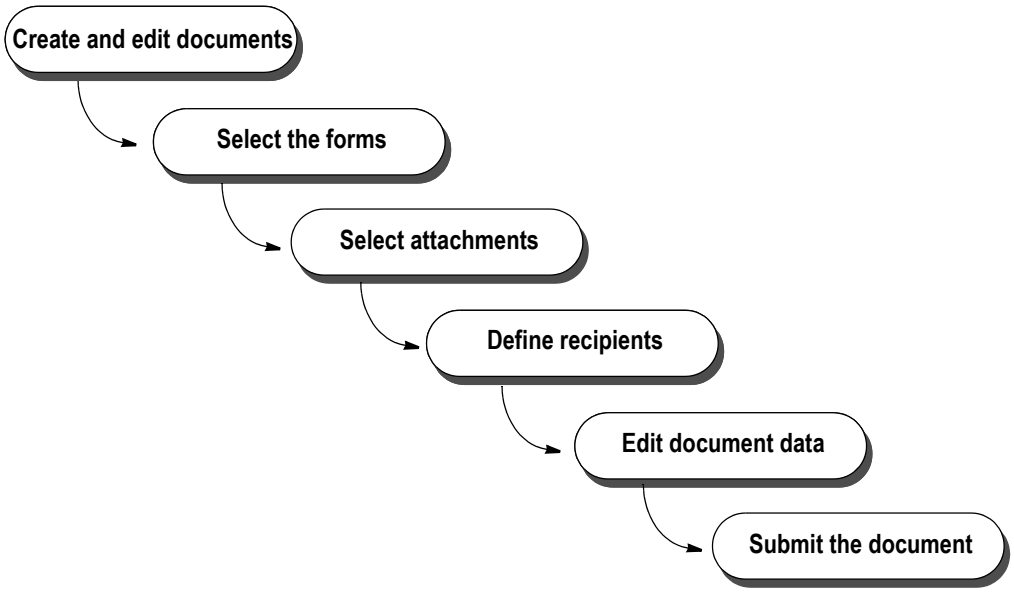
CONFIGURING THE WEB APPLICATION

Document Factory places documents into an interactive editing scenario via the Assembler. During Assembler processing, if a document requires editing, the manual processing indicators are set for the Status and Approval State values. These TRNS columns, *Status* and *Approval State*, are evaluated by Documaker Interactive to determine which transactions are visible to and accessible by Documaker Interactive users.

Note You can configure the values the Assembler sets for different scenarios using INI options, such as Assembler_StatusCode. For more information on these options, see *Using Assembler Configuration Resources* on page 159.

DOCUMAKER INTERACTIVE WORKFLOW

Documaker Interactive comes with this pre-defined work flow:



The installed application assumes end users want to perform these tasks:

- Create a new document
- Edit a document
- Preview a document
- Assign a document to a user
- Delete a document
- Add attachments to a document
- Submit a document
- Approve or reject a document - The approve and reject functions are defined for a group of users with a specific ability set.

You can grant or restrict access to these functions for each group of users using Documaker Administrator's Entities and Abilities functions. In Documaker Interactive, these functions are accessed on tabs that group transactions together by current status and approval state values.

Note Your implementation does not have to perform all of these steps and can enter this task flow at different points. For instance, your implementation could exclude attachments, or the ability to add and remove forms. It could also omit the ability to edit the document. or even define addressees. If addressee handling is omitted, however, your implementation would have to modify confirmation validation rules and distribution would have to be set up to work without the omitted information.

Viewing documents

The installed application automatically groups documents in the inbox and presents them to end users on these filtered tabs:

Tab	Description
My assignments	Lists documents assigned to the logged in user that are in Draft or Rejected approval state and require further editing.
Unassigned	Lists documents assigned to the logged in user's group without a named owner that are in Draft or Rejected approval state and require further editing.
Distributed	Lists documents that have been successfully approved and are in the distribution process or have completed the distribution process and have not yet been moved to the Historical set of Assembly Line tables.
Tracking	Lists documents created by the current user that are now awaiting the approval of another user.
For Review	Where the approve and reject document functions are defined for a group of users with the Approver ability set.

Note You can control access to these tabs and views for each group of users using Documaker Administrator. You can also customize these filtered tabs to meet the needs of your implementation.

Customizing the filtered tabs modifies the application rather than merely changes its configuration. This means future upgrades would also have to be similarly modified.

Customizing the display

The installed application also includes...

- Viewing detail tabs for a document displayed in the Inbox tabs on the inbox
- The ability to store and maintain form favorites

  You can grant or restrict access to these functions for each group of users using Documaker Administrator.
- Modify Inbox table column headings and application titles

These labels have pre-defined values to demonstrate their use within the reference implementation. You can customize these values for your implementation via the DMKR_TRANSLAT table in the administration schema. You can also use this table to add and control language-specific descriptions of these labels.

Documaker Interactive users would update the browser language setting to reference correct locale specific terms, if defined in the translation table. If a label is not defined for the selected locale, English is used.

Note The order of the columns and whether they are viewed or hidden is controlled within the code and the system would have to be customized to modify columns in this manner.

- Skins

  The color, font, contrast, and style of each page is controlled by the application's *skin.* A skin is a style sheet based on the CSS 3.0 syntax that is specified in one place for an entire application. All web applications within Documaker Enterprise Edition come with these skins:

  - Fusion
  - Fusion Projector

Note A developer can change the styles, icons, properties, and text of an ADF Faces component. For more information on how to add/modify a skin, please refer to the Oracle jDeveloper web site:

http://www.oracle.com/technetwork/developer-tools/jdev/overview/index.html

- Personal preferences

  The system also lets end users set preferences within the application that are specific to that user and web application. These preferences include:

  - Colors and contrast settings
  - Time zone for date/time display
  - Accessibility preferences

- Online help

  You can create online help specific to your implementation if desired. This Help could outline company procedures, provide reminders, or whatever information you feel is beneficial to the end user.

Applying business rules

The installed application uses the BPEL workflow engine backed by Oracle Business Rules to provide the following:

- Pre-built approval workflow, which you can customize as needed
- Single-or multilevel approval workflow
- Validation and approval of documents based on document data

You can, for instance, implement business rules to require approval when...

- An insured amount exceeds $100,000
- A specific user (or user group) drafts or submits the document for distribution

To do this, you can extend the Oracle Business Rules supplied with the installed application. For more information see *Customizing Approval Business Rules* on page 34.

UNDERSTANDING DOCUMAKER INTERACTIVE VALIDATION AND APPROVAL RULES

Documents generated as a part of the Documaker Interactive application must contain enough information to be distributed and may be configured to require approval prior to distribution. This topic reviews the document validation and approval process.

- Drafters create documents where they set the addressee information and update the document data. The first step in confirming that a document is ready for distribution from Documaker Interactive is validating that all required information is present.

  These elements are used to determine the completeness of the document:

Element	Description
Required fields	These fields only pass validation criteria in the client. (They are actually validated on the server by the RequiredFieldCheck rule in the Distributor process but this is much further in processing). A document's required fields are unique per form and implementation. Required fields established within Documaker Studio when the form template is created by the Library Administrator.
Addressee information	Validation is executed in both the client and the server.

  If the document does not contain the required addressee information, the submit and validation process rejects the document and sets the approval state to Rejected. The document will remain in the Drafter's inbox to be updated.

- Approvers must accept documents before those documents are distributed. Documents are routed to approvers based on the pre-defined business rules. These business rules evaluate the document's maximum approval level and compare it to the approval level of the user who submitted the document.

  If the Drafter or current approver's approval level is higher than or equal to the document approval level, the document can progress towards distribution. If not, the document is passed on to the next valid approver.

- If the document passed validation but did not get the required approval, the approval state is *Pending Approval*. The document appears in the Drafter's Tracking tab and on the designated Approver's For Review tab.

  If the document is ready for distribution, the approval state will be *Pending Distribution* and the document appears in the Drafter's Tracking tab.

The submit process — which initiates the validation and approval logic — is controlled by the Business Process Approval Language (BPEL), which is installed with the system.

Here is a list of the error codes you could see if the BPEL-based validation and approval process fails:

ID	Code	Details
35000	AP35000	Unknown request, such as an invalid request type was sent to the BPEL web service from Documaker Interactive: Correspondence.
35001	AP35001	Approval Error – the approval business rules returned an error.
35002	AP35002	Unknown Approver Type – the approval business rules returned a value for setting the CURRUSER or CURRGROUP that is not an Entity within the system.
35003	AP35003	Validation Error – the validation business rules failed to successfully validate the document data.

Assuming the client side addressee validation has passed, here is a description of what happens when the Drafter selects the Submit action.

1. Documaker Interactive: Correspondence locks the transaction by setting the InUse flag to *Y*.
2. Documaker Interactive: Correspondence updates the Action by setting the value to "4".
3. Documaker Interactive: Correspondence calls the BPEL web service.
   - If this call fails:
     - You receive this error:

       ```
       Unable to process your request. Please try again or contact
       your system administrator.
       ```

     - The transaction is unlocked, but the approval state remains unchanged — either Draft, Pending Approval, or Rejected.
   - If this call succeeds, continue to the next step.
4. The BPEL web service validates the request to approve or reject.
   - If this validation fails, the rules instruct the BPEL web service to update the following TRNS column values.

TRNS column	Value
ApprovalState	No change
InUse	N
ReasonID	10
Route_Desc	Invalid approval request type (not Approve or Reject)
Curruser	No change
Action	9 (BPEL update)

5. The BPEL web service then invokes a set of business rules to validate the data within the request to make sure the data is properly formatted and contains the needed addressee information.

- If this validation fails, the business rules instruct the BPEL web service to update the following TRNS column values.

TRNS column	Value
ApprovalState	40 (Rejected)
InUse	N
ReasonID	9
Route_Desc	
Curruser	No change
Action	9 (BPEL update)

- If the validation passes, continue to the next step.

6. The BPEL web service then invokes another set of business rules to validate the approval state of the transaction and determine if the transaction can be marked as *Pending Distribution*. The approval state of the document is compared with the current user and the current users' permissions and approval level.

   - If the document approval level is less than or equal to the current user's approval level, the business rules instruct the BPEL web service to update the following TRNS column values.

TRNS column	Value
ApprovalState	50 (Pending Distribution)
InUse	N
ReasonID	-
Route_Desc	-
ErrorID	"-" (Clear the last ErrorID in the TRNS table when the submission is successful.)
Curruser	ORIGUSER (Set back to the ORIGUSER value so it will show up on the correct user's Distributed tab.)
Action	9 (BPEL update)
STATUSCODE	"B"

   - Otherwise, if the document approval level is greater than the current user's approval level, the business rules instruct the BPEL web service to update these TRNS column values:

TRNS column	Value
ApprovalState	20 (Pending Approval)
InUse	N

TRNS column	Value
ReasonID	-
Route_Desc	-
CurrUser	User at next approval level
CurrGroup	Group of next user or next approver group
Action	9 (BPEL update)

- If there is an approval business rule error, such as the next approver user or group does not exist, the business rule instructs the BPEL web service to update the following TRNS column values:

TRNS column	Value
ApprovalState	Rejected
InUse	N
ReasonID	8 (No available approvers at next level)
Route_Desc	-
CurrUser	ORIGUSER
CurrGroup	ORIGGROUP
Action	9 (BPEL update)

- If the approval process returns invalid data (specifically an unknown approver type), the business rule instructs the BPEL web service to update these TRNS column values:

TRNS column	Value
ApprovalState	Rejected
InUse	N
ReasonID	11 (Invalid approver type)
Route_Desc	-
CurrUser	ORIGUSER
CurrGroup	ORIGGROUP
Action	9 (BPEL update)

UNDERSTANDING THE RULES LOGIC

An Oracle SOA Business Rules Decision Service Component is used by the approval BPEL process to determine the next state of the document that is submitted for approval.

This business rules component comes with a rules dictionary named *iDMkrApprovalRules.rules* which includes a default rule set named *CorrespondenceApprovalRuleset* for correspondence approval rules.

The rules take a form set XML (defined by "formset.xsd") that has the submitted *DOCUMENT* as input and returns a result XML (defined by approvalrulesresult.xsd).

The rules expect the following fields in input form set XML:

- An integer specifying a form's required approval level in its corresponding FORM element's INFO element with the name *Approval Level*. Here is an example:

  ```
  <INFO NAME="Approval Level">2</INFO>)
  ```

- System generated ENTITYID for the current user in CURRUSER element of WIPKEYS.

The result XML has the next state for the DOCUMENT as determined by the rules and any extra data required by that state. The next state is returned in the STATUS element of ApprovalRulesResult XML with one of these strings:

- Pending Distribution
- Pending Approval
- Approval Error

For the Pending Distribution state, there is no other associated data. For *Pending Approval*, the NEXTAPPROVER and NEXTAPPROVERTYPE element will have data for next approver. For *Approval Error*, the ERRORINFO element will have error information.

How the business rules determine the state of the document or TRNS record

The rules implemented in the default rule set support a multilevel approval process. They process the submitted document according to this procedure:

1. Determines the highest form approval level required for this "DOCUMENT" from the input XML form set data (from each "FORM" element's "INFO" element with the name "APPROVALLEVEL".
2. Determines the Highest Approver Level that the user specified in "CURRUSER" has. It uses the Documaker Interactive: Correspondence Abilities component and Documaker Interactive: Correspondence ApproverLevels component here to evaluate the approver abilities and approver levels (if any) assigned to the user and his groups.

3. Compares the required approval level (from step1) with the user's approver level (from step 2) to determine the next state for the “DOCUMENT” to return in the result XML as below:

    - Required Approval Level is zero or matches or lower than User Approver Level, “STATUS” is “Pending Distribution”
    - Required Approval Level is greater than User Approver Level, “STATUS” is “Pending Approval”, “NEXT APPROVER” is the ability system's ENTITYID for the next approver, and “NEXTAPPROVERTYPE” is “User” or “Group” if the next approver is a user or group respectively. The rules use the Documaker Interactive: Correspondence Abilities component and Documaker Interactive: Correspondence ApproverLevels component here to compile a list of users and groups that have approver abilities and that are assigned the next higher approver level than the current user's approver level. The next approver is determined from this list using a round-robin method and it can be either a user or a group.
    - Any error condition, “STATUS” is “Approval Error”, “ERRORINFO” is the error message for the error.

Note If there are no approver levels set up for users or groups in the system, then the rules treat all the users and groups that have approver abilities to have an approver level 1 implicitly. So any documents that require an approval level of 1 can be approved by these users or groups. But any documents that require an approval level greater than 1 will get an approval error saying that there are no available approvers at the next level.

Approver Abilities and Approver Levels

The business rules for approval process rely on the functionality of the Documaker Interactive: Correspondence Abilities component and Documaker Interactive: Correspondence ApproverLevels component to determine the approvers and approval levels. These components are ADF libraries that communicate with the Entities and Abilities tables within the Documaker Registry schema (dmkr_admin, by default).

These tables store data about users and groups and their corresponding abilities and approver levels. So a user or his groups must have approver abilities and be assigned to some approver level for the business rules to recognize them as an approver or a next approver. The Documaker Administrator application is used by the administrators to assign such capabilities to the users or groups.

Sample Users, Groups and Abilities

To illustrate the approval process and for testing, use the following tables as reference for users and groups, Documaker Interactive abilities and their assigned approval levels, and the business rules result column in the last table that shows the result for each sample use case:

Enterprise Identity Repository

User	Groups
Alan Abrams	Associate Typist

User	Groups
Bob Babbit	Associate Typist
Clive Chan	Typist
Debra Delaney	Typist
Emily Evans	Typist
Frank Fish	Typist, Mgr
Gilbert Gold	Typist, Mgr
Hilda Hinton	Mgr
Ian Ivanoff	Mgr
Jake James	VP
Karen Kane	VP
Linda Lamas	IT

Documaker Interactive Abilities

Entity (User or Group)	Ability set
Associate Typist	Drafters
Typist	Drafters
Mgr	Approvers
VP	Approvers
IT	Administrators
Service	Print Preview Approver
Marketing	Print Preview Administrator

Documaker Interactive Approver Levels

Entity (User or Group)	Approver level
Frank Fish	1
Gilbert Gold	1
Mgr	2
VP	3
Karen Kane	4

Sample Scenarios and Approval Business Rules Result

Current user *	Document approval level +	Rules result
Alan Abrams	1	STATUS: Pending Approval NEXTAPPROVER: Frank Fish NEXTAPPROVERTYPE: User
Frank Fish	1	STATUS: Pending Distribution
Jake James	1	STATUS: Pending Distribution
Bob Babbit	2	STATUS: Pending Approval NEXTAPPROVER: Gilbert Gold NEXTAPPROVERTYPE: User
Karen Kane	2	STATUS: Pending Distribution
Ivan Ivanoff	3	STATUS: Pending Approval NEXTAPPROVER: VP NEXTAPPROVERTYPE: Group
No user specified	1	STATUS: Approval Error ERRORINFO: CURRUSER is not given
Unknown user (for example, 100)	1	STATUS: Approval Error ERRORINFO: Failed to find user with ID: 100
Karen Kane	5	STATUS: Approval Error ERRORINFO: No Approver available at next required approval level: 5

* Referenced by the entityID in CURRUSER element in the submitted DOCUMENT

\+ The required maximum approval level is defined in the submitted DOCUMENT

Approval levels assigned to sample forms in the Correspondence MRL

Form	Approval Level
AM-472	1
AM-GBL	2
AM-LI-845	3
AM-LI-9642	4
CG 00 01	5
CG 20 04	1
CG 20 12	2
CG 20 15	3
CG 21 00	4
CG 21 04	5

Form	Approval Level
CG 21 45	1
CM FM SCHED	2
DESTINY POLICY PAGES	3
EAPPLICATION	4
FS 20 NY	1
FS 20 TX	2
GFORMS PROOF OF LOSS COVER LETTERS & ATTACHMENT	3
HO 00 01	4
IL 00 21	1
IL 00 54	2
LI-128 NY	3
LI-153	4
LI-450	1
LI-473 01-2004	2
LI-529	3
LIFE WELCOME	4
LOCATIN DETAIL	5
LOCATION SCHEDULE	1
M-576	2
M-577	3
MI-2876	4
MK-9576	1
ORD-87698 TX	2
OVERFLOW EXAMPLE	3
PA 00 01	4
PA 01 00	1
PA 09 00	2
PA 10 00	3
PA 11 00	3

Form	Approval Level
PA 33 00	1
PA-5921	2
PROOF OF LOSS	3
SUBLOCATION	1
TIFFINCLUDE	2
UL APPLICATION	1
UL APPLICATION REJECTION NOTICE	1
UL APPLICATION RESPONSE	1
UM 00 00	2
VUL FL-B 01-2004	3

REPLACING THE APPROVAL PROCESS

Documaker Interactive: Correspondence defaults to using the BPEL Approval Decision Service but you can use the urlText property in the APPCONFIGCONTEXT table to specify the optional pass-through service if you prefer to use a service that approves every document submitted for distribution.

To use the pass-through approval process, use the Documaker Administrator to modify the Correspondence application, BPEL_CLIENT_DATA group (in the BPEL_CLIENT_DATA Category and the WORKFLOW Context) settings as shown here:

Property	Description
urlText	This property supplies the URL for the pass-through service. Here is the value for the pass-through service: `http://ip:port/BPELPassthroughService-BPELService-context-root/CorrespondenceProcesses_pt?WSDL` The default is to use the BPEL Approval Decision Service.

The BPEL approval process uses the decisionService value in the configuration to identify the Oracle Business Rules in place and the documakerService to identify the DWS location for calling IDS to update the TRNS index.

The pass-through approval process only uses the documakerService information as it has pre-defined values for updating the TRNS index.

CUSTOMIZING APPROVAL BUSINESS RULES

The business rules defined for the approval process can be edited after deployment (at run time) by business analysts or administrators according to their custom implementation.

Only users with the SOADesigner application role can access the metadata from SOA Composer. By default, all users with the WLS Administrator privileges have this role. WebLogic server's Enterprise Manager (Fusion Middleware Control) can be used to assign this role to any additional users or groups.

The rules dictionary iDMkrApprovalRules.rules can be edited at run time using a web-based tool called SOA Composer that is available with the SOA Suite.

The SOA Composer's URL is

http://soa_ server_host:soa_ server_port/soa/composer

The open rules menu item can be used to browse for all rules dictionaries or the direct URL for the "iDMkrApprovalRules.rules" dictionary in revision 1.0 of the deployed SOA composite can be used which will be

http://soa_ server_host:soa_ server_port /soa/composer?docPath=/deployed-composites/iDMkrApprovalRulesProj_rev1.0/oracle/rules/oracle/documaker/idocumaker/apprrules/iDMkrApprovalRules.rules

Refer to the SOA suite's documentation for further details on the SOA composer for editing rules (Oracle Fusion Middleware User's Guide for Oracle Business Rules).

If you make changes to the deployed rules dictionary, you must manually copy this file into your JDeveloper project if you want to have the changes persist in the next deployment of the SOA composite.

To access the rules dictionary file from a deployed composite, the SOA composite must be exported before undeploying it. The exported archive will contain the changed rules dictionary file (iDMkrApprovalRules.rules) which can be reused in a new deployment of the SOA composite.

Refer to the SOA suite's documentation for further details on exporting a running SOA composite ("Exporting a Running SOA Composite Application" of Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite).

ENABLING UCM

Documaker Interactive provides lets you include attachments with a document. These attachments can come from the local file system or from Oracle's Universal Content Management (UCM) system.

Use Documaker Administrator to enable the UCM attachments tab via the Add UCM Attachment Ability. You can also use the Attachment show, Attachment list, Attachment Add, and Attachment Delete abilities to fully configure the functionality.

When you enable UCM attachment capabilities with Documaker Administrator, you update these values in the APPCONFIGCONTEXT table:

Group	Property	Description
WIP_ACTION_ADD	UCM_IdcConnection	Enter the UCM Connection String – or the idc://ucm server:port value needed to locate UCM.
WIP_ACTION_MODIFY	UCM_IdcConnection	Enter the UCM Connection String – or the idc://ucm server:port value needed to locate UCM.
UCM_CONNECT	connectionString	Update the host name from the default value of localhost.
	passWord	Enter the password for the UCM user.
	userName	Enter the user name assigned to the user who has access to UCM and the appropriate permissions.

When you finish, restart the idm_server.

Note This is only necessary if UCM was not enabled during the installation.

The default configuration for UCM expects these fields as index values in the UCM repository. Review these values with your UCM administrator and add or remove any values in the UCM destination Default mapping table if they differ. This is done by checking these values in the in the Documaker Administrator, Archiver application configuration:

- dDocAccount
- dDocAuthor
- dDocType
- dSecurityGroup

UNDERSTANDING BATCHES

Document Factory provides a wide range of batching and sorting options so you can arrange and distribute documents in the most desirable, cost effective, efficient, and personalized manner.

Documaker provides rules which you can use to indicate how each recipient of a document should be placed into a recipient batch (BCH) record. Common options include by recipient, such as INSURED, AGENT, HOME OFFICE, or by data element such as document or transaction type, such as NB, RN, invoices, and so on.

Document Factory adds additional capabilities by letting you re-batch or re-group the INI-designated batches, called *recipient batches*, into more specific batches.

HOW BATCHES ARE DETERMINED

There are two sets of criteria evaluated during the Document Factory batching process. The first criteria, defined in the FSISYS.INI file, is usually based on recipient but it could be any criteria.

The initial batch name names, defined in the FSISYS.INI file, are also referenced as the batch groups in the BCHINGS definition within the Document Factory.

The reference implementation defines one FSISYS.INI file batch, called *BATCH1*, as the initial or batch group. BATCH1 is the name of the recipient batch (or parent/initial batch) in the BCHINGS table defined in Document Factory. From this batch group, additional criteria assigned in the BCHINGS table further segregate batches for distribution.

For Documaker Interactive: Correspondence and the sample BCHINGS created by the Document Factory installation process, the criteria used is the distribution type associated with the specific *addressee* of the recipient. Since the system is using the distribution type associated with the addressee, the master resource library (MRL) must support having an addressee recipient to capture the addressee data.

When the extract file is processed, the specific information in the extract data is mapped to an element in the NA file content — called the Addressee record. (Alternatively, this information can be indicated by user selection in Documaker Interactive, but either way the MRL needs to have a recipient defined with an Addressee record.)

Typically, you would have one recipient identified in the MRL's BDF file and that recipient is designated in the BDF to support Addressee processing, but Document Factory processing can also accommodate other recipients defined for use in the MRL.

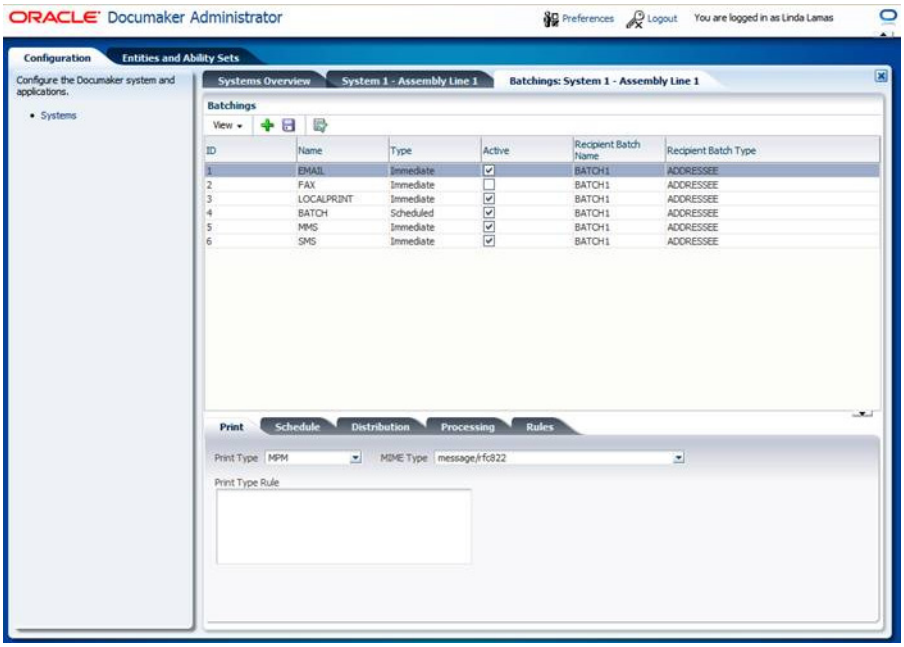
DEFINING THE BATCH

The batches you define in Document Factory do not have to be based on distribution type. When you install the system, these batch definitions are set up, based on the distribution method:

Batch definition	Type	Print type
Email	Immediate	MPM
Local Print	Immediate	PDF
Mail	Scheduled	PCL
Fax	Immediate	XMP
SMS	Immediate	PST
MMS	Immediate	XER

The default batch definition is Immediate (Immediate/PDF).

A batch definition can either be active or inactive. If inactive, all other scheduling options are ignored as the batch is not considered as available for recipient processing.



Each batch listed in Document Factory has an associated *recipient* batch. This is the name of the originating or parent recipient batch defined in the FSISYS.INI file, or the initial batch name for the recipients of the transaction.

Each Document Factory batch has an associated recipient batch type. This recipient batch type is used to match a recipient record and a Document Factory batch. The recipient batch type associated with a recipient record can be one of these types:

Type	Description
Addressee	Choose this type if the ADR_INDEX value is greater than zero (0). In other words, this is the value of the Recipient Batch Type if the Recipient has an associated Addressee.
Standard	Choose this type if the ADR_INDEX value equals zero (0) and the following configuration value is not set.

Type	Description
An INI value, configured by the Distributor	This type is defined by the StandardType option in the DocFactory control group in the FSIUSER3.INI file.

So, for each recipient, the recipient is placed in an initial batch based on the Recipient Batch Name and criteria and further evaluated by the Recipient Batch Type.

If you are using addressee processing for the recipient and the ADR_INDEX value is greater than zero, then the ADDRESSEE type will be matched. Otherwise, the system assumes the Recipient Batch Type is Standard. If, however, the following entry is in the FSIUSER_2.INI file, this value is also checked for inclusion into the Factory batch:

```
< DocFactory >
   StandardType = value
```

Where value can be a literal or a dynamically determined value such as a GVM from the TRNS table layout.

The batching process first evaluates the recipient batch type to identify a match. If the type matches, the system evaluates the recipient batch name.

Document Factory batches also have an associated type which designates the timing by which the Presenter process closes the batch and generates the print stream for the records in the batch at that time. This type can be either *immediate*, as soon as the recipient is placed into the batch, or *scheduled*, which indicates the print stream will be generated at a later time.

This option lets you control when batches are sent to archive, processed into print streams, emails, or notifications. By default, only the mail batches are set to be scheduled, but you can change this if necessary.

Defining the Print Type

Each batch must have either a defined print type or a rule that enables the print type to be set *on the fly* per recipient. If you want the incoming job or transaction data to identify the output type for a specific print type, map that data into a GVM that results in a column associated with the job, transaction, or recipient in the Assembly Line. Then you use the Print Type Rule field to identify the column that contains the print type value. Here is an example of the syntax:

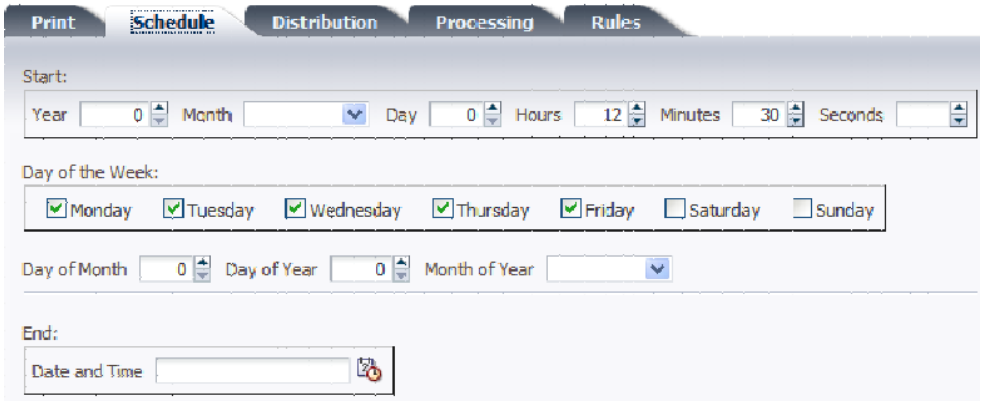
```
RCPS.ADR_DISTRIBUTION
```

Note Options for each print type value, such as PDF, PCL, and AFP, are controlled in the FSISYS.INI file. For more information on these options and values, see the Documaker Server Administration Guide.

The Presenter process generates the print stream and creates an entry in the PUBS table to store the resulting output. The system can store this output in various format, or MIME types, depending on the print type. A default MIME type is associated with each print type, but you can overwrite this value to choose a different storage format.

SCHEDULING A BATCH

The system accumulates recipient transactions in a batch until the batch is closed is closed, based upon the information that you enter on the Schedule tab. Publishing is also controlled by data that you specify on the Schedule tab:



There are several ways to control when a batch is closed and when publishing starts. This is a three step process:

1. Indicate the effective and expiration dates for the batch
2. Set the time or frequency
3. Indicate the week and/or date that the batch should process

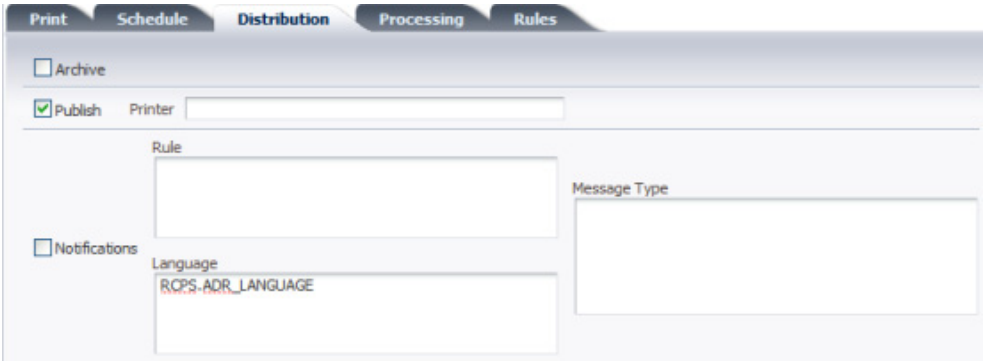
To	Then	Description
Schedule the closing of a batch	Use the Start and End fields	Use (Start) Year, Month, and Day fields to identify the effective date for the batch. Use the (End) Date and Time field to identify the expiration date for the batch. Use this option to set the last date on which the batch can be published. Note: If these fields are blank or zero (0), the batch is effective immediately.

To	Then	Description
Begin publishing a batch	Specify the time or frequency in which the batch is published with the Start Hours, Minutes, and Seconds fields.	Hours – specifies the time, based on a 24-hour clock, at which the batch should be published. Midnight to 1am is represented with a zero (0). If this field is blank, each hour is eligible for publishing and the Minutes field is used. Minutes – specifies the time, based on 60 minutes in an hour, at which the batch should be published. If Hours is blank, the system will begin processing the batch at the specified minute of each hour. For example, to have a batch published every hour, on the hour, leave Hours blank and set Minutes to zero (0). Seconds – specifies the seconds, based on 60 seconds in a minute, at which the batch should be published. Note: If you leave the Hours, Minutes, and Seconds fields blank, the system processes the batch immediately — assuming no other criteria is in place.
	Specify the day of the week the batch is published with Day of the Week fields.	Note: This is used with the Start Hours, Minutes, and Seconds fields that control the time or frequency of the batch publishing.
	Specify a date on which you want the system to publish the batch using the Day of the Month and Month of the Year fields (or the Day of the Year field).	Note: The Day of Week and Date options are independent of one another. Each option creates a valid entry in the batch schedule. For example, if you specify October 30, 2010, a Saturday, and set the Day of the Week field to Monday, the batch will publish on Saturday, October 30, 2010 and on Monday, November 1, 2010.

Setting the Distribution Options

While the Scheduling options control when the batch will be processed, the Distribution options let you define how the result will be distributed. For instance, you can do the following:

- Send PDF files of what each recipient will receive to archive
- Send the batch to a printer
- Email the batch
- Send notifications to recipients
- Specify a message to send to recipients



Use the properties to define how the batch will be distributed:

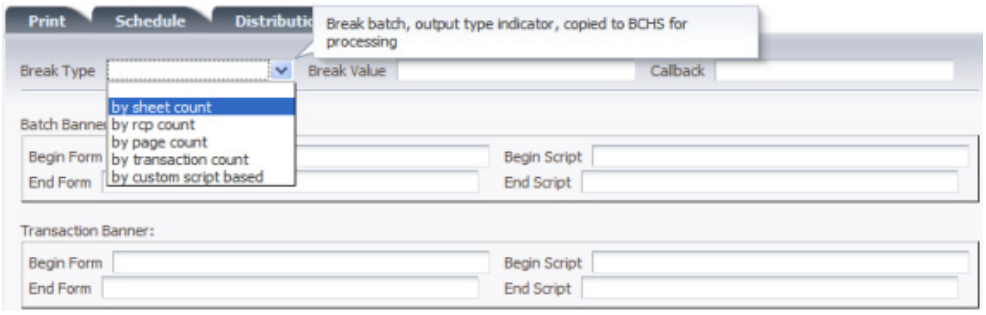
Field	Definition
Archive	To send a PDF version for each recipient identified in the batch to the archive destination (such as UCM), check this field.
Publish	To send the print stream or email to the designated printer, check this field.
Printer	To specify a group of printers if you need to direct a batch to a given printer or type of printer. Use the Printer field, for example, to specify a particular printer who's paper stock is configured appropriately for the output in the batch.
Notifications	By default, the system sends notifications for those recipients in the SMS and MMS batches. You can also enable notifications for other batches using the Notifications field.
Rule	Use this field to further identify specific recipients who should receive a notification by specifying a column or select statement to choose the particular recipients.

Message Type	Use this field to identify the SMS notification message template that is further defined in *PUBNTFMSGTYP* on page 49. This template lets you apply fields from the RCPS, JOBS, or TRNS tables, to the template. This lets you not only say
	`Your document is now available from www.oracle.com.`
	but also to say
	`%RCPS.ADR_NAME% your document is now available from www.oracle.com.`

Language	Use this field to specify the column of RCPS data that identifies the end recipient's preferred notification language. The default is English but if other language templates are set up in the PUBNTFS table, the system uses your preferred language.

Controlling the Batch Size and Banner Pages

Use the Processing tab to control the size of the batch.



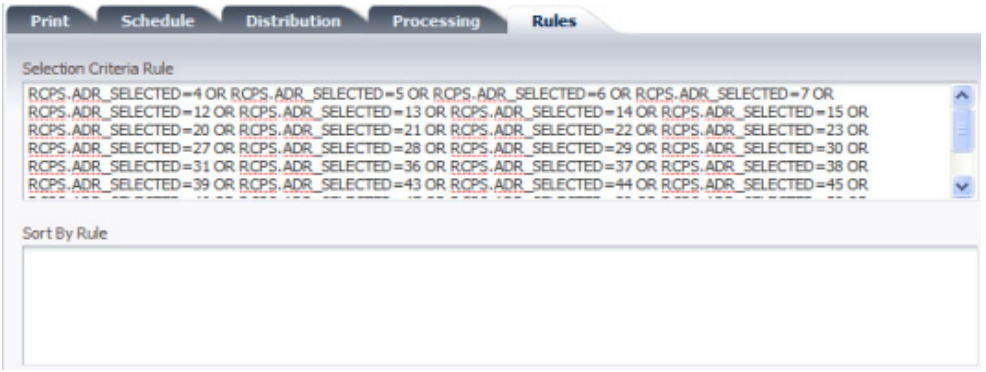
You have a variety of options:

- Number of sheets
- Number of recipients
- Number of pages
- Number of transactions
- Custom control
- Callback function

Use these options, along with a specific value, to divide large print streams into smaller, more manageable files. This helps balance the load across printers and makes the files more manageable if you need to reprint a particular set of data.

Including Recipients in a Batch

Once you set up the INI options that establish the initial recipient batch criteria, Document Factory batches each recipient using the batching rules you set up in the Documaker Factory Administrator. The pre-configured batches are defined by distribution type. The distribution is associated with a given recipient of the document or transaction. The distribution values are defined on the Rules tab.



Use the Selection Criteria Rule field to set the criteria for associating a given recipient with a particular print batch. The criteria can select from any given column defined in the RCPS or TRNS tables. This example shows the criteria for placing a recipient in the email batch.

Note These values are mapped into the ADR_SELECTED column by the use of the Addressee map via extract data, using the XDD, or populated into the field from Documaker Interactive: Correspondence, for a given recipient.

You can identify the criteria for these batches using any of the columns in the TRNS or RCPS tables. These columns are defined by the dmrk_asline DDL. The data for these columns comes from the corresponding GVM values. The GVM values are defined by:

GVM	DFD	INI	Group
TRNS	TRNDFDFL.DFD	FSISYS.INI	DATA
RCPS	RCBDOCF.DFD	FSIUSER_2.INI	DATA

The data is populated into the TRNS GVMs from the extract data as defined in the TRN_Fields control group of the FSISYS.INI file.

Data for the RCPS values are mapped into the GVMs from the Addressee record layout mapped in the XDD or obtained by data entered in Documaker Interactive: Correspondence.

Selected Distribution	Value in ADR_SELECTED
None	0
BATCH	1
LOCAL	2
BATCH,LOCAL	3
EMAIL	4
EMAIL,BATCH	5
EMAIL,LOCAL	6
BATCH,LOCAL,EMAIL	7
MMS	8
MMS,BATCH	9
MMS,LOCAL	10
MMS,BATCH,LOCAL	11
MMS,EMAIL	12
MMS,EMAIL,BATCH	13
MMS,EMAIL,LOCAL	14
MMS,BATCH,LOCAL,EMAIL	15
SMS	16

Selected Distribution	Value in ADR_SELECTED
SMS,BATCH	17
SMS,LOCAL	18
SMS,BATCH,LOCAL	19
SMS,EMAIL	20
SMS,EMAIL,BATCH	21
SMS,EMAIL,LOCAL	22
SMS,BATCH,LOCAL,EMAIL	23
SMS,MMS	24
SMS,MMS,BATCH	25
SMS,MMS,LOCAL	26
SMS,BATCH,LOCAL,EMAIL	27
SMS,MMS,EMAIL	28
SMS,MMS,EMAL,BATCH	29
SMS,MMS,EMAIL,LOCAL	30
SMS,MMS,BATCH,LOCAL,EMAIL	31
FAX	32
FAX,BATCH	33
FAX,LOCAL	34
FAX,BATCH,LOCAL	35
FAX,EMAIL	36
FAX,EMAIL,BATCH	37
FAX,EMAIL,LOCAL	38
FAX,BATCH,LOCAL,EMAIL	39
FAX,MMS	40
FAX,MMS,BATCH	41
FAX,MMS,LOCAL	42
FAX,BATCH,LOCAL,EMAIL	43
FAX,MMS,EMAIL	44
FAX,MMS,EMAIL,BATCH	45

Selected Distribution	Value in ADR_SELECTED
FAX,MMS,EMAIL,LOCAL	46
FAX,MMS,BATCH,LOCAL,EMAIL	47
FAX,SMS	48
FAX,SMS,BATCH	49
FAX,SMS,LOCAL	50
FAX,SMS,BATCH,LOCAL	51
FAX,SMS,EMAIL	52
FAX,SMS,EMAIL,BATCH	53
FAX,SMS,EMAIL,LOCAL	54
FAX,SMS,BATCH,LOCAL,EMAIL	55
FAX,SMS,MMS	56
FAX,SMS,MMS,BATCH	57
FAX,SMS,MMS,LOCAL	58
FAX,SMS,BATCH,LOCAL,EMAIL	59
FAX,SMS,MMS,EMAIL	60
FAX,SMS,MMS,EMAL,BATCH	61
FAX,SMS,MMS,EMAIL,LOCAL	62
FAX,SMS,MMS,BATCH,LOCAL,EMAIL	63

Determining the sort criteria

Use the Sort By rule to identify the columns in the RCPS table that determine the sort criteria for the printed output. This assumes the printed output can support more than one recipient at a time.

For example, this applies to such print types as PCL, AFP, or Metacode where output for multiple recipients would be in one print stream, typically scheduled batches, but would not apply to email output – as only one recipient email is generated at time.

Additionally, columns from any Assembly Line processing table that has a link to the RCPS table can be used in the sort criteria.

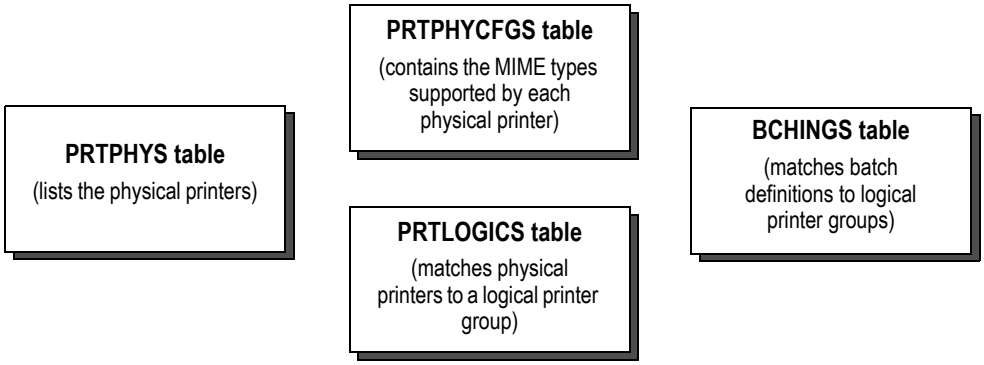
The Sort By rule supports a *table.column* name reference with an ascending/ descending indicator. You can also sort multiple columns.

SETTING UP PRINTERS

When you start Document Factory, it detects the available printers available and recognizes the print, or MIME types, each printer supports. The available printers are stored in the PRTPHYS assembly line table, also known as the physical printers table. The MIME types each physical printer can support are stored in the PRTPHYCFGS table.

By default, each detected printer is linked to a logical printer or group of physical printers. Typically, a group of printers might be those you want to support a certain type of output. However, the system installs with a single logical printer, called *Default_Printer*, and all detected printers are associated with this logical printer.

Using Documaker Administrator, each batch definition is assigned to a logical printer group to help direct output to a specific printer or set of printers. This assignment of a logical printer group to a batch definition is stored in the BCHINGS table.



Each of these tables is evaluated by the Publisher when it distributes print streams to the correct output device. The Publisher first checks the BCHINGS information to determine the logical printer group associated with the print stream. The logical printer group provides a list of possible physical printers or output devices for the Publisher to distribute the print stream.

The Publisher then compares the print stream MIME type to the MIME types supported by each of the physical printers in the logical printer group:

If a match is	Then the Publisher
Found	Sends the print stream to the matching physical printer or output device.
Not found	Emits an error, updates the PUBPUBSTATUS associated with the PUBS record, and writes an error to the ERRS table.

When you install a new printer on the server, you must restart Document Factory for the printer to be recognized by the system. During restart, Document Factory applies the MIME types that the printer communicates via a standard protocol. If the printer is configured to use another protocol, not all MIME types the printer can support are recognized.

After adding a new printer, use the Documaker Administrator to confirm the printer is set up correctly by performing these steps:

1. If you have more than the DEFAULT_PRINTER available, in the Printers for the Assembly Line move the Physical Printer to the desired Logical Printer reference.
2. Make sure the supported MIME types for the new printer are correct.
3. Make sure the printer is active and that its status is set to Printer Ready or 621.
4. Specify the sequence in which the Publisher should route documents to the printer based on the other printers within the logical printer group.

In summary, to set up a printer, you perform these steps:

1. Install a printer.
2. Restart Document Factory.
3. Use Documaker Administrator to check the printer configuration for the Assembly Line. Make sure the new printer is recognized by the system and that it's status is active. If not, it may be that the printer is defined for a different user than the owner of the server or that it does not make itself known by the standard protocol.

Note The printer can be defined but inactive if the PRTPHYSTATUS value is set to anything other than 621. To enable the printer, make sure this column value is set to 621.

4. Check the PRTPHYCFGS records for the new printer to confirm the correct MIME types were associated. If not, add the needed rows for the printer ID. The MIME types are:
   - application/afp
   - application/pdf
   - application/vnd.documaker-vipp
   - application/vnd.documaker-xer-barr
   - application/vnd.documaker-xer-barrword
   - application/vnd.documaker-xer-jes2
   - application/vnd.documaker-xer-mrg2
   - application/vnd.documaker-xer-mrg4
   - application/x-pcl
   - application/xml
   - image/gif
   - image/png
   - image/tiff
   - message/rfc822

5. Check the PRTLOGICS configuration to associate the Printer ID with a logical printer group. To add a new logical printer group, add a new record to this table. These logical printer groups can be linked with a batch definition, stored in the BATCHINGS table, when you create or edit a batch definition using Documaker Administrator.

Publishing to a Printer or Output Destination

The Publisher is responsible for sending print output to the logical printer associated with the batch configuration. This destination, can either be a printer or an email server. The Publisher recognizes the printers or email servers available within the PRTPHYSCFG (to confirm) table and writes to them based on the Publisher Group settings within the plug-in context.

The following plug-ins are included with the system. These plug-ins define the Java class used to direct printed output to a logical printer.

Plug-in	Description
EmailPublisher	The EmailPublisher class initiates when the PUBS row being printed for a given batch matches the rfc22 MIME type. In this case, the EmailPublisher configuration options identify the email server and connection information needed to *publish* or send the email
PrinterPublisher	The PrinterPublisher class initiates when the PUBS row being printed for a given batch matches the MIME type listed for the PrinterPublisher.

Here are the details of the configuration options in the EmailPublisher configuration group:

Options	Description
PublisherPlugin	References the EmailPublisher. The Host and Sender are not used.
EmailServers	References the email transport class used by the system to send email. *Do not* change these settings.
SMTPEmailServer	References the host name associated with the Assembly Line configuration option. The SMTPEmailServer - Email Publishing options route email to a defined server used across the Assembly Line.

To change the SMTP email server, follow these steps:

1. Update the AL configuration for SMTPEmailServer to the SMTP host.
2. Specify the from email address or sender of the emails from the Document Factory system.

Note If you need to store the print stream to disk prior to delivery, use the Archiver file system destination to write out a copy of the archive document to disk or set up a printer on the application server and direct the printer's port to print to file.

Note that Oracle does not recommend writing the print files, PUBS, to disk as these files are stored in the database and distributed with ODEE processing.

UPDATING PUBLICATION NOTIFICATION TEXT

Document Factory lets you notify a document recipient that their publication is available for viewing online. Once the publication is created and archived or stored in a user accessible location, the publication notifier generates either an SMS or email message to tell the user that the document is ready for viewing.

The notification process first evaluates the notification rule set within the batch configuration. If blank, this rule uses the ADR_SELECTED value to determine if the RCPS should receive a notification. The choices are Email, SMS, or MMS. In other words, if only batch mail was selected, just enabling the notification on the batch would not initiate the distribution. The ADR_SELECTED value must also indicate that a recipient should receive a distribution or this rule should be modified.

Once it is determined that a notification should be send, the PubNotifier is configured to either:

- UseEmailForSMS
- UseSMSService

The EmailProvider option is used in both cases. Lastly, the Language and Message Type batch configuration options are used to pull from the Publication Notification template definition to send the correct notification message to the recipient in the desired format.

The SMS notification message format is determined by the provided template. The template is stored in the Publication Notification Setup Table (PUBNTFS) and can be edited directly in the database table. Data available to the Publication Notifier, specifically data in the RCPS record, is available for use to personalize the SMS notification.

If you are using addressee-based recipient processing, each addressee can have a preferred language. Assuming that a unique language template is available in the PUBNTFS table, the Publication Notifier selects the specific template associated with the addressee's preferred language.

The PUBNTFS table contains these columns:

Column	Description
PUBNTFMSGTYP	The Publication Notification Message Type key. This lets you map the data elements to the message template.
PUBNTFLANG	The language of the template. This is used to select the appropriate language for the recipient of the message. The default is *EN* for English if the appropriate language can not be found.
PUBNTFSUBJ	The subject line template for email notifications.
PUBNTFTXT	The simple text template for SMS or text emails.
PUBNTFHTML	The HTML format of a notification for MMS or HTML emails.
PUBNTFSPEECH	The text to speech for notifications.

SELECTING THE LANGUAGE

You can modify the language used to display Document Factory, Document Factory Dashboard, Document Factory Administrator, and Documaker Interactive: Correspondence by setting the browser's language setting:

In this browser	Choose this option
Mozilla Firefox	Tools, Options, Content, Languages
Microsoft Internet Explorer	Tools, Internet Options, Appearance, Languages

Note The list of available languages depends on the product and resource translations made available to your implementation.The default language is English.

GENERATING CUSTOM REPORTS

The Document Factory's robust schema and use of XML for interim document formats means that generating reports or data to analyze Document Factory activities is as easy as defining the reporting criteria, as the information is readily available when needed.

While the Dashboard provides extensive search capabilities, you can always run queries against the database tables to pull information when necessary. For example, if you need to determine the count of a certain form that was generated within a particular time period, you can locate all the transactions in that range by querying the TRNS table start and end time ranges, and then parse the TRNNAPOLXML data for the desired form names.

Note This assumes that you are using the default option to store NA/POL information in XML format. Alternatively, you could parse the TRNSNAPOLXML data for trigger names to identify the exact criteria or reason that triggered each form.

See *Configuring the Assembler* on page 157.

The Assembly Line schema also predefines several columns in the TRNS table for tracking customer specific data elements. Using these fields provides a way to capture, process, and query on data elements unique to an organization.

The Documaker Administrator provides a facility for reporting on the configuration data maintained in the registry schema. Data at all levels of the registry; system, assembly line, and application, can be viewed in the Documaker Administrator and from there a report generated that snapshots the information in the tables at the time the report is run.

Generating a Registry Data Report

To generate a report of data in the Registry or Administrative tables of Document Factory, first locate the level of data need within the Documaker Administrator. Then click the Show Printable Page for Table Data printer icon to generate the report.

CUSTOMIZING DOCUMENT FACTORY

Document Factory supports a wide array of document automation capabilities, however, you may find there are times when a particular request or implementation requirement causes you to customize the system. The following information outlines certain scenarios or methods for customization.

Setting Custom GVM Values

Modifying the transaction index values or GVM variables used for batch processing, archive index values, data population on banner pages.

The TRNS table contains placeholders for customer-specific GVM values. In Oracle Documaker Standard Edition processing, the GVM fields are referenced in the TRNDFDFL.DFD and RCBDFDFL.DFD files. In Oracle Documaker Enterprise Edition, these fields are defined in the TRNS.DFD file and are contained in the TRNS table within the Assembly Line schema. To use customer-specific GVM placeholders, you must perform these steps:

1. Uncomment the fields in the TRNS.DFD file.
2. Populate the fields with data. To populate the fields with data you can either:
    - Use the TRN_FIELDS control group in the FSISYS.INI file to populate the fields with data from the extract file.
    - Update the TRNS index values via Web services, either using the doCallIDS method, when updating an existing document, or via the doPublishFromImport method, which provides a way to set both the Job and TRNS index values when a job is submitted.

Note Do not modify the other DFD files provided with the reference implementation. These files are integrated into the assembly line schema. Additional fields and columns will not be recognized.

Modifying the Form Set Data

You can modify the form set data several ways:

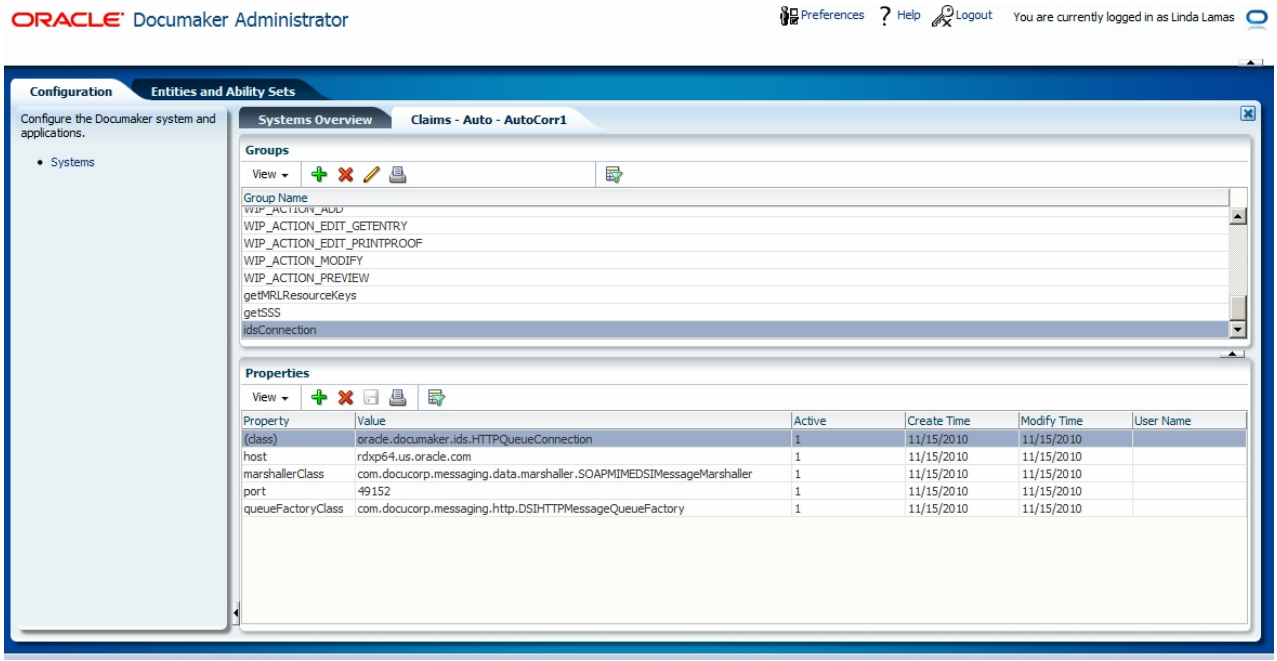
- Using pre- and post-transaction processing DAL scripts.
- Using a custom AFGJOB_1.JDT rule. You can use a custom rule to update field information, add forms, change graphics, and so on, however, keep in mind that the NA/POL information is no longer stored on disk or in native format by default but instead is store in XML in a BLOB within the TRNS table.

MODIFYING THE QUEUEING APPLICATION

Documaker Interactive: Correspondence sends requests to Docupresentment (IDS) and receives results. You use the Documaker Enterprise Administrator to set up Documaker Interactive so it can communicate with Docupresentment.

This example uses the *AutoCorr1* application. Log into the Documaker Enterprise Administrator using an account with administrator privileges. Click the Systems link. In the table that appears to the right, you will see an entry for *Claims* in the System column.

Click the icon to the left of *Claims* to expand the list of Assembly Lines. Click the icon to expand the *Auto* Assembly Line. Pick the *AutoCorr1* application, then click the Configure button. Scroll in the list of group names until you see the *idsConnection* group. You should see a screen similar to the one below.



The example screen shot shows that Documaker Interactive is currently set up to communicate with IDS using the HTTP protocol, although the default installation for WebLogic will use JMS queues. Other protocols are available.

To	Then
Change a setting	Double-click on the setting's row and change the Property and Value fields
Create a new property	Click the Create New Property icon (a green +) to add a new row to the table
Delete a property	Select the property and click the Delete Selected Property icon (a red X).

Using HTTP Queues

To use HTTP queues for communications between Documaker Interactive and IDS, use these options and values:

Option	Description
Section Name = SYSTEM_IDSHTTPConnection	
(class)	This is the implementation class or the class that will do the work. For HTTP, you would use this: oracle.documaker.ids.HTTPQueueConnection
host	Defines the IDS host. The default is localhost.
marshallerClass	Defines the IDS marshaling class. This example uses the SOAP marshaller over HTTP: com.docucorp.messaging.data.marshaller.SOAPMIMEDSIMessageMarshaller
port	Defines the default IDS listening port.
queueFactoryClass	Defines the queuing mechanism. This is the default for HTTP: com.docucorp.messaging.http.DSIHTTPMessageQueueFactory
The (class) and marshallerClass options are case sensitive and must be entered exactly as shown.	

Note For more information on the HTTP settings, see the *Using HTTP* topic in the Internet Document Server Guide.

Using WebLogic JMS Queues

The base configuration includes a default WebLogic (WLS) JMS connection. This provides an example of how to hook the system into IDS using WebLogic JMS as its messaging mechanism.

Option	Description
Section Name = SYSTEM_IDSWLSJMSConnection	
(class)	Defines the implementation class for the WLS JMS configuration: oracle.documaker.ids.WebLogicJMSConnection Do not change this value.
initialContextFactory	Defines a method to load during initialization. This lets Documaker Interactive load all the configuration data when you boot the application: weblogic.jndi.WLInitialContextFactory Do not change this value.
inputQueueName	Defines the name of the input queue. This should be defined in the WLS application server's JMS configuration.
The (class) and marshallerClass options are case sensitive and must be entered exactly as shown.	

Option	Description
marshallerClass	This connection implementation uses a SOAP implementation class for its marshaller: com.docucorp.messaging.data.marshaller.SOAPMIMEDSIMessageMarshaller Do not change this value.
outputQueueExpiry	Defines the length of time that should pass, in milliseconds, before an IDS request should expire.
outputQueueName	Defines the name of the JMS output queue. This should be defined in the WLS application server's JMS configuration.
providerURL	Defines the WLS JMS lookup location. Here is an example: `t3://localhost:7001` Replace localhost and the port with your WLS instance or cluster end point.
queueConnectionFactoryName	Defines the name of the WLS connection factory you are using for your queues.
securityCredentials	Defines the WLS security credentials, if needed.
securityPrincipal	Defines the WLS user name, if needed.

The (class) and marshallerClass options are case sensitive and must be entered exactly as shown.

Note For more information on the JMS settings, see the *Using the Java Message Service* topic in the Internet Document Server Guide.

Using WebSphere MQ Queues

There is also a default WebSphere MQ connection in the base configuration. This provides an example of how to hook Documaker Interactive into IDS using WebSphere MQ as its messaging mechanism.

Option	Description
Section Name = SYSTEM_IDSWebsphereMQConnection	
(class)	Defines the implementation class for the base system configuration: oracle.documaker.ids.MQSeriesConnection Do not change this value.
inputQueueName	Define your MQ server with an input queue that matches this name.
outputQueueExpiry	Defines the length of time that should pass, in milliseconds, before an IDS request should expire.
poolingEnabled	Determines if this queue allows pooling. The default is No.
inputPoolSize	Defines the size of the input pool. The default is 10.
outputPoolSize	Defines the size of the output pool. The default is 10.

The (class) and marshallerClass options are case sensitive and must be entered exactly as shown.

Option	Description
inputQueueMaxWait	Defines the maximum wait time, in milliseconds, for the input queue. The default is 5.
marshallerClass	This connection implementation uses a SOAP implementation class for its marshaller: com.docucorp.messaging.data.marshaller.SOAPMIMEDSIMessageMarshaller Do not change this value.
host	Defines the location of the MQ server. The default is localhost.
queueChannel	Defines the queue channel to use.
queueManager	Defines the queue manager to use.
tracing	Determines whether tracing is allowed. The default is No.
exceptionLogging	Determines whether exceptions are logged. The default is No.
outputQueueName	Define your MQ server with an output queue that matches this name.

The (class) and marshallerClass options are case sensitive and must be entered exactly as shown.

Note For more information on WebSphere, see the *Using WebSphere MQ* topic in the Internet Document Server Guide.

INCREASING THE SIZE OF THE DATAFILE

By default Documaker Enterprise Edition creates a Datafile for the Assembly Line's Tablespace that has a maximum size of 2GB. This limit prevents a demo system from using all of the disk space.

For a production system, you need to remove this limitation. You can use Oracle Database Enterprise Manager to remove this limitation:

To	Enter
Increase the size limit	`ALTER DATABASE DATAFILE 'C:\ORACLE\ORADATA\IDMAKER\DMKR_ASLINE.DBF' AUTOEXTEND ON MAXSIZE 4096M`
Remove the size limit	`ALTER DATABASE DATAFILE 'C:\ORACLE\ORADATA\IDMAKER\DMKR_ASLINE.DBF' AUTOEXTEND ON MAXSIZE UNLIMITED`

STARTING AND STOPPING PROCESSING

The following table provides links to detailed instructions on starting and stopping the various Document Factory processes:

To start or stop the	See
Supervisor	*Starting and Stopping the Supervisor* on page 79
Scheduler	*Starting and Stopping the Scheduler* on page 105
Receiver	*Starting and Stopping the Receiver* on page 133
Identifier	*Starting and Stopping the Identifier* on page 147
Distributor	*Starting and Stopping the Distributor* on page 175
Batcher	*Starting and Stopping the Batcher* on page 195
Presenter	*Starting and Stopping the Presenter* on page 213

You will also find information on how to verify that a process is running.

Chapter 3

Configuring Document Factory

This chapter provides the information you need to set up or modify how Document Factory performs in your implementation.

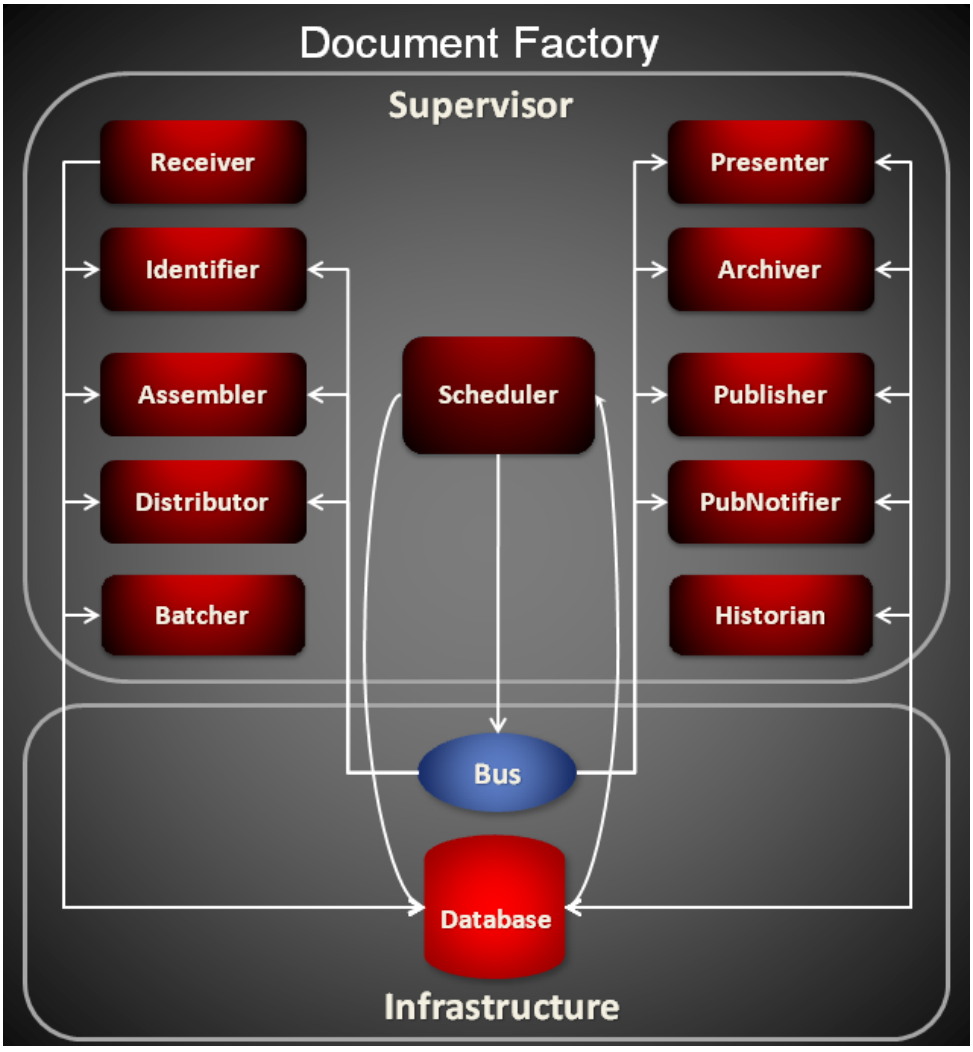
This chapter discusses these topics

OVERVIEW

The Automated Document Factory (ADF) is a processing model for creating and delivering mission-critical documents. ADF equates concepts of factory production to document production by integrating template design, data input and transformation, delivery preparation, and response management within a document publishing environment.

Within Oracle Document Factory, assembly line workers submit jobs to produce documents. Assembly line activities are monitored by the Supervisor process, which controls the worker's status. The Scheduler proces passes work from process to process using a message bus to alert each process that work is ready.

All activity in Document Factory is stored and monitored in a database. The use of a database and queues enables scalability, failover, and enhances reporting capabilities. The other processes compose and assemble documents at different stages using the database.



UNDERSTANDING THE DATABASE

The backbone of the Document Factory infrastructure is the database. The Document Factory database contains two primary schemas:

- Administration schema
- Assembly line schema

In general, the Administration schema is responsible for maintaining configuration details about the system, the assembly line, and the applications or workers on the assembly line. The Administration schema also stores the users, groups, and permissions for the web-based applications surrounding the Documaker Document Factory.

The Assembly Line schema maintains the job processing activity, including error and logging activity. The Assembly Line also has a set of historical tables that you can use to maintain processing data long after processing has completed. The following illustrations show the tables and keys of each table within both schemas of a Document Factory.

Note For information about logging to the database, see *Logging to the Database* on page 273.

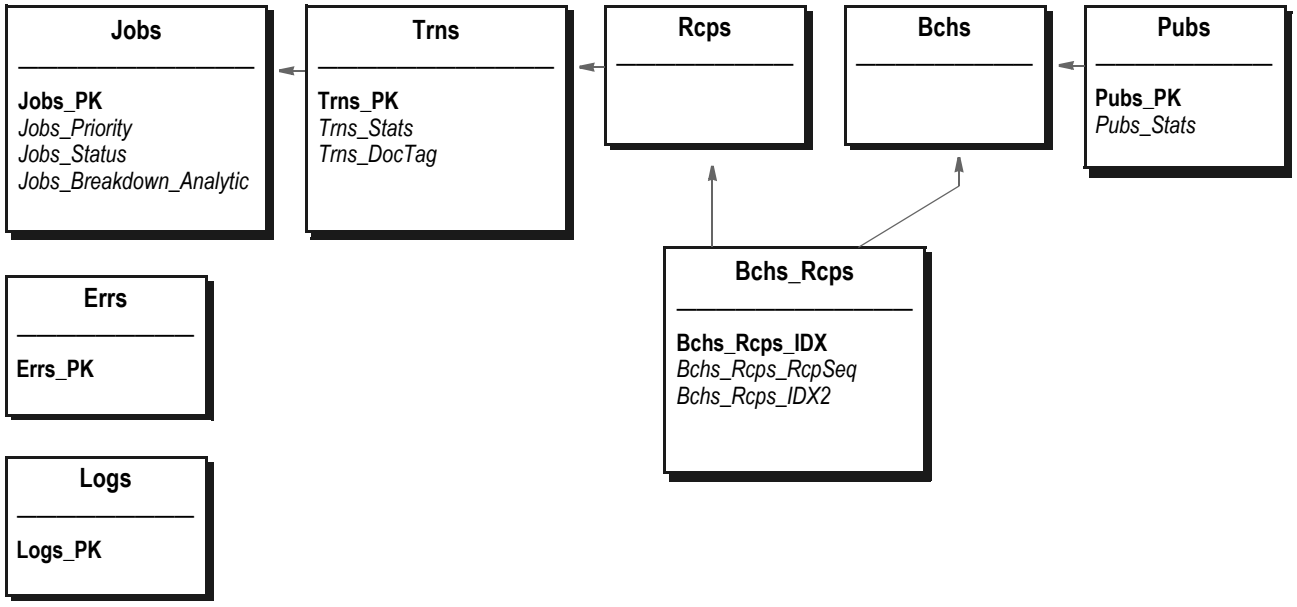


Figure 8: Document Factory Assembly Line Schema

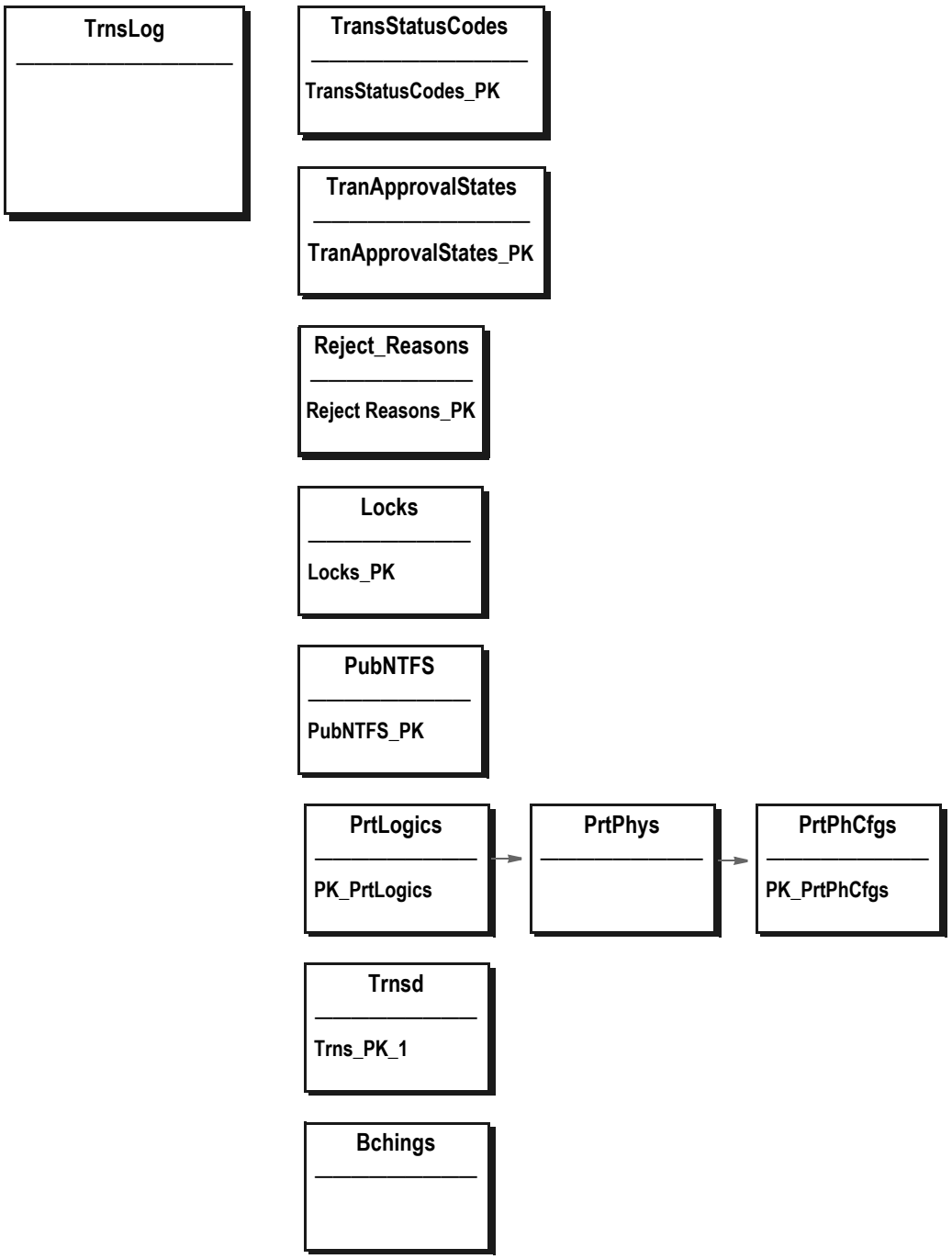


Figure 9: Document Factory Assembly Line Schema (continued)

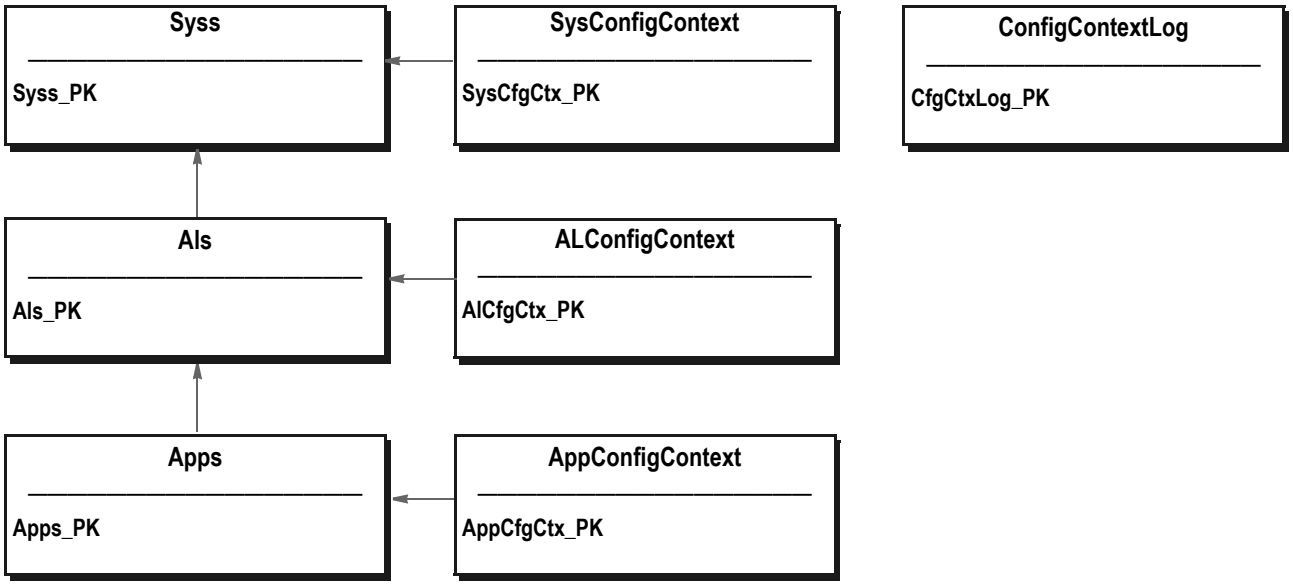


Figure 10: Document Factory Registry Schema

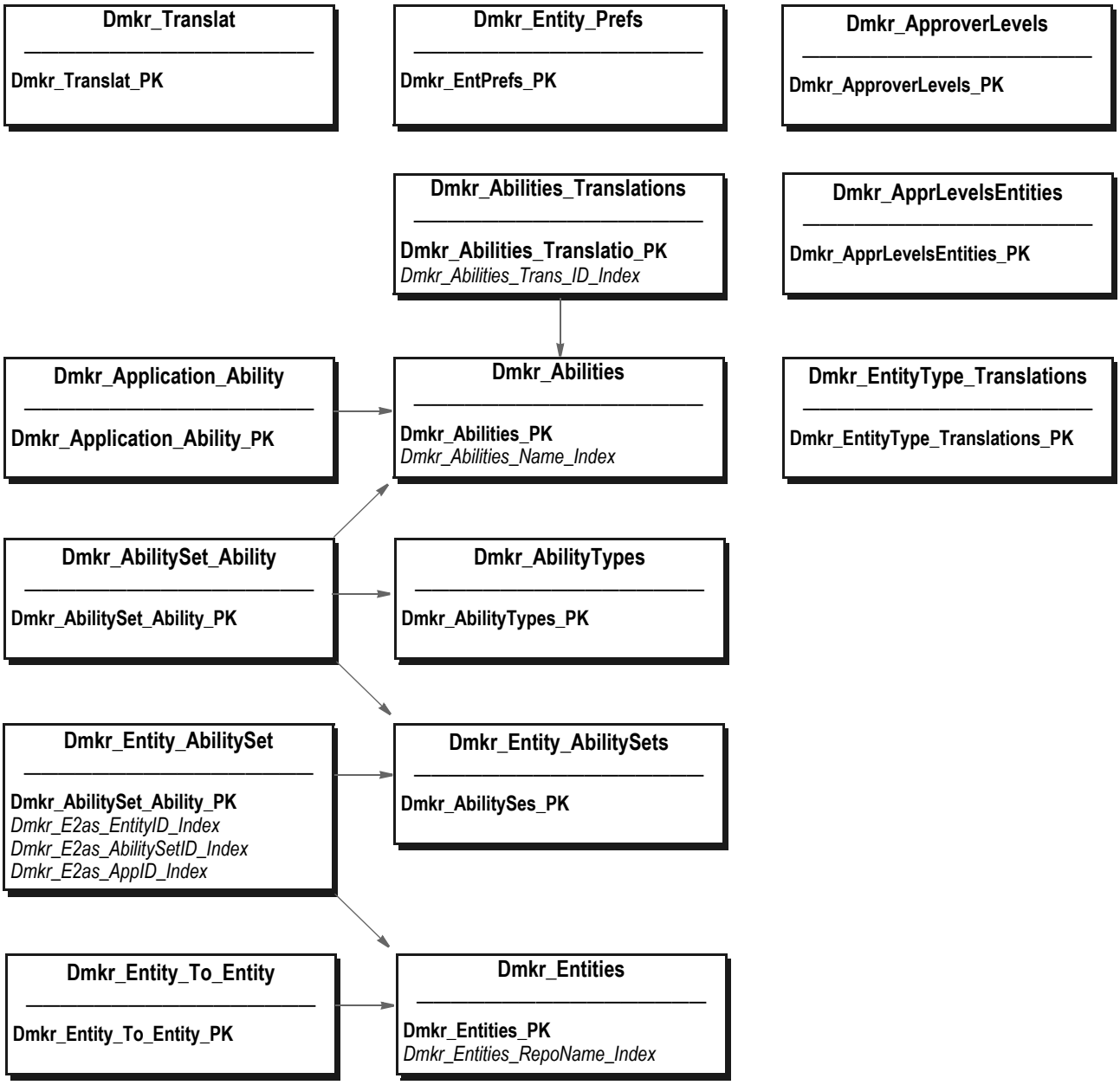


Figure 11: Document Factory Registry Schema (continued)

DEFINING THE CONFIGURATION

You configure and maintain Documaker Enterprise applications using a graphical web interface. No editing of the deployable (EAR) file, or the files within it, is necessary. This interface controls a basic structure that applies to all Documaker Enterprise applications:

Section Name	Options	Values	Description
idsConnection			The *Section Name* represents the name of the area of Documaker Interactive you want to configure, such as IDS connection information. For example, you could put the default connection information in a section named *idsConnection*.
	Host	locahost	Within each section, you can have multiple options which are specific to that section. *Host* and *Port* are examples of options needed to define connection information. The values for the Host and Port options could be something like *localhost* for Host and *49152* for Port.
	Port	49152	

Note See *Configuring Document Factory* on page 61 for complete information on Document Factory configuration options.

Here is an example IDS connection configuration:

Option	Description
Section Name = idsConnection	
(class)	Defines the implementation class or the class that will do the work. You can plug in different implementations, depending on the connection you have, such as HTTP, JMS, MQ, and so on. The default is oracle.documaker.ids.WebLogicJMSConnection.
host	Defines the IDS host. The default is localhost.
marshallerClass	Defines the IDS marshaling class.
port	Defines the default IDS listening port.
queueFactoryClass	Defines the queuing mechanism. This is the default for HTTP: com.docucorp.messaging.http.DSIHTTPMessageQueueFactory

Note that every section has a class to run it, which is specified in the *(class)* property. This lets you easily change at run time how the system behaves. Keep in mind that if you change the class, that new class will expect a new set of options.

For example, to use a WebLogic JMS connection, you would put these options into the configuration:

Option	Description
Section Name = idsConnection	
(class)	Defines the implementation class. Here is a WebLogic JMS example: oracle.documaker.ids.WebLogicJMSConnection
inputQueueName	Defines the input queue name. This is required for a WebLogic configuration. Here is an example: jms/resultq
queueConnectionFactoryName	All JMS queues need a connection factory. This is required for a WebLogic configuration. Here is an example: jms/IDSConnectionFactory
outputQueueExpiry	Defines, in milliseconds, the timeout interval.
marshallerClass	Defines the message marshalling implementation. Here is an example: com.docucorp.messaging.data.marshaller.SOAPMIMEDSIMessageMarshaller
initialContextFactory	Defines the JNDI lookup implementation class. Here is an example: Weblogic.jndi.WLInitialContextFactory
providerURL	Defines the address of the JMS listeners.
securityPrincipal	Defines the user name.
securityCredentials	Defines the user's password.
outputQueueName	Defines the name of the output queue. Here is an example: jms/requestq

This example shows the section name (idsConnection) is the same, but since the implementation class is different, the parameters needed for this functional area differ from the HTTP connection implementation.

USING THE SUPERVISOR

The Supervisor is a Java process that deploys/undeploys, starts/stops, and manages other Java or C processes. It is responsible for managing and monitoring all the processes that run in a Document Factory assembly line.

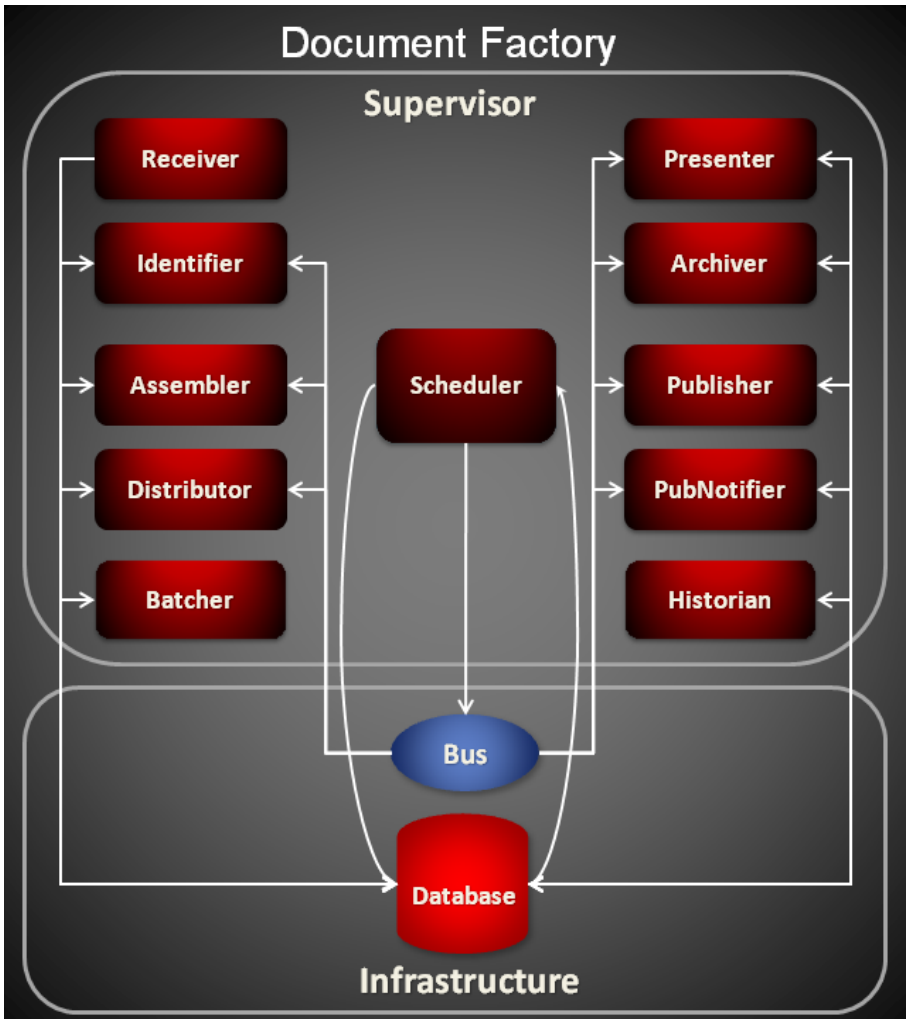
The Supervisor also has JMX capabilities to monitor the health of the Java processes in the assembly line. Additionally, the Supervisor can start and stop extra process instances for each process in the assembly line to balance the workload. Finally, the supervisor also has email notification capabilities so it can send a report when a process in the assembly line fails.

The Supervisor starts and manages these processes:

Process	Description
Scheduler	The Scheduler process is a Java process responsible for orchestrating work between the rest of the other processes in the assembly line, with the exception of the Receiver and Batcher. It achieves this by monitoring the different database tables to determine if work is ready for any of the other processes and by sending notifications to them via a message bus to let them know that there is work to be done.
Receiver	The Receiver process is a Java process that monitors one or more hot directories for input files to read. It converts these input files into job objects and inserts these job objects as new records in the Jobs table.
Identifier	The Identifier process is a Java process that waits for message bus notifications from the Scheduler process that there is work to be done. The Identifier reads the Jobs table records inserted by the Receiver and breaks them into one or more transaction records that are inserted in the TRNS table.
Assembler	The Assembler process is a C process that waits for message bus notifications from the Scheduler process that there is work to be done. The Assembler reads the data from TRNS table records inserted by the Identifier and creates the NA/POL data for a document using Documaker rules. It then updates the same TRNS table records with the NA/POL data.
Distributor	The Distributor process is a C process that waits for message bus notifications from the Scheduler process that there is work to be done. The Distributor reads the NA/POL data created by the Assembler and runs Documaker rules to insert one or more recipient distribution records in the RCPS table.
Batcher	The Batcher process is a Java process that monitors the TRNS table for records that are ready for processing. It then retrieves a TRNS table record that is ready and matches the RCPS table records for it. The Batcher then uses the information in the TRNS and RCPS records to cross-reference any batching configuration options in BCHINGS table records. The Batcher then uses this information to insert new batch records in the BCHS table. The Batcher also inserts new batch-to-recipient association records in the BCHS_RCSP table.
Presenter	The Presenter process is a C process that waits for message bus notifications from the Scheduler process that there is work to be done. The Presenter reads RCPS, BCHS and BCHS_RCPS table records that where inserted by the Distributor and Batcher processes and creates prints streams. It then inserts these print streams as new records in the Pubs table.
Publisher	The Publisher process is a Java process that waits for message bus notifications from the Scheduler process that there is work to be done. The Publisher reads Pubs table records created by the Presenter process and publishes or sends their print streams to different distribution media such as SMS, email, or a printer.

Process	Description
PubNotifier	The PubNotifier process is a Java process that waits for message bus notifications from the Scheduler process that there is work to be done. The PubNotifier reads the Pubs table records created by the Presenter process and generates notifications so the recipient to know that his or her publication is available for viewing.
Archiver	The Archiver process is a Java process that waits for message bus notifications from the Scheduler process that there is work to be done. The Archiver reads the Pubs table records created by the Presenter process and archives or sends their print streams to the Universal Content Management (UCM) application.
Historian	The Historian process is a Java process responsible for moving data from the live processing tables to the history tables, purging historical, error, and log data, as well as purging BLOB/XML data from specified columns in live or historical tables. The Historian does this by using data and retention filters configured by Document Factory administrators. The Historian operates outside of the assembly line process and is instead activated as a scheduled task using the Quartz scheduling mechanism included with Document Factory.

Here is an illustration of the Supervisor and the processes it manages and monitors:



Directory Structure

Here is information about the directory and file resource structure for the Supervisor that helps explain how the Supervisor works:

Directory	Description
docfactory	This is the root directory. It houses all subdirectories and file resources needed by the Supervisor.
docfactory/bin	The bin subdirectory contains start up and other useful scripts: • bin/docfactory - the start up script for the Supervisor. • bin/patch-report - a script that can be used to obtain patch information for the Document Factory.
docfactory/lib	The lib subdirectory contains all JAR files needed by the Supervisor and all other Java processes in an assembly line. This directory is used by the Supervisor, Scheduler, Receiver, Identifier, Batcher, Archiver, Publisher, and PubNotifier Java processes.
docfactory/lib/endorsed	The endorsed subdirectory contains additional JAR files needed by the Supervisor and all other Java processes in an assembly line. However, this directory only contains JAR files that should override code that is already provided by the JRE and is needed for the assembly line processes to operate correctly. Please go to this web site for more information regarding the Java Endorsed Standards Override Mechanism: http://download.oracle.com/javase/6/docs/technotes/guides/standards/
docfactory/config	The config subdirectory contains these file resources needed to configure the Supervisor: • config/META-INF/MANIFEST.MF - A manifest file that contains the patch version information for the Supervisor. • config/context/.bindings - A JNDI file resource containing the Java Naming and Directory Interface (JNDI) names of the Java Database Connectivity (JDBC) data sources that are used by the Supervisor and all other processes in the assembly line. The .bindings file contains JNDI data sources for the configuration and assembly line schemas in the database. All processes in an the Document Factory assembly line share this resource for retrieving JNDI data sources. • config/context/log4j.xml - A Log4J configuration file that contains the loggers needed to output error or diagnostic information by the Supervisor. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j/

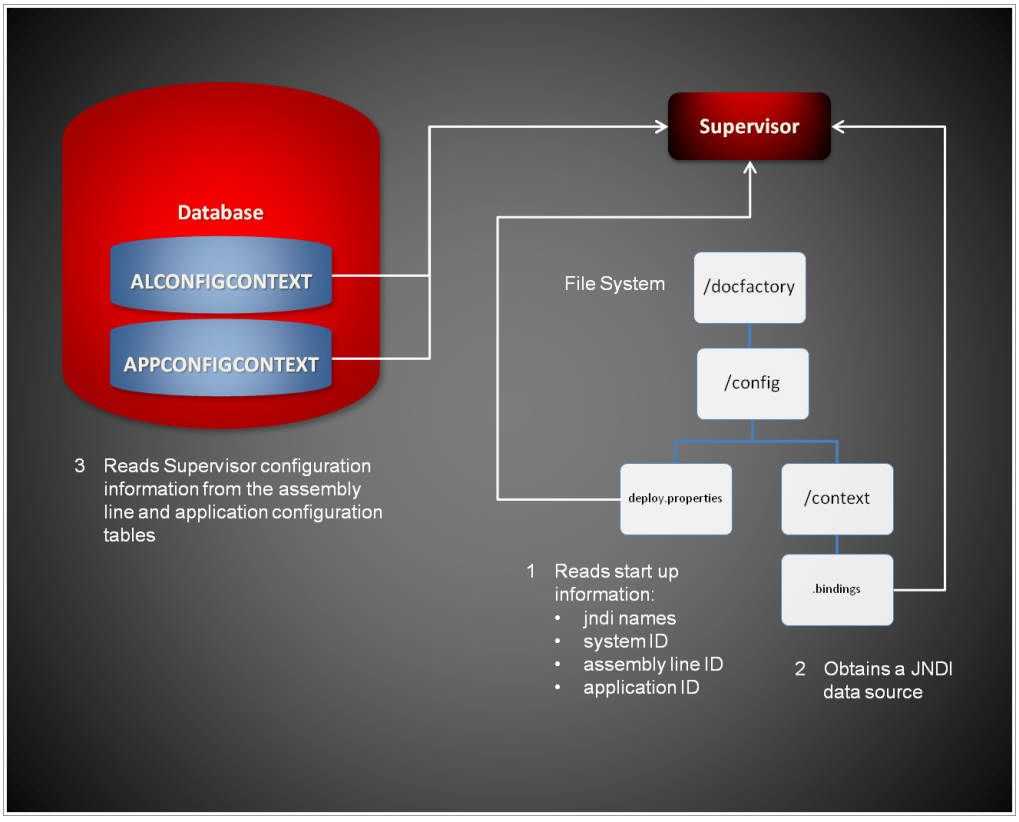
Directory	Description
docfactory/deploy	The deploy subdirectory contains a deployment JAR file for each process that is to be deployed and managed by the Supervisor. The Supervisor reads each configuration JAR file and expands it to the temp subdirectory. The expanded directory for each process is then used to read the process configuration and start it.The deploy subdirectory contains these JAR files: • scheduler.jar • receiver.jar • identifier.jar • assembler.jar • distributor.jar • batcher.jar • presenter.jar • publisher.jar • pubnotifier.jar • archiver.jar • historian.jar
docfactory/temp	The temp subdirectory contains the content of each expanded process deployment JAR file that was successfully deployed from deploy subdirectory and started by the Supervisor process.
docfactory/logs	The logs subdirectory contains the Log4J output from the Supervisor.
docfactory/errors	The errors subdirectory contains ZIP files for each process that is managed by the Supervisor and encountered a fatal error. Only the last five ZIP files are kept for each process. Files are *rolled*, meaning if five files exist for a process and the Supervisor needs to generate another one, the Supervisor overwrites the oldest one out of the five files with the new one. Each ZIP file contains diagnostic information.
docfactory/internal-db	The internal-db subdirectory is created by the Supervisor and it contains internal tables needed for process management. The Supervisor uses these internal tables to associate unique identifiers with each process and each process instance that it manages.
docfactory/global	The global subdirectory is created by the Supervisor and it contains subdirectories for each process. These subdirectories are used for process management and contain process ID files and named pipe files, which are used for inter-process communication between the Supervisor and each process it manages.
bin	The bin directory coexists at the same level as the docfactory root directory. It contains C/C++ libraries needed by the Supervisor and all other Document Factory processes in the assembly line.
bin/lib	The lib subdirectory of the bin directory that coexists at the same level as the docfactory root directory contains all Java packages that are used via JNI by the Assembler, Distributor, and Presenter C processes.
jre	The jre directory coexists at the same level as the docfactory root directory. It contains the Java Runtime Environment needed by the Supervisor and all other Document Factory processes in the assembly line.

Note See *Configuring the Supervisor* on page 80 and the configuration topics for each process for more information about the different configuration resources.

INITIALIZING THE SUPERVISOR

When the Supervisor starts, it reads its minimal startup configuration from the config/deploy.properties file. The information in the deploy.properties file tells the Supervisor which system ID, assembly line ID, and application ID values to use when retrieving its configuration information from the configuration tables.

The Supervisor then gets a JNDI data source to the configuration tables by reading the config/context/.bindings file. Next, it retrieves its configuration information from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables in the assembly line using the GROUP_NAME column value of Supervisor.



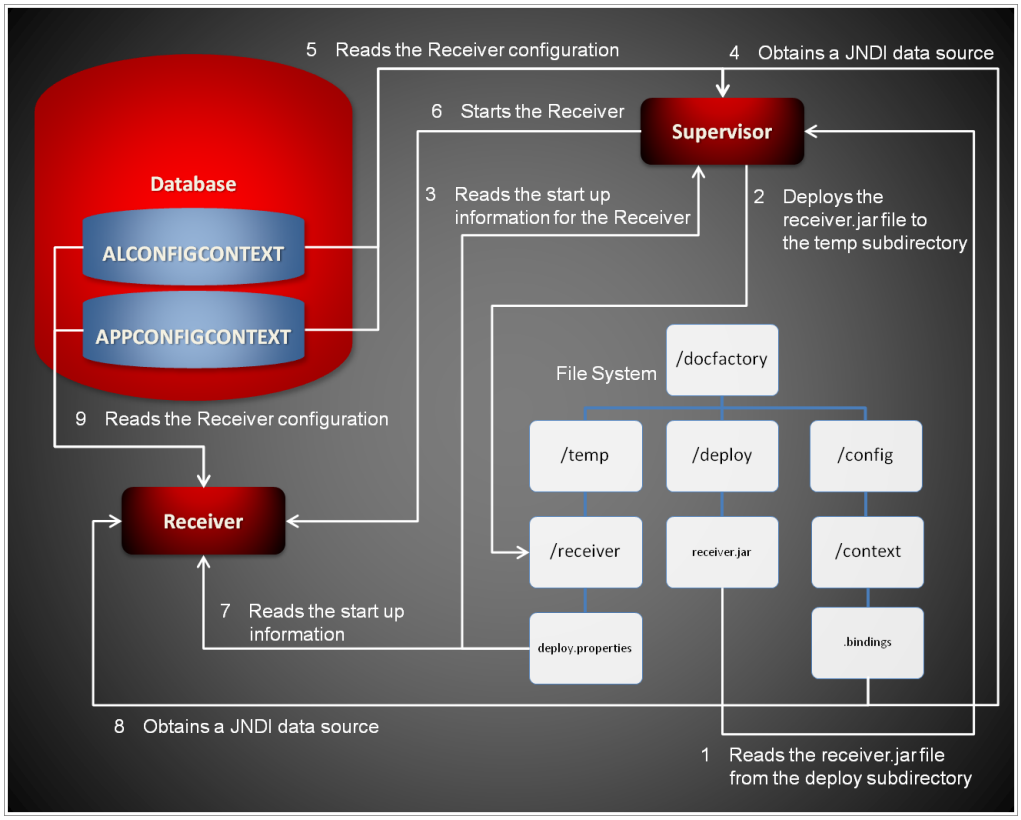
Note See *Configuring the Supervisor* on page 80 for additional information about the deploy.properties and .bindings files.

DEPLOYING PROCESSES

The Supervisor reads each process configuration JAR file from the deploy subdirectory and deploys it to a \temp subdirectory. It then reads the startup information for each process from its expanded directory in the \temp subdirectory. For example, if the Supervisor is starting the Receiver, the Supervisor deploys/expands the deploy/receiver.jar file into the temp/receiver directory.

The Supervisor then reads the minimal start up information for the Receiver from the temp/Receiver/deploy.properties file, retrieves the configuration options for the Receiver from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables using the JNDI data source specified in the config/context/.bindings file and then starts the Receiver process.

The Receiver in turn, also reads configuration options upon start up using its temp/Receiver/deploy.properties file and the config/context/.bindings file. These steps are repeated for each process that is deployed, started, and managed by the Supervisor.



Starting and Stopping a Process

To	Then
Verify a process is running	Verify there is a running process with the name docfactory_*ProcessName*, where *ProcessName* is the name of the process configuration JAR file. For example, if you want to verify the Identifier process is running, then verify there is a running process with the name docfactory_identifier.
Stop a process	Remove the process configuration JAR file from the deploy subdirectory. For example, if stopping the Identifier, remove the identifier.jar file from the deploy subdirectory.
Restart a process	Overwrite the process configuration JAR file in the deploy subdirectory with a process configuration JAR file of the same name that has a different time stamp. For example, if restarting the Identifier, replace the identifier.jar file in the deploy subdirectory with another identifier.jar file that contains a different time stamp.

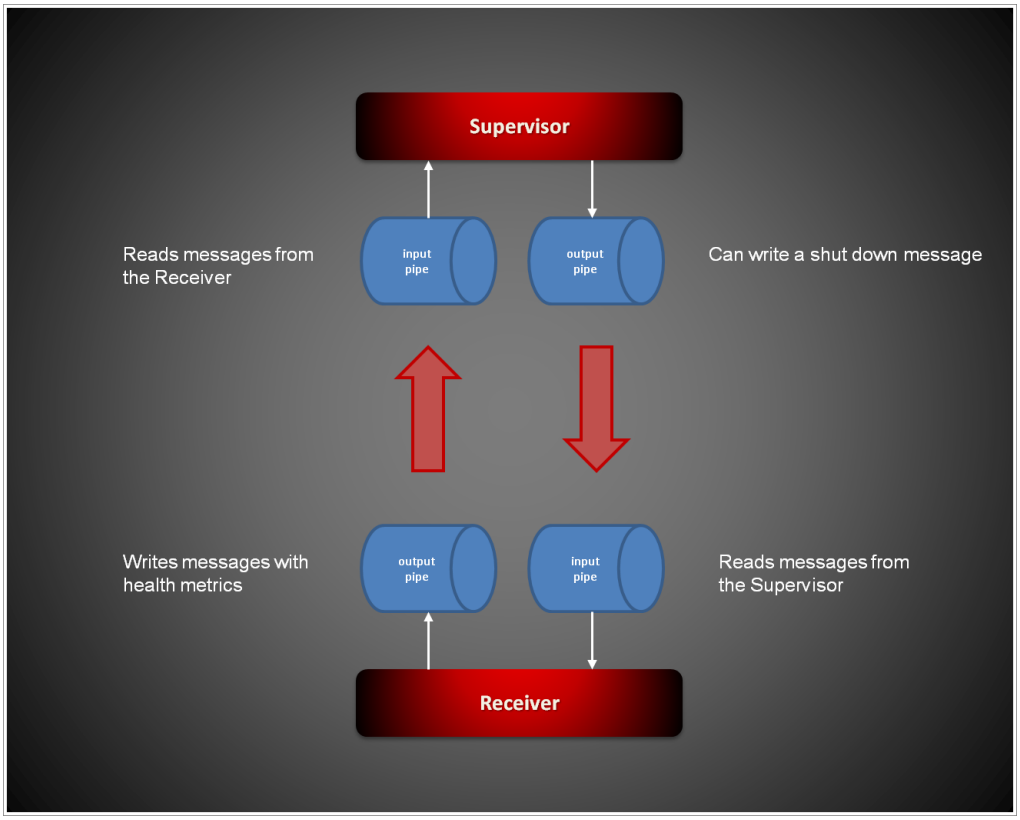
COMMUNICATING WITH PROCESSES

The Supervisor uses named pipes for inter-process communication with each process it manages. What this means is that both the Supervisor and the process being managed by the Supervisor, must create two named pipes each.

The Supervisor creates an input and an output pipe for each process instance it manages, and in turn, the process instance being managed creates an input and an output pipe as well.

The input pipe for the Supervisor becomes the output pipe for the process being managed, and the output pipe for the Supervisor becomes the input pipe for the process being managed.

Using named pipes, the process being managed can report its health metrics to the Supervisor and the Supervisor can tell a process when to shut down or terminate.



Note Java processes that run under the Supervisor use the Documaker-Process.jar package which provides all interprocess communication functionality needed to communicate with the Supervisor. The Java processes extend the oracle.documaker.process.worker.Worker class in the Documaker-Process.jar package to run under the Supervisor.

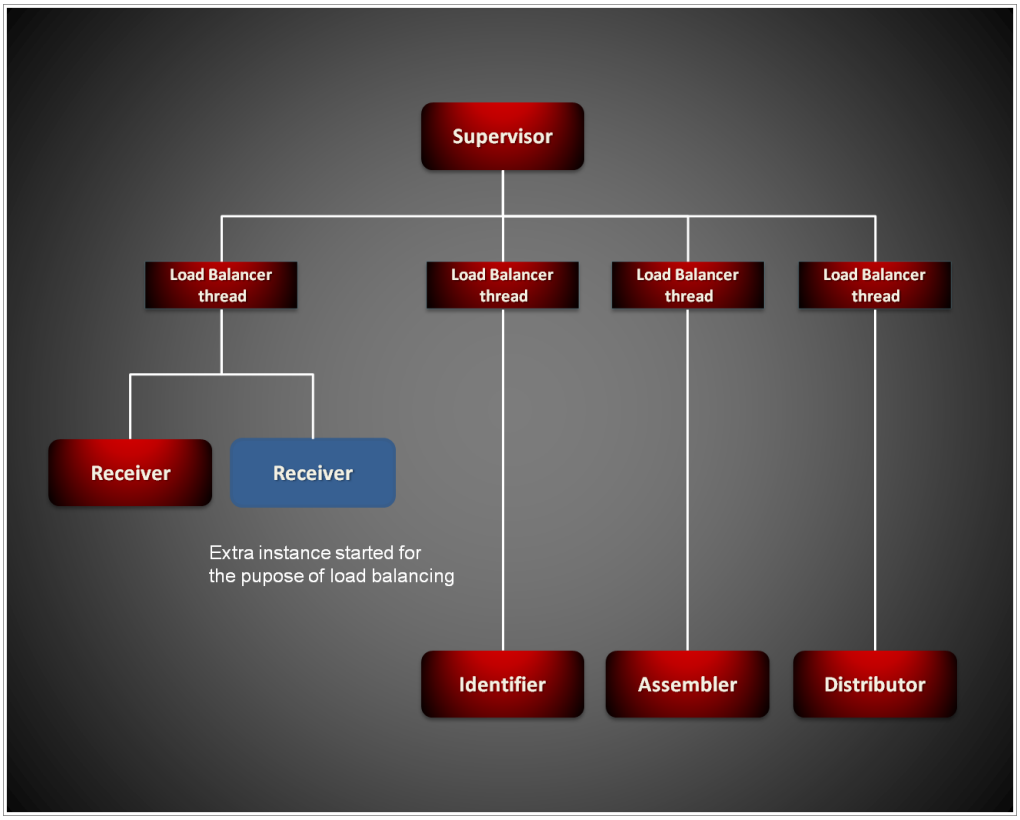
See the Documaker-Process API documentation for additional information about the Worker class.

PROVIDING LOAD BALANCING

The Supervisor creates a separate load balancing thread for each process instance that it manages. The load balancing thread gets information metrics from the inter-process communication between the Supervisor and a process instance it manages.

Using several configuration options, along with the metrics reported by each process to the Supervisor, the Supervisor can determine if all process instances for a particular process are busy and if it needs to start more to balance the workload.

The Supervisor also knows when to stop any extra process instances that are idle and which where started for the sole purpose of load balancing.



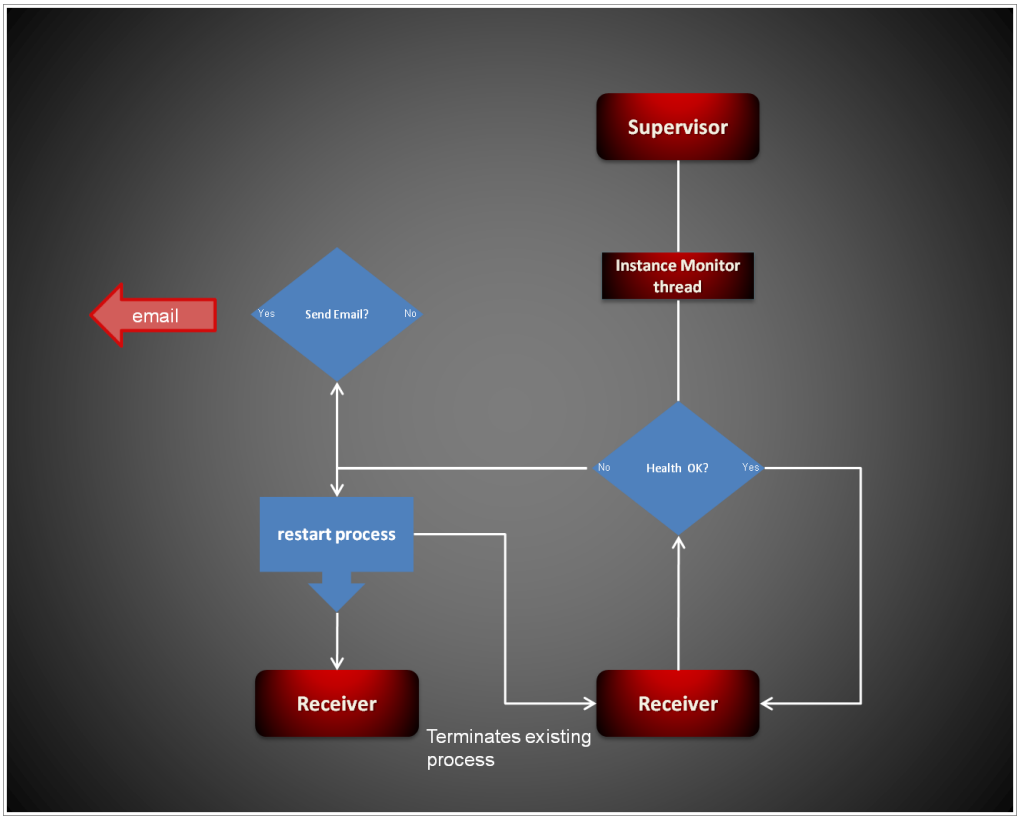
Note See the UseLoadBalancing configuration option in the Configuration topic for each process to find out more about load balancing and the configuration options that are available.

MONITORING THE PROCESSES

The Supervisor also creates a separate process instance monitor thread to monitor the health of each process. Each process the Supervisor manages reports certain health metrics to the Supervisor instance monitor thread via inter-process communication.

Based on these metrics and certain configuration options, the Supervisor instance monitor thread knows if it needs to restart a process instance. Also, the Supervisor instance monitor thread restarts a process instance if it fails to respond or report its metrics or terminates abnormally.

Additionally, the Supervisor instance monitor thread can send email notifications with diagnostic information in the event a process instance fails.



Note See the Configuration topic for each process to find out more about what options are available for health metrics and monitoring.

Starting and Stopping the Supervisor

To	Then
Verify the Supervisor is running	Verify there is a running process with the name docfactory_supervisor.
Start the Supervisor	Invoke the bin/docfactory script from a terminal or console window.
Stop the Supervisor	Press CTRL+C in the terminal or console window where the Supervisor is running to stop it. It can take a few minutes for the Supervisor to stop as it needs to send a shut down message to each process and wait for each process to terminate before it can shut down.

CONFIGURING THE SUPERVISOR

The configuration information for the Supervisor is stored in these resources:

Resource	Contains the
deploy.properties file	Minimal startup configuration information.
.bindings file	Java Naming and Directory Interface (JNDI) data sources for the Supervisor and all child processes the Supervisor manages.
log4j.xml file	Log4J diagnostic and error output captured during start up.
APPCONFIGCONTEXT table	Configuration options for the Supervisor and all child processes the Supervisor manages.

deploy.properties File

The deploy.properties file contains the minimal startup configuration options used to read the configuration for the Supervisor from the APPCONFIGCONTEXT table. This file is located in the \config subdirectory of Document Factory.

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT table for the Supervisor configuration.
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT table for the Supervisor configuration.
application.id	The value of APP_ID column in the APPCONFIGCONTEXT table for the Supervisor configuration.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT table.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=1
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

.bindings File

The .bindings file contains the Java Naming and Directory Interface (JNDI) data sources used by the Supervisor and any process the Supervisor starts. It is located in the config\context subdirectory of Document Factory.

Each JNDI data source contains these configuration options:

Option	Description
ClassName	The data source fully-qualified class name. Use the javax.sql.DataSource value.
FactoryName	The data source factory fully-qualified class name. Use the org.apache.commons.dbcp.BasicDataSourceFactory value. The BasicDataSourceFactory class supports connection pooling.
driverClassName	The Java Database Connectivity (JDBC) driver class name.
url	The JDBC URL.
maxOpenPreparedStatements	The maximum number of prepared statements to cache in the connection pool. Use the value -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	How often the idle object evictor thread should run and clean up the stale connection handles. Use the value -1 to disable the idle object evictor thread.
validationQuery	A validation query that should be run when borrowing objects from the connection pool.
username	The JDBC user name.
password	The JDBC password.
testOnBorrow	Set this option to Yes if validationQuery should be used when borrowing an object from the connection pool. The default is No.
initialSize	The initial connection pool size.
maxActive	The maximum number of active connections in the pool.
maxIdle	The maximum number of idle connections in the pool.
minIdle	The minimum number of idle connections in the pool.
maxWait	The maximum time (in milliseconds) to wait for a connection object to be retrieved from the pool before issuing an error.

Note These values are updated when each assembly line is installed.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
```

```
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type= maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
DMKRConfig/RefAddr/9/Type=maxWait
#Unix friendly Documaker Doc. Factory JNDI DataSource
DMKRFactory/ClassName=javax.sql.DataSource
DMKRFactory/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRFactory/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRFactory/RefAddr/0/Encoding=String
DMKRFactory/RefAddr/0/Type=driverClassName
DMKRFactory/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRFactory/RefAddr/1/Encoding=String
DMKRFactory/RefAddr/1/Type=url
DMKRFactory/RefAddr/10/Content=-1
DMKRFactory/RefAddr/10/Encoding=String
DMKRFactory/RefAddr/10/Type=maxOpenPreparedStatements
DMKRFactory/RefAddr/11/Content=-1
DMKRFactory/RefAddr/11/Encoding=String
DMKRFactory/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRFactory/RefAddr/12/Content=select 1 from dual
DMKRFactory/RefAddr/12/Encoding=String
DMKRFactory/RefAddr/12/Type=validationQuery
DMKRFactory/RefAddr/2/Content=dmkr_asline
DMKRFactory/RefAddr/2/Encoding=String
DMKRFactory/RefAddr/2/Type=username
DMKRFactory/RefAddr/3/Content=oracle12
DMKRFactory/RefAddr/3/Encoding=String
DMKRFactory/RefAddr/3/Type=password
DMKRFactory/RefAddr/4/Content=true
```

```
DMKRFactory/RefAddr/4/Encoding=String
DMKRFactory/RefAddr/4/Type=testOnBorrow
DMKRFactory/RefAddr/5/Content=1
DMKRFactory/RefAddr/5/Encoding=String
DMKRFactory/RefAddr/5/Type=initialSize
DMKRFactory/RefAddr/6/Content=8
DMKRFactory/RefAddr/6/Encoding=String
DMKRFactory/RefAddr/6/Type= maxActive
DMKRFactory/RefAddr/7/Content=8
DMKRFactory/RefAddr/7/Encoding=String
DMKRFactory/RefAddr/7/Type=maxIdle
DMKRFactory/RefAddr/8/Content=0
DMKRFactory/RefAddr/8/Encoding=String
DMKRFactory/RefAddr/8/Type=minIdle
DMKRFactory/RefAddr/9/Content=60000
DMKRFactory/RefAddr/9/Encoding=String
DMKRFactory/RefAddr/9/Type=maxWait
```

log4j.xml File

The log4j.xml file contains loggers that are used during the start up of the Supervisor, prior to the Supervisor loading the Log4J configuration from the APPCONFIGCONTEXT table. See the Log4J configuration options in the *APPCONFIGCONTEXT Table* on page 83 for more information.

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Supervisor*:

Option	Description
UseJMX	(Optional) This option controls if JMX is used to monitor the health metrics for the Supervisor. Enabling this option lets the Supervisor also monitor class loading, memory usage, garbage collection, and deadlocks in Java code. You can enter Yes or No. The default is No.
JMXCheckIntervalSeconds	(Optional) This option controls the time interval used to run JMX checks when the UseJMX option is enabled. The default is 60 seconds.
JMXMemoryChecks	(Optional) This option controls the total count of consecutive JMX memory checks that must be present, where the memory usage by the Supervisor exceeds the value provided for the MaxMemoryUsagePercent option for each check, at which point the Supervisor will restart. The interval for each check is controlled by the JMXCheckIntervalSeconds option. The default is -1, which disables this option.
JMXVerboseMemory	(Optional) This option controls if the Supervisor turns on verbose memory to output GC statistics when the UseJMX option is enabled. You can enter Yes or No. The default is No.
JMXVerboseClassLoader	(Optional) This option controls if the Supervisor turns on verbose class loading when the UseJMX option is enabled. You can enter Yes or No. The default is No.

Here is an example:

Option	Value
UseJMX	Yes
JMXCheckIntervalSeconds	30
JMXMemoryChecks	5
JMXVerboseMemory	Yes
JMXVerboseClassLoader	Yes

Log4J configuration options

For specific information on the Log4J configuration options, see *Defining Log4J Configuration Options* on page 277.

STARTING A PROCESS

When the Supervisor starts a process, it reads the startup configuration information for a process from the configuration jar file and from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables in the \deploy subdirectory.

Note See the following topics for more information on starting other Document Factory processes:

- *Using the Scheduler* on page 94
- *Configuring the Receiver* on page 132
- *Configuring the Identifier* on page 146
- *Configuring the Assembler* on page 157
- *Configuring the Distributor* on page 174
- *Configuring the Batcher* on page 191
- *Configuring the Presenter* on page 209
- *Configuring the Historian* on page 236
- *Configuring the Archiver* on page 229

Configuration JAR File

There is a separate configuration jar file for each process. A configuration jar file for a process contains several configuration resources.

Component	Description
deploy.properties	Contains the minimal startup configuration information for the process.
log4j.xml	Used to control the different log4j loggers to capture diagnostic output.
log4j.dtd	Used by log4j.xml file.

deploy.properties File

Contains minimal startup configuration options for a process.

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the process configuration.
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the process configuration.
application.id	The value of APP_ID column in the APPCONFIGCONTEXT table for the process configuration.
config	The configuration name for the process. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in APPCONFIGCONTEXT table for the process configuration.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

log4j.xml File

This file is used to capture Log4J diagnostic and error output during start up. See the Configuration section for each specific process in Document Factory. Log4j is a Java logging or tracing API. For more information, see this web site:

.http://logging.apache.org/log4j/

ALCONFIGCONTEXT Table

This table contains any information that may be shared across multiple processes. See the configuration information for each specific process in Document Factory.

APPCONFIGCONTEXT Table

The options and values are read from this table when the Group_Name value is the config value specified in the deploy.properties file.

For example, if the config value in the deploy.properties file is *Assembler*, the system uses the values in APPCONFIGCONTEXT table where the Group_Name is *Assembler* when it starts the Assembler process.

Option	Description
StartCommand	The process name to start. In the case of a Java process the name should be Java or the full path and name to the Java executable. In the case of a C/C++ process it should be the full path and executable name.
StartArguments	These are the arguments the process expects. In the case of a Java process, this should be the arguments the JavaClass expects. In the case of a C/C++ application these should be the arguments the StartCommand executable expects.

env.mode.*	The environment variables the process expects to run. The Supervisor creates an environment variable for each env.mode.xxx configuration option it encounters. The naming convention is shown here:
	env.*mode.name*
	Where *mode* can be either zero (0), meaning prepend, one (1), meaning append, or two (2), meaning overwrite, and *name* is the name of the environment variable.
	When the mode is not defined, the default is two (2). Here are some examples:
	`env.0.PATH`
	`env.ORACLE_HOME`
	The second example uses the default overwrite mode.
	Note: Use only for Java processes that extend the Worker class specified WorkerClass configuration option.

JavaClass	The Java class used to start the worker class specified in WorkerClass configuration option. Use the oracle.documaker.process.ProcessShell value. ProcessShell class is a process shell in Documaker-Process.jar package that provides all functionality needed to communicate with the Supervisor process and to start and manage the worker class specified in WorkerClass configuration option. Do not use this option if you are not using the WorkerClass option. Use only for Java processes that extend the Worker class specified in the WorkerClass configuration option.
JVMOptions *	Any JVM options the Supervisor process uses to start JavaClass. There is no default.
WorkerClass *	The class that extends the oracle.documaker.process.worker.Worker class in Documaker-Process.jar package and is started by the class specified in JavaClass configuration option.
WorkerThreads *	How many threads of WorkerClass should be created by JavaClass. You can use the value 1. The default is one (1).
WorkerIntervalMillis *	How often each WorkerClass thread should perform its work. The default is 5000 milliseconds.
WorkerStartDelayMillis *	How long each WorkerClass thread should wait after startup and before performing any work. The default is 10000 milliseconds.
ShutdownHookClass	The class that extends the oracle.documaker.process.shutdown.ShutdownHook class in Documaker-Process.jar package.

* = Used only when the JavaClass option is defined with a value of oracle.documaker.process.ProcessShell.
+ = The name of the configuration jar file or the value of config property in deploy.properties file.

Option	Description
HouseKeeperClass *	Each process that runs under Supervisor can perform any necessary cleanup via this class. This class extends the oracle.documaker.process.housekeeping.HouseKeeper class in Documaker-Process.jar package.
HouseKeeperIntervalMillis *	How often the HouseKeeperClass thread should perform its work. The default is 15000 milliseconds.
HouseKeeperStartDelayMillis *	How long the HouseKeeperClass thread should wait after startup and before performing any work. The default is 30000 milliseconds.
IPCIntervalMillis *	How often the inter-process communication (IPC) thread should perform its work. This option is used by JavaClass to report back to the Supervisor process. The default is 1000 milliseconds. There is no reason to change this setting, unless you want to reduce the amount of communication between the Supervisor and process.
IPCStartDelayMillis *	How long the inter-process communication (IPC) thread should wait after startup and before performing any work. This option is used by JavaClass to report back to the Supervisor process. The default is 10000 milliseconds.
Log4jIntervalMillis *	How often the Log4J resource monitor thread should perform its work. This option is used to monitor log4j.xml file deployed under temp\configName+ working directory and reload it when a change is detected. The default is 1000 milliseconds.
Log4jStartDelayMillis *	How long the Log4J resource monitor thread should wait after startup and before performing any work. This option is used to monitor log4j.xml file deployed under temp\configname + working directory and reload it when a change is detected. The default is 10000 milliseconds.
StartDirectory	This should be the start directory for a Java or C/C++ process. Leave this value blank if you wish to deploy a configuration to the temp directory and have it create a current directory for the new deployment.
Instances	(Optional) The number of instances the Supervisor should start for a process configuration. The default is two (2).

* = Used only when the JavaClass option is defined with a value of oracle.documaker.process.ProcessShell.
+ = The name of the configuration jar file or the value of config property in deploy.properties file.

Option	Description
UseLoadBalancing	(Optional) This option controls wether the Supervisor checks the idle time of a process's instances that are running and starts additional ones when all of them are busy. Instances are considered busy when their idle time is less than the value provided in the MinIdleTimeSeconds option. The Supervisor uses the value provided in the IdleTimeChecks option to determine the number of idle time checks to run before it starts additional instances. When additional instances are started for load balancing purposes, they are shut down by the Supervisor if their idle time exceeds the value in the MaxIdleTimeSeconds option. The maximum number of instances running is the value for the MaxInstances option (including the instances configured in the Instances option). The Supervisor checks the idle time of the current instances at the interval specified in the IdleTimeCheckIntervalSeconds and if all are busy, it starts an additional number of instances equal to the value provided in the IncrementCount option. Please note that the Supervisor does not start checking the busy time of the current instances until the time provided in the IdleTimeCheckDelaySeconds option elapses. Make sure the value for the delay is ample enough to provide for all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. You can enter Yes or No. The default is Yes.
MaxInstances	(Optional) This option controls the maximum number of instances that can run when the UseLoadBalancing option is enabled. The default is the number of processors times two.
IncrementCount	(Optional) This option controls how many additional instances are started during the current check when all instances running are busy and the UseLoadBalancing option is enabled. The default is two (2).
IdleTimeCheckIntervalSeconds	(Optional) This option controls how often the Supervisor checks the idle time of the instances that are running to determine if they are busy so it can start additional ones when the UseLoadBalancing option is enabled. The default is 10 seconds.
IdleTimeCheckDelaySeconds	(Optional) This option controls the initial delay before the first idle time check is performed by the Supervisor when the UseLoadBalancing option is enabled. This time should be ample enough to allow all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. The default is 120 seconds.
IdleTimeChecks	(Optional) This option defines the number of consecutive idle time checks that must fail, meaning all instances were busy during each check, before more instances are started when the UseLoadBalancing option is enabled. Each check takes place at the IdleTimeCheckIntervalSeconds interval. The default is 12.
MinIdleTimeSeconds	(Optional) This option controls the minimum idle time for each instance. The idle time represents how long it has been since an instance processed the last request. If the Supervisor detects an instance has an idle time less than the value provided for this option, it considers it busy for the purpose of load balancing. The default is 5 seconds.

* = Used only when the JavaClass option is defined with a value of oracle.documaker.process.ProcessShell.
+ = The name of the configuration jar file or the value of config property in deploy.properties file.

Option	Description
MaxIdleTimeSeconds	(Optional) This option controls the maximum idle time for an additional instance. The idle time represents how long it has been since an instance performed processing. If the Supervisor detects an instance, which was started for the purpose of load balancing, has reached an idle time greater than the value provided for this option, it sends the instance a shutdown request. The default is 120 seconds.
MaxTransactions	(Optional) This option controls the maximum number of transactions an instance can process before it is restarted by the Supervisor. The default is -1, which disables this option.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before it is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
OrderedRestartIntervalSeconds	(Optional) This option controls the interval used for restarting each process instance in a sequential/ordered manner when the MaxTransactions or MaxUpTime options are used. The Supervisor restarts one instance at a time and waits for an amount of time equal to the value specified for this option before it restarts the next one and so on until it has restarted all of them. If you set this option to less than 60 seconds, you can negatively affect performance. The default is 60 seconds.
WatchList	A comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted.
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is 5.
MaxMemoryUsagePercent	(Optional) This option controls the maximum percentage of the total JVM memory that can be used by an instance before the Supervisor will restart it. Note that the total memory used in this calculation does not include any memory used by native code. This option is used with the MemoryChecks option. The default is 95.
MemoryChecks	(Optional) This option controls the total count of consecutive memory checks that must be present, where the memory usage by an instance exceeds the value provided for the MaxMemoryUsagePercent option for each check, at which point the Supervisor will restart it. The interval for each memory check is controlled by the CheckIntervalSeconds option. The default is -1, which disables this option.
CheckIntervalSeconds	(Optional) This option controls the time interval used by the Supervisor to check the health of each instance. The default is 1 second.

* = Used only when the JavaClass option is defined with a value of oracle.documaker.process.ProcessShell.
+ = The name of the configuration jar file or the value of config property in deploy.properties file.

Option	Description
UseJMX	(Optional) This option controls if JMX is used to monitor additional health metrics for each Java process instance. Enabling this option lets the Supervisor also monitor class loading, memory usage, garbage collection, and deadlocks in Java code for each instance. Please note that enabling this option requires an additional and separate TCP/IP port for each instance so that it can be started with a JMX agent. You can enter Yes or No. The default is No. Only use this option for debugging or testing purposes. Do not use this option in production mode because it causes extra overhead and it requires additional ports be used. Only use this option with a Java process.
JMXPort	(Optional) This option controls the starting JMX port to use when starting each Java instance with a JMX agent if the UseJMX option is enabled. Please note that the starting port value should consider that each additional instance that is started will try to use a continuous/incremental port number. The default starting port value is 49163.
JMXCheckIntervalSeconds	(Optional) This option controls the time interval used to run JMX checks for each Java instance when the UseJMX option is enabled. The default is 60 seconds.
JMXMemoryChecks	(Optional) This option controls the total count of consecutive JMX memory checks that must be present, where the memory usage by a Java instance exceeds the value provided for the MaxMemoryUsagePercent option for each check, at which point the Supervisor will restart it. The interval for each check is controlled by the JMXCheckIntervalSeconds option. The default is -1, which disables this option.
JMXVerboseMemory	(Optional) This option controls if the Supervisor turns on verbose memory to output GC statistics for each Java instance when the UseJMX option is enabled. You can enter Yes or No. The default is No.
JMXVerboseClassLoader	(Optional) This option controls if the Supervisor turns on verbose class loading for each Java instance when the UseJMX option is enabled. You can enter Yes or No. The default is No.

***** = Used only when the JavaClass option is defined with a value of oracle.documaker.process.ProcessShell.
+ = The name of the configuration jar file or the value of config property in deploy.properties file.

Here is an example for the Receiver Java process:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/jre/bin/docfactory_receiver
env.0.PATH	/oracle_home/*InstallationLocation*/jre/bin, /oracle_home/*InstallationLocation*/jre/bin/client
JavaClass	oracle.documaker.process.ProcessShell
JVMOptions	-Xmx128m -Duser.name=oracle
WorkerClass	oracle.documaker.receiver.Receiver
WorkerThreads	2
WorkerIntervalMillis	1000

InstallationLocation = The installation location where you installed Document Factory.

Option	Value
WorkerStartDelayMillis	5000
ShutdownHookClass	oracle.documaker.receiver.shutdown.ReceiverShutdownHook
IPCIntervalMillis	1000
IPCStartDelayMillis	10000
Log4jIntervalMillis	1000
Log4jStartDelayMillis	15000
Instances	1
UseLoadBalancing	Yes
MaxInstances	8
IncrementCount	1
IdleTimeCheckIntervalSeconds	15
IdleTimeCheckDelaySeconds	240
IdleTimeChecks	5
MinIdleTimeSeconds	5
MaxIdleTimeSeconds	120
MaxTransactions	-1
MaxReportIntervalSeconds	180
MaxUpTimeSeconds	-1
MaxMemoryUsagePercent	95
MemoryChecks	5
CheckIntervalSeconds	1
UseJMX	No
JMXPort	49192
JMXCheckIntervalSeconds	30
JMXMemoryChecks	5
JMXVerboseMemory	No
JMXVerboseClassLoader	No
WaitForShutdownSeconds	60

InstallationLocation = The installation location where you installed Document Factory.

Option	Value
OrderedRestartIntervalSeconds	60
WatchList	/oracle_home/*InstallationLocation*/docfactory/config/context/.bindings,/oracle_home/*InstallationLocation*/docfactory/config/deploy.properties
MaxRestarts	5

InstallationLocation = The installation location where you installed Document Factory.

Here is an example for the Assembler C process:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/bin/docfactory_assembler
StartArguments	/ini=fsiuser_1.ini /debug=0 /phase=1
env.0.PATH	/oracle_home/*InstallationLocation*/oracle_instantclient_11_2,,/oracle_home/*InstallationLocation*/jre/bin,/oracle_home/*InstallationLocation*/jre/bin/client,/oracle_home/*InstallationLocation*/bin
env.ORACLE_HOME	/oracle_home/*InstallationLocation*/bin
env.NLS_LANG	AMERICAN_AMERICA.AL32UTF8
env.TNS_ADMIN	/oracle_home/*InstallationLocation*/oracle_instantclient_11_2/NETWORK/ADMIN
env.JVM_OPTIONS	-Xmx256m,-Duser.name=oracle,-Dlog4j.configuration=/oracle_home/*InstallationLocation*/docfactory/temp/assembler/log4j.xml,-Dlog4j.file=/oracle_home/*InstallationLocation*/docfactory/temp/assembler/logs/log4j.log,-Djndi.context=/oracle_home/*InstallationLocation*/docfactory/config/context,-Dfactory.jndi.name=DMKRFactory,-Dconfig.jndi.name=DMKRConfig,-Dschema=DMKR_ASLINE
StartDirectory	/oracle_home/*InstallationLocation*/dmres/correspondence
Instances	2
UseLoadBalancing	No
MaxInstances	8
IncrementCount	1
IdleTimeCheckIntervalSeconds	15
IdleTimeCheckDelaySeconds	240
IdleTimeChecks	5
MinIdleTimeSeconds	5
MaxIdleTimeSeconds	120
MaxTransactions	-1

InstallationLocation = The installation location where you installed Document Factory.

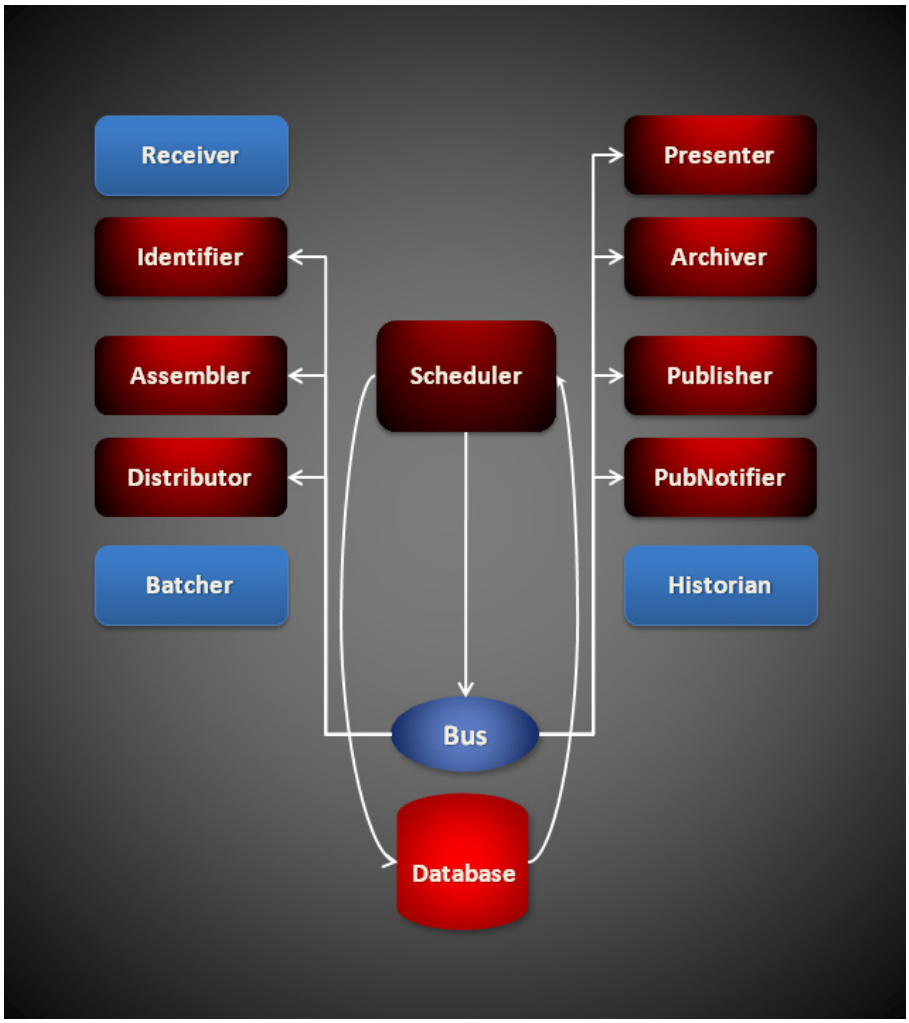
Option	Value
MaxReportIntervalSeconds	180
MaxUpTimeSeconds	-1
WaitForShutdownSeconds	60
OrderedRestartIntervalSeconds	60
WatchList	/oracle_home/*InstallationLocation*/dmres/correspondence/fsiuser_1.ini,/ oracle_home/*InstallationLocation*/dmres/correspondence/fsisys.ini
MaxRestarts	5

InstallationLocation = The installation location where you installed Document Factory.

USING THE SCHEDULER

The Scheduler is a Java process that orchestrates the work between all other processes in the Document Factory assembly line, with the exception of the Receiver and Batcher. It achieves this by polling the different tables in the assembly line for status codes that indicate a transaction is ready for the next process in the assembly line.

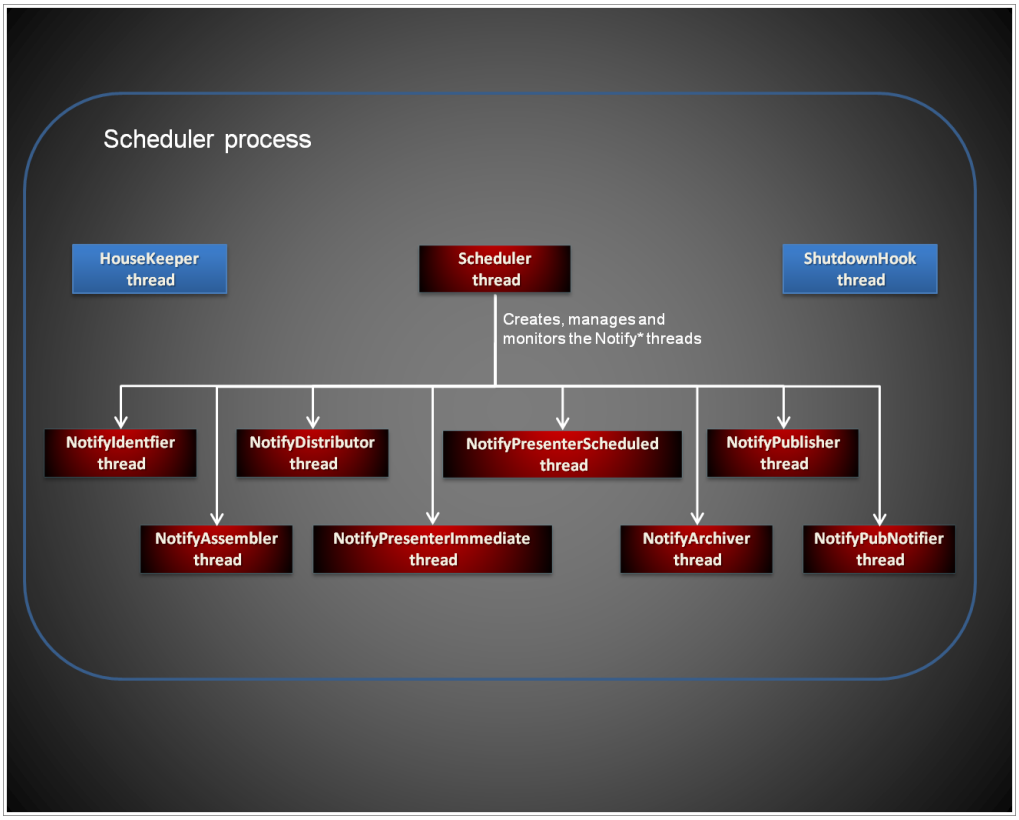
The Scheduler then sends message notifications through a message bus to inform a process there are transactions ready for it. The Scheduler is multi-threaded, meaning it uses a separate thread to orchestrate the work for each process in the assembly line.



Here is a list of the threads the Scheduler uses:

THE SCHEDULER THREAD

The Scheduler thread is the main thread of the Scheduler process and it is the thread that starts and manages the Notify* threads. It can detect when any Notify* thread is not running and restart it. It is also responsible for detecting shut down messages from the Supervisor and shutting down all the Notify* threads prior to terminating.

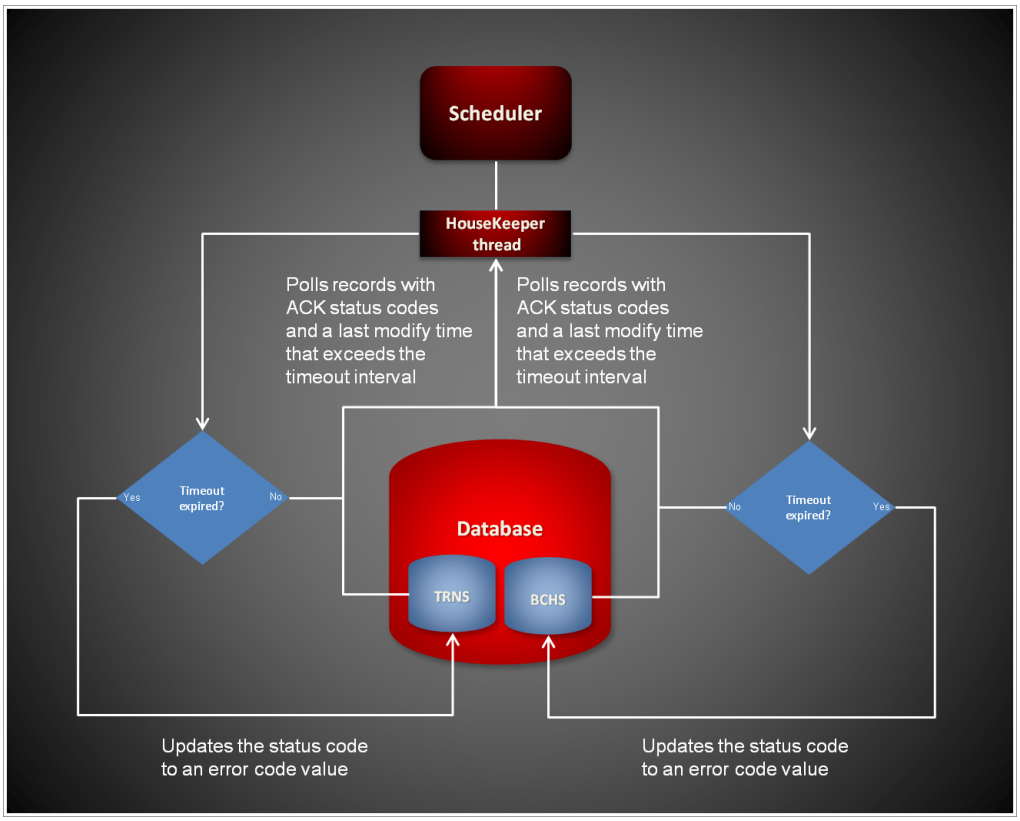


THE HOUSEKEEPER THREAD

This thread is responsible for detecting transactions received by other processes an flagging them with an *Error* code status if the process that received them has not updated their status code from ACK status within a certain period of time after receiving them.

It does this by monitoring the different ACK status codes in the TRNS and BCHS tables and setting them to the *Error* code status if the TransactionTimeoutMillis option value for this thread has expired. The TransactionTimeoutMillis value is compared to the values for the Trns.MODIFYTIME or Bchs.BCHMODIFYTIME columns to determine if it has expired.

For example, if the HouseKeeper thread finds a TRNS record with a status code of Identifier-ACK (131) and the value for the TransactionTimeoutMillis option is set to 360000 milliseconds, but the MODIFYTIME column value for the TRNS record indicates the last time the record was modified with the ACK status has exceeded the timeout value, then the HouseKeeper thread will set the status code to Identifier-Error (141).



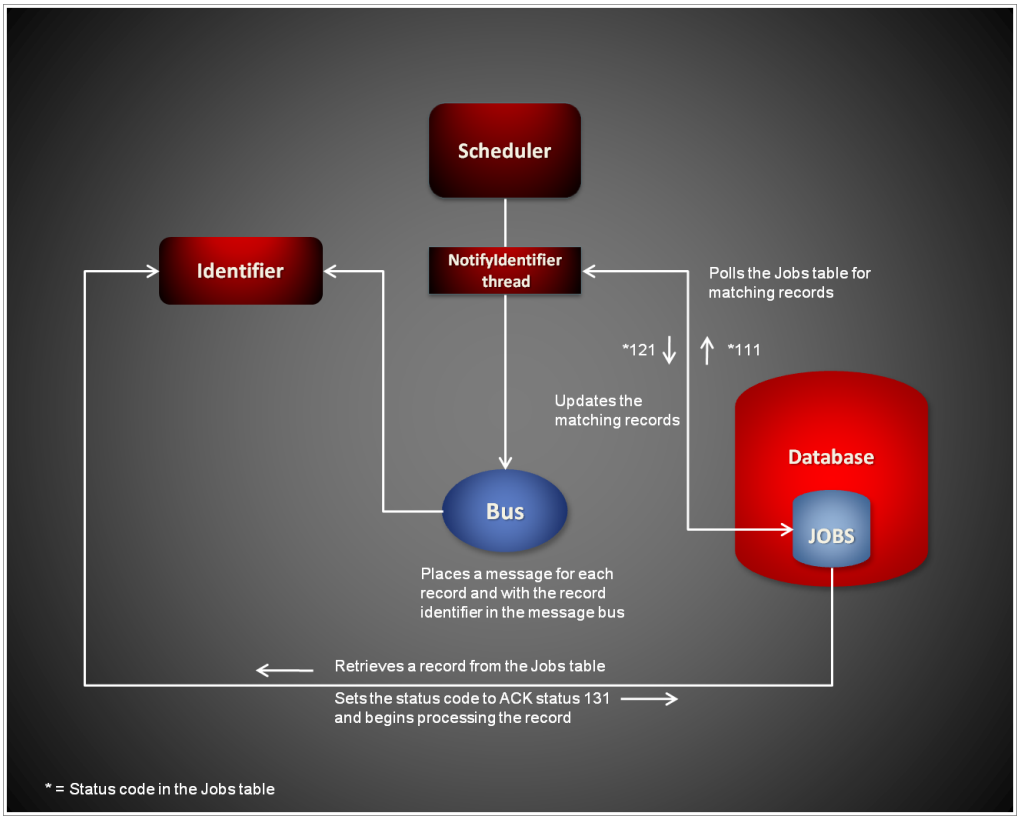
THE SHUTDOWNHOOK THREAD

This thread is invoked during normal shutdown to perform internal process clean up.

THE NOTIFYIDENTIFIER THREAD

This thread monitors the Jobs table for records with a status code value of Identifier-Ready (111). It then changes the status code value for each record that is ready to Identifier-InProgress (121) and places a notification message in the message bus for the Identifier process to indicate there is a Jobs table record ready for processing.

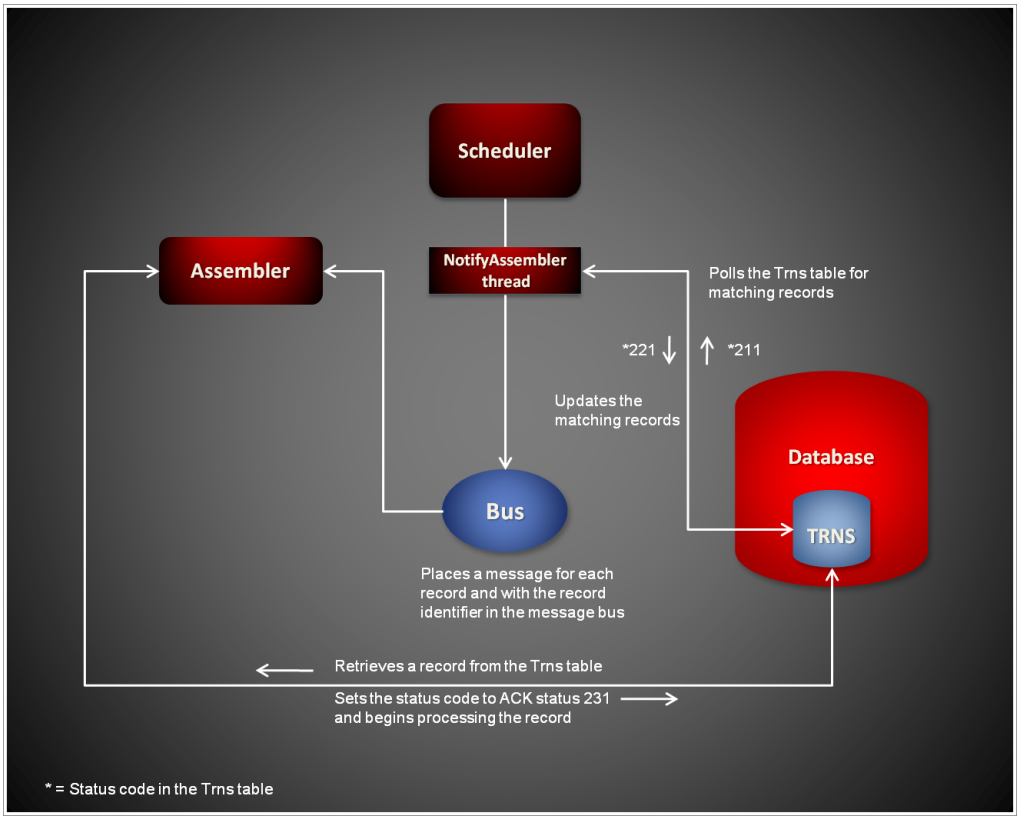
The notification message contains the record identifier value for the record that is ready. If there is an error during processing, the NotifyIdentifier thread changes the status code value to Identifier-Error (141).



THE NOTIFYASSEMBLER THREAD

This thread monitors the TRNS table for records with a status code value of Assembler-Ready (211). It then changes the status code value for each record that is ready to Assembler-InProgress (221) and places a notification message in the message bus for the Assembler process to indicate there is a TRNS table record ready for processing.

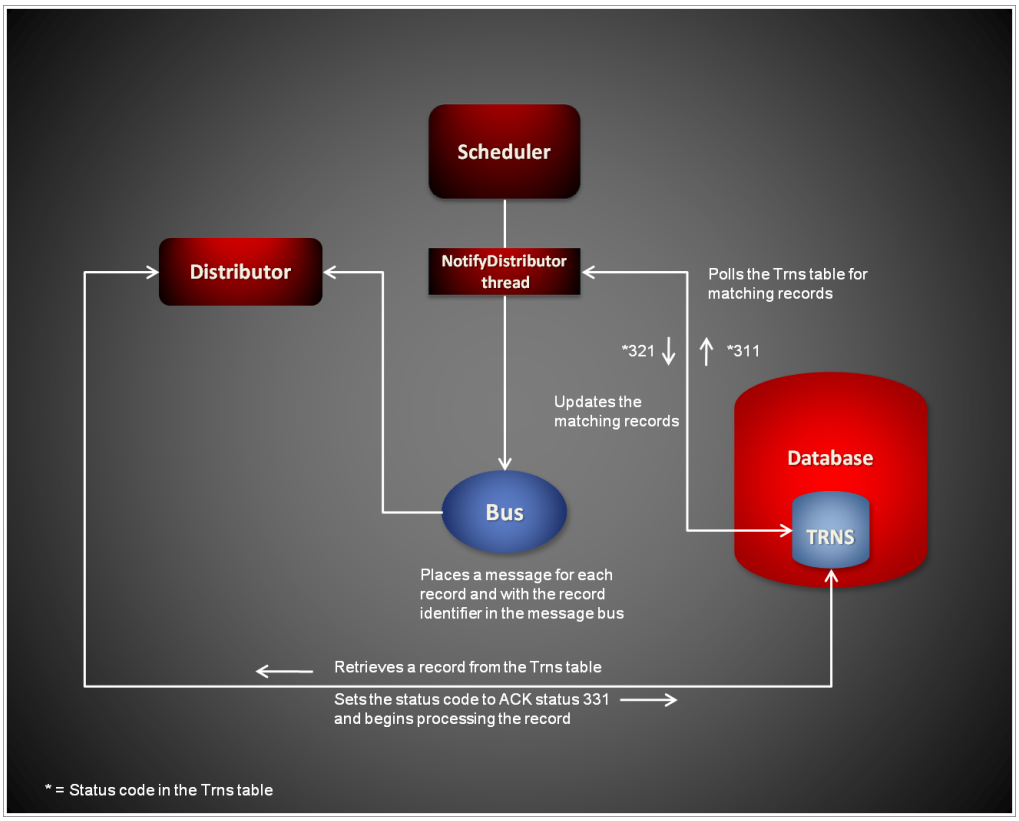
The notification message contains the record identifier value for the record that is ready. If there is an error during processing, the NotifyAssembler thread changes the status code value to Assembler-Error (241).



THE NOTIFYDISTRIBUTOR THREAD

This thread monitors the TRNS table for records with a status code value of Distributor-Ready (311). It then changes the status code value for each record that is ready to Distributor-InProgress (321) and places a notification message in the message bus for the Distributor process to indicate there is a TRNS table record ready for processing.

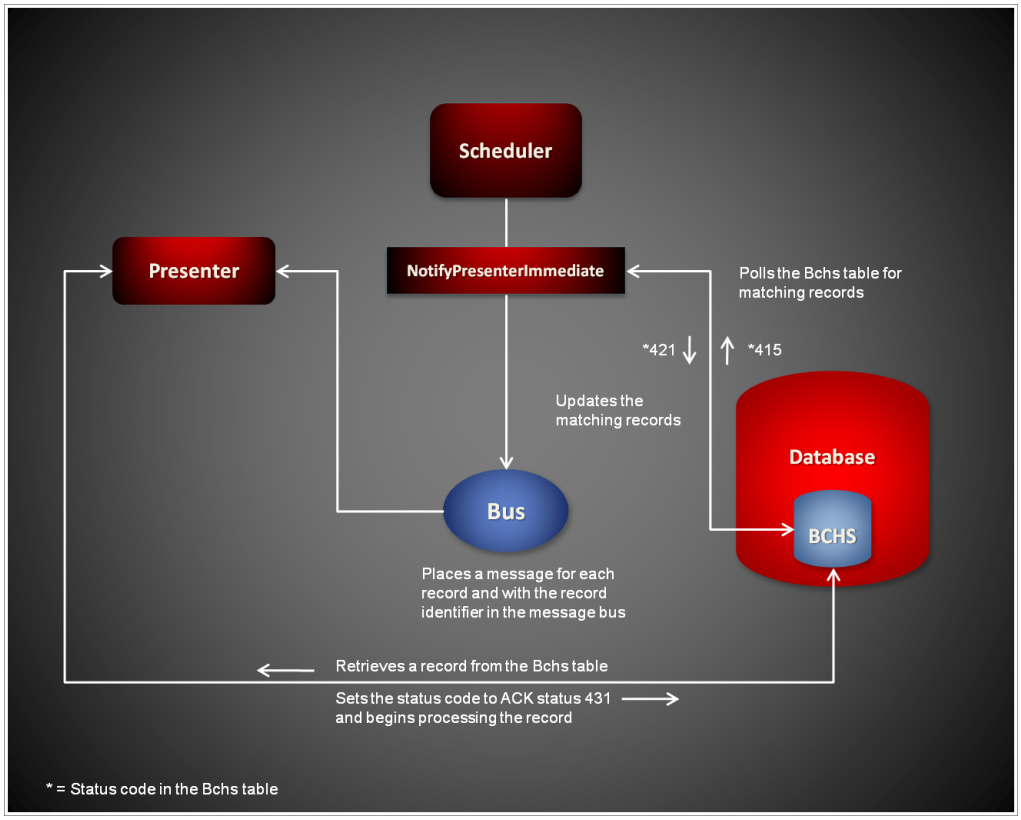
The notification message contains the record identifier value for the record that is ready. If there is an error during processing, the NotifyDistributor thread changes the status code value to Distributor-Error (341).



THE NOTIFYPRESENTERIMMEDIATE THREAD

This thread monitors the Bchs table for records with a status code value of Batcher-InProgress (415). It then changes the status code value for each record that is ready to Presenter-InProgress (421) and places a notification message in the message bus for the Presenter process to indicate there is a Bchs table record ready for processing.

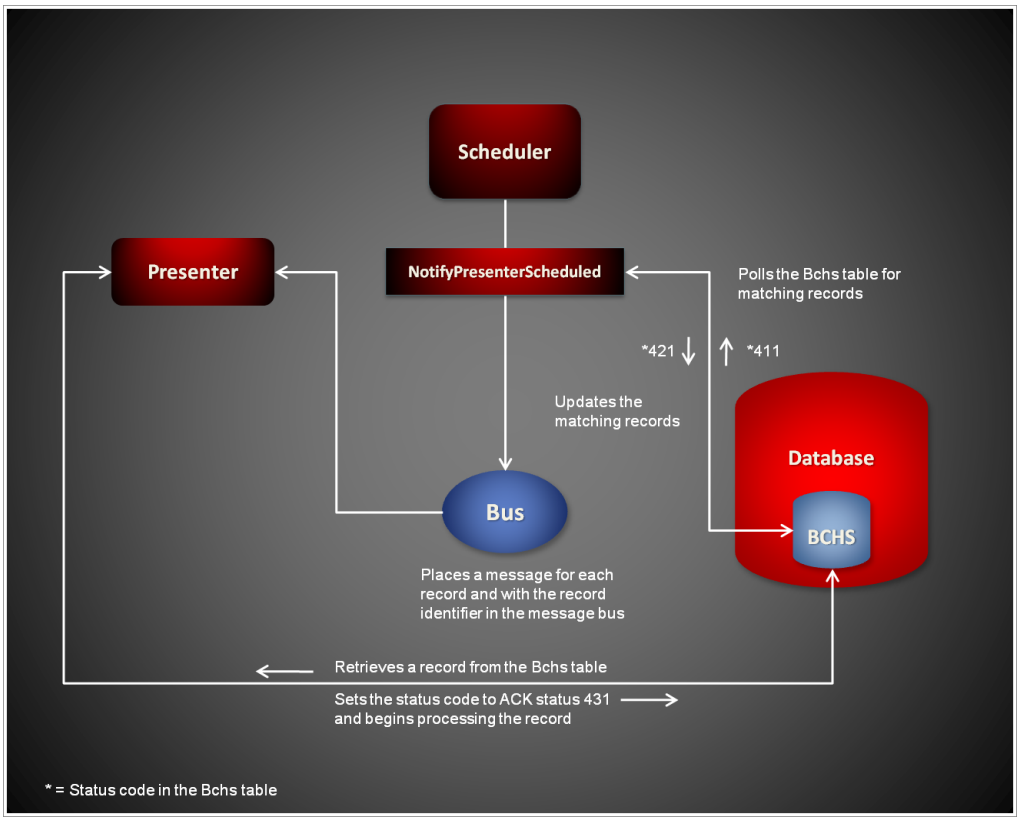
The notification message contains the record identifier value for the record that is ready. If there is an error during processing, the NotifyPresenterImmediate thread changes the status code value to Presenter-Error (441).



THE NOTIFYPRESENTERSCHEDULED THREAD

This thread monitors the Bchs table for records with a status code value of Presenter-Ready (411) and a non-null value for the BCHSTARTINGTIME column that is less than the current time. It then changes the status code value for each record that is ready to Presenter-InProgress (421) and places a notification message in the message bus for the Presenter process to indicate there is a Bchs table record ready for processing.

The notification message contains the record identifier value for the record that is ready. If there is an error during processing, the NotifyPresenterScheduled thread changes the status code value to Presenter-Error (441).

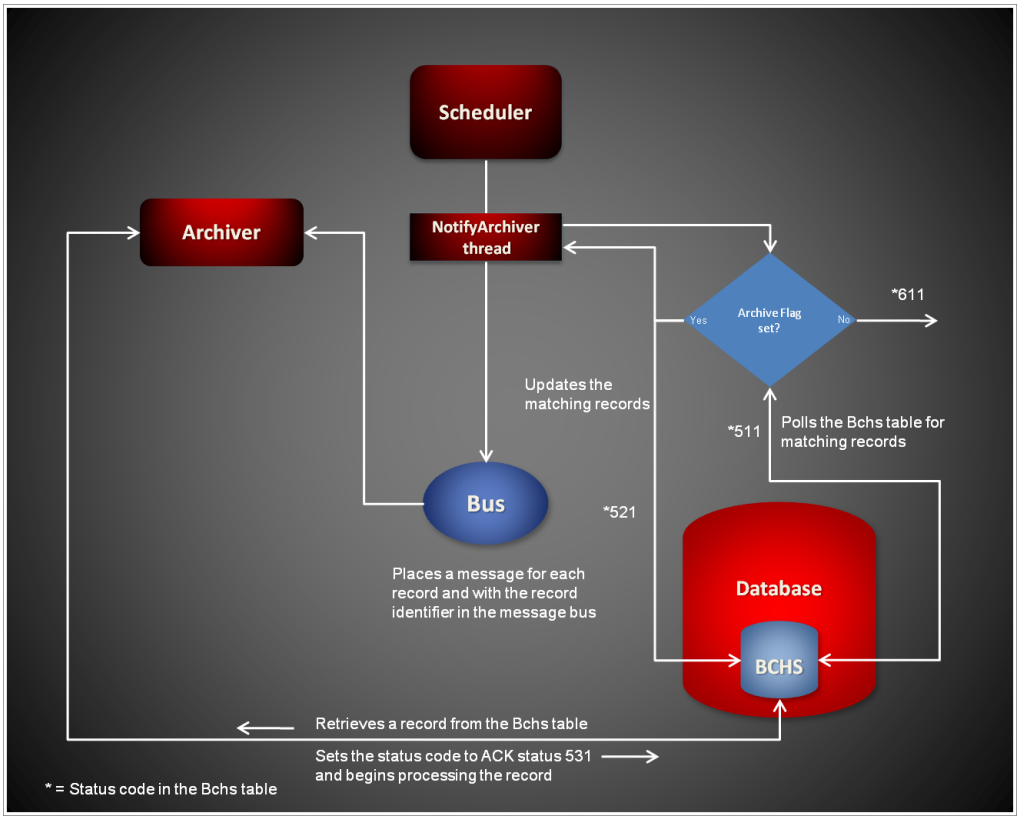


THE NOTIFYARCHIVER THREAD

This thread monitors the Bchs table for records with a status code value of Archiver-Ready (511) and with a BCHARCHIVE column value of one (1). It then changes the status code value for each record that is ready to Archiver-InProgress (521) and places a notification message in the message bus for the Archiver process to indicate there is a Bchs table record ready for processing.

The notification message contains the record identifier value for the record that is ready. If a Bchs record contains a status code value of Archiver-Ready (511) but the BCHARCHIVE column value is zero (0), the NotifyArchiver thread changes the status code to Publisher-Ready (611).

If there is an error during processing, the NotifyArchiver thread changes the status code value to Archiver-Error (541).

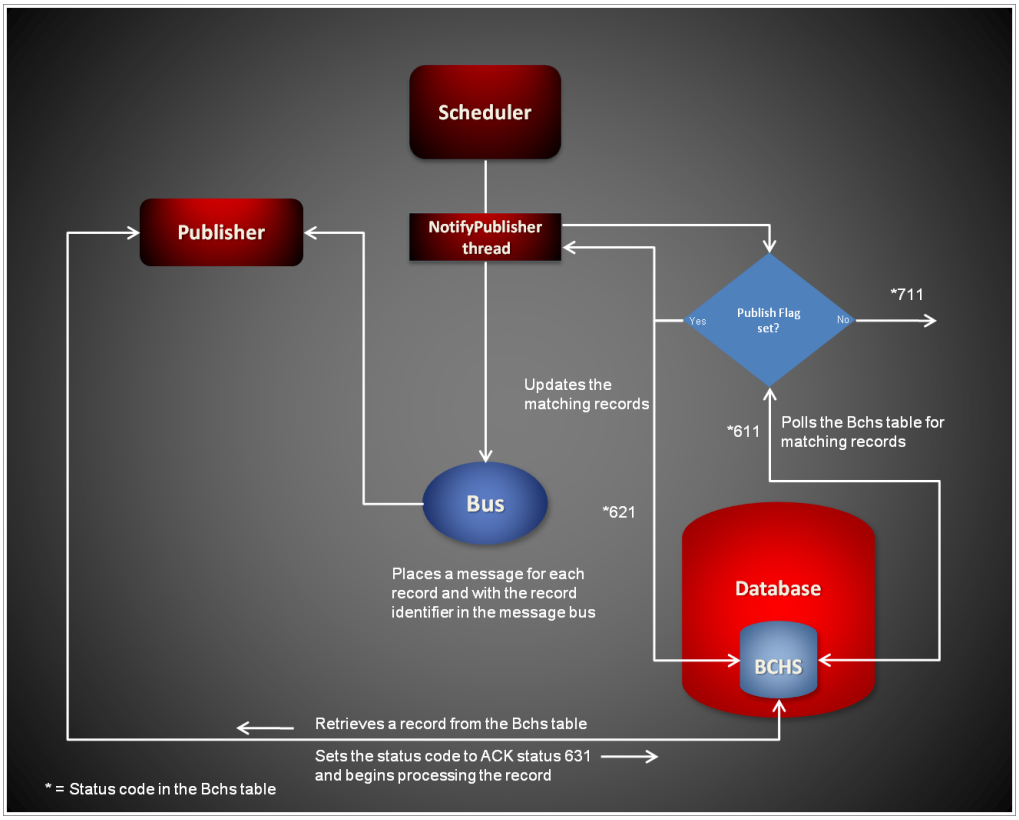


THE NOTIFYPUBLISHER THREAD

This thread monitors the Bchs table for records with a status code value of Publisher-Ready (611) and with a BCHPUBLISH column value of one (1). It then changes the status code value for each record that is ready to Publisher-InProgress (621) and places a notification message in the message bus for the Publisher process to indicate there is a Bchs table record ready for processing.

The notification message contains the record identifier value for the record that is ready. If a Bchs record contains a status code value of Publisher-Ready (611) but the BCHPUBLISH column value is zero (0), the NotifyPublisher thread changes the status code to PubNotifier-Ready (711).

If there is an error during processing, the NotifyPublisher thread changes the status code value to Publisher-Error (641).

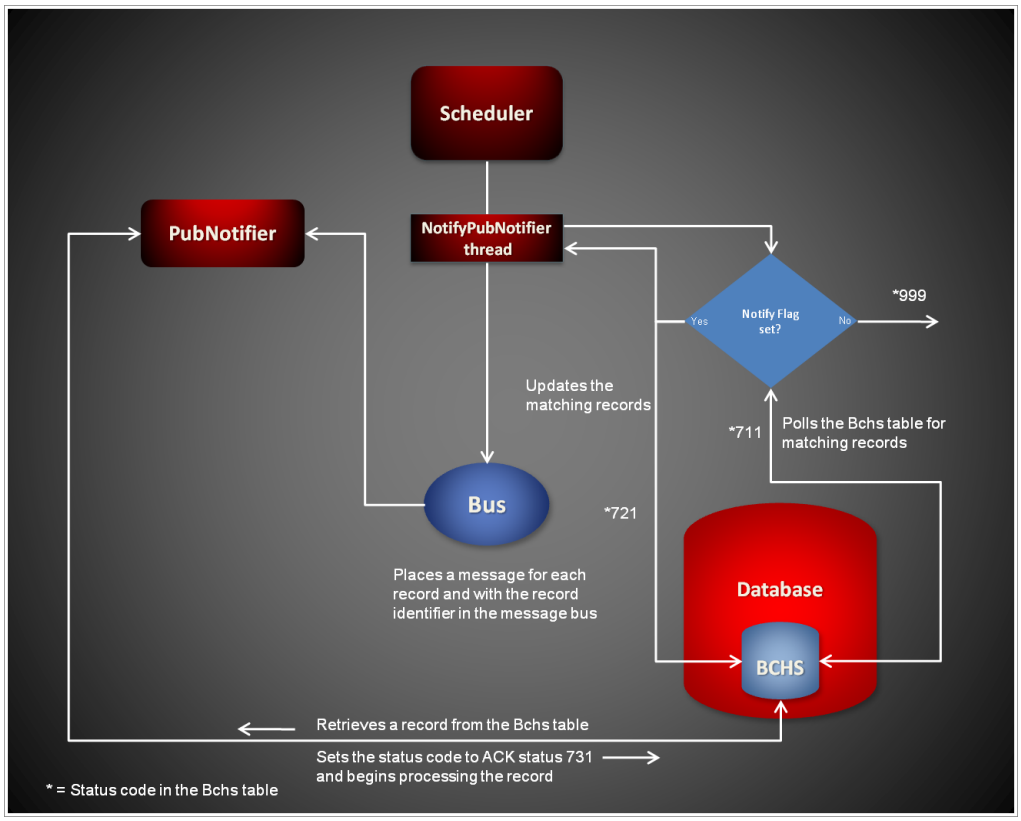


THE NOTIFYPUBNOTIFIER THREAD

This thread monitors the Bchs table for records with a status code value of PubNotifier-Ready (711) and with a BCHENABLENTF column value of one (1). It then changes the status code value for each record that is ready to PubNotifier-InProgress (721) and places a notification message in the message bus for the PubNotifier process to indicate there is a Bchs table record ready for processing.

The notification message contains the record identifier value for the record that is ready. If a Bchs record contains a status code value of PubNotifier-Ready (711) but the BCHENABLENTF column value is zero (0), the NotifyPubNotifier thread changes the status code to Processing-Complete (999).

If there is an error during processing, the NotifyPubNotifier thread changes the status code value to PubNotifier-Error (741).



STARTING AND STOPPING THE SCHEDULER

To	Then
Verify the Scheduler is running	Verify there is a running process with the name docfactory_scheduler.
Start the Scheduler	Place the scheduler.jar file in the deploy directory of Document Factory.
Stop the Scheduler	Remove the scheduler.jar file from the deploy directory of Document Factory.

Note The scheduler.jar configuration file is uncompressed and deployed to the temp/scheduler directory. This directory becomes the working directory for the Scheduler. Any output, including Log4J output, uses this directory as the starting directory.

Configuring the Main Scheduler Thread

The configuration for the Scheduler is stored in these resources:

Resource	Contains the
scheduler.jar file	Minimal startup configuration information.
.bindings file	Java Naming and Directory Interface (JNDI) data sources.
APPCONFIGCONTEXT table	Configuration options.
ALCONFIGCONTEXT table	Configuration options for the Scheduler status codes and message bus.

scheduler.jar File

The scheduler.jar file is located in the \deploy subdirectory of the Document Factory. It contains these configuration resources:

Component	Description
deploy.properties	Contains the minimal startup configuration information.
log4j.xml	Used to capture Log4J diagnostic and error output during start up. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j/
log4j.dtd	Used by the log4j.xml file.

deploy.properties File

The deploy.properties file contains the minimal startup configuration options used to read the configuration for the Scheduler from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables. This file is extracted and placed in the temp\scheduler working directory.

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Scheduler configuration.
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Scheduler configuration.
application.id	The value of APP_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Scheduler configuration.
config	The configuration name for the Scheduler. The default is Scheduler. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Scheduler configuration.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.

Option	Description
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=2
config=Scheduler
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

log4j.xml File

The log4j.xml file is extracted and placed in the temp/scheduler working directory. The log4j.xml file contains loggers that are used during the start up of the Scheduler, prior to the Scheduler loading the Log4J configuration from the APPCONFIGCONTEXT table. See the Log4J configuration options in the *APPCONFIGCONTEXT Table* on page 110 for more information.

.bindings File

The .bindings file is located in the config\context subdirectory of the Document Factory. It contains the Java Naming and Directory Interface (JNDI) data sources used by the Scheduler. Each JNDI data source contains these configuration options:

Option	Description
ClassName	The data source fully-qualified class name. Use the javax.sql.DataSource value.
FactoryName	The data source factory fully-qualified class name. Use the org.apache.commons.dbcp.BasicDataSourceFactory value. The BasicDataSourceFactory class supports connection pooling.
driverClassName	The Java Database Connectivity (JDBC) driver class name.
url	The JDBC URL.
maxOpenPreparedStatements	The maximum number of prepared statements to cache in the connection pool. Use the value -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	How often the idle object evictor thread should run and perform clean up of the stale connection handles. Use the value -1 to disable the idle object evictor thread.
validationQuery	A validation query that should be run when borrowing objects from the connection pool.
username	The JDBC user name.

Option	Description
password	The JDBC password.
testOnBorrow	Set this option to Yes if validationQuery should be used when borrowing an object from the connection pool. The default is No.
initialSize	The initial connection pool size.
maxActive	The maximum number of active connections in the pool.
maxIdle	The maximum number of idle connections in the pool.
minIdle	The minimum number of idle connections in the pool.
maxWait	The maximum time (in milliseconds) to wait for a connection object to be retrieved from the pool before issuing an error.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type= maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
```

```
DMKRConfig/RefAddr/9/Type=maxWait
#Unix friendly Documaker Doc. Factory JNDI DataSource
DMKRFactory/ClassName=javax.sql.DataSource
DMKRFactory/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRFactory/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRFactory/RefAddr/0/Encoding=String
DMKRFactory/RefAddr/0/Type=driverClassName
DMKRFactory/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRFactory/RefAddr/1/Encoding=String
DMKRFactory/RefAddr/1/Type=url
DMKRFactory/RefAddr/10/Content=-1
DMKRFactory/RefAddr/10/Encoding=String
DMKRFactory/RefAddr/10/Type=maxOpenPreparedStatements
DMKRFactory/RefAddr/11/Content=-1
DMKRFactory/RefAddr/11/Encoding=String
DMKRFactory/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRFactory/RefAddr/12/Content=select 1 from dual
DMKRFactory/RefAddr/12/Encoding=String
DMKRFactory/RefAddr/12/Type=validationQuery
DMKRFactory/RefAddr/2/Content=dmkr_asline
DMKRFactory/RefAddr/2/Encoding=String
DMKRFactory/RefAddr/2/Type=username
DMKRFactory/RefAddr/3/Content=oracle12
DMKRFactory/RefAddr/3/Encoding=String
DMKRFactory/RefAddr/3/Type=password
DMKRFactory/RefAddr/4/Content=true
DMKRFactory/RefAddr/4/Encoding=String
DMKRFactory/RefAddr/4/Type=testOnBorrow
DMKRFactory/RefAddr/5/Content=1
DMKRFactory/RefAddr/5/Encoding=String
DMKRFactory/RefAddr/5/Type=initialSize
DMKRFactory/RefAddr/6/Content=8
DMKRFactory/RefAddr/6/Encoding=String
DMKRFactory/RefAddr/6/Type= maxActive
DMKRFactory/RefAddr/7/Content=8
DMKRFactory/RefAddr/7/Encoding=String
DMKRFactory/RefAddr/7/Type=maxIdle
DMKRFactory/RefAddr/8/Content=0
DMKRFactory/RefAddr/8/Encoding=String
DMKRFactory/RefAddr/8/Type=minIdle
DMKRFactory/RefAddr/9/Content=60000
DMKRFactory/RefAddr/9/Encoding=String
DMKRFactory/RefAddr/9/Type=maxWait
```

Configuring the Main Scheduler Thread

The Scheduler thread reads configuration information from deploy.properties file and the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.

APPCONFIGCONTEXT Table

The options and values are read from this table when the Group_Name value is the *config* value specified in the deploy.properties file.

For example, if the config value in the deploy.properties file is *Scheduler*, the system uses the values in APPCONFIGCONTEXT table where the Group_Name is *Scheduler* when it starts the Scheduler process.

Option	Description
StartCommand	The start command. This value is used by the Supervisor to start the class specified in JavaClass configuration option. The default is Java.
StartArguments	The start arguments for JavaClass. There is no default.
JavaClass	The Java class that is used to start the worker class specified in WorkerClass configuration option. Use the oracle.documaker.process.ProcessShell value. ProcessShell class is a process shell that provides all functionality needed to communicate with the Supervisor process and to start and manage the worker class specified in WorkerClass configuration option.
JVMOptions	Any JVM options the Supervisor process uses to start JavaClass. There is no default.
Instances	(Optional) The number of instances the Supervisor should start for a process configuration. The default is two (2).
UseLoadBalancing	(Optional) This option controls wether the Supervisor checks the idle time of a processes' instances that are running and starts additional ones when all of them are busy. Instances are considered busy when their idle time is less than the value provided in the MinIdleTimeSeconds option. The Supervisor uses the value provided in the IdleTimeChecks option to determine the number of idle time checks to run before it starts additional instances. When additional instances are started for load balancing purposes, they are shut down by the Supervisor if their idle time exceeds the value in the MaxIdleTimeSeconds option. The maximum number of instances running is the value for the MaxInstances option (including the instances configured in the Instances option). The Supervisor checks the idle time of the current instances at the interval specified in the IdleTimeCheckIntervalSeconds option and if all are busy, it starts an additional number of instances equal to the value provided in the IncrementCount option. Please note that the Supervisor does not check the busy time of the current instances until the time provided in the IdleTimeCheckDelaySeconds option elapses. Make sure the value for the delay provides enough time for all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. You can enter Yes or No. The default is Yes.
MaxInstances	(Optional) This option controls the maximum number of instances that can run when the UseLoadBalancing option is enabled. The default is the number of processors times two.
IncrementCount	(Optional) This option controls how many additional instances are started during the current check when all instances running are busy and the UseLoadBalancing option is enabled. The default is two (2).

Option	Description
IdleTimeCheckIntervalSeconds	(Optional) This option controls how often the Supervisor checks the idle time of the instances that are running to determine if they are busy so it can start additional ones when the UseLoadBalancing option is enabled. The default is 10 seconds.
IdleTimeCheckDelaySeconds	(Optional) This option controls the initial delay before the first idle time check is performed by the Supervisor when the UseLoadBalancing option is enabled. This time period should allow all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. The default is 120 seconds.
IdleTimeChecks	(Optional) This option defines the number of consecutive idle time checks that must fail, meaning all instances were busy during each check, before more instances are started when the UseLoadBalancing option is enabled. Each check takes place at the IdleTimeCheckIntervalSeconds interval. The default is 12.
MinIdleTimeSeconds	(Optional) This option controls the minimum idle time for each instance. The idle time represents how long it has been since an instance processed the last request. If the Supervisor detects an instance has an idle time less than the value provided for this option, it considers it busy for the purpose of load balancing. The default is 5 seconds.
MaxIdleTimeSeconds	(Optional) This option controls the maximum idle time for an additional instance. The idle time represents how long it has been since an instance performed processing. If the Supervisor detects an instance, which was started for the purpose of load balancing, has reached an idle time greater than the value provided for this option, it sends the instance a shutdown request. The default is 120 seconds.
MaxTransactions	(Optional) This option controls the maximum number of transactions an instance can process before it is restarted by the Supervisor. The default is -1, which disables this option.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before it is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
OrderedRestartIntervalSeconds	(Optional) This option controls the interval used for restarting each process instance in a sequential/ordered manner when the MaxTransactions or MaxUpTime options are used. The Supervisor restarts one instance at a time and waits for an amount of time equal to the value specified for this option before it restarts the next one and so on until it has restarted all of them. If you set this option to less than 60 seconds, you can negatively affect performance. The default is 60 seconds.
WatchList	A comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted.
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is 5.
WorkerClass	The class that extends the oracle.documaker.process.worker.Worker Thread class and is started by the class specified in JavaClass configuration option. Use the oracle.documaker.scheduler.Scheduler value.

Option	Description
WorkerThreads	How many threads of WorkerClass should be created by JavaClass. You can use the value 1. The default is one (1).
WorkerIntervalMillis	How often each WorkerClass thread should perform its work. The default is 5000 milliseconds.
WorkerStartDelayMillis	How long each WorkerClass thread should wait after startup and before performing any work. The default is 10000 milliseconds.
ShutdownHookClass	The class that extends the oracle.documaker.process.shutdown.ShutdownHook class. Use the oracle.documaker.scheduler.shutdown.SchedulerShutdownHook value.
HouseKeeperClass	The class that extends the oracle.documaker.process.housekeeping.HouseKeeper class. Use the oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper value.
HouseKeeperIntervalMillis	How often the HouseKeeperClass thread should perform its work. The default is 15000 milliseconds.
HouseKeeperStartDelayMillis	How long the HouseKeeperClass thread should wait after startup and before performing any work. The default is 30000 milliseconds.
IPCIntervalMillis	How often the inter-process communication (IPC) thread should perform its work. This option is used by JavaClass to report back to the Supervisor process. The default is 1000 milliseconds.
IPCStartDelayMillis	How long the inter-process communication (IPC) thread should wait after startup and before performing any work. This option is used by JavaClass to report back to the Supervisor process. The default is 10000 milliseconds.
Log4jIntervalMillis	How often the Log4J resource monitor thread should perform its work. This option is used to monitor log4j.xml file deployed under temp\scheduler working directory and reload it when a change is detected. The default is 1000 milliseconds.
Log4jStartDelayMillis	How long the Log4J resource monitor thread should wait after startup and before performing any work. This option is used to monitor log4j.xml file deployed under temp\scheduler working directory and reload it when a change is detected. The default is 10000 milliseconds.

Here is an example:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/jre/bin/docfactory_scheduler
JavaClass	oracle.documaker.process.ProcessShell
JVMOptions	-Xmx128m
Instances	1
UseLoadBalancing	No
WorkerClass	oracle.documaker.scheduler.Scheduler

InstallationLocation = The installation location where you installed Document Factory.

Option	Value
WorkerThreads	1
WorkerIntervalMillis	5000
WorkerStartDelayMillis	5000
ShutdownHookClass	oracle.documaker.scheduler.shutdown.SchedulerShutdownHook
HouseKeeperClass	oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper
HouseKeeperIntervalMillis	5000
HouseKeeperStartDelayMillis	10000
IPCIntervalMillis	1000
IPCStartDelayMillis	1000
Log4jIntervalMillis	5000
Log4jStartDelayMillis	10000

InstallationLocation = The installation location where you installed Document Factory.

Log4J configuration options

For specific information on the Log4J configuration options, see *Defining Log4J Configuration Options* on page 277.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Identifier-Ready	This is the status code value that tells the Scheduler process that transactions are ready for the Identifier. The default is 111.

Here is an example:

Option	Value
Identifier-Ready	111

Configuring the Housekeeper Thread

The SchedulerHouseKeeper thread reads configuration information from deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Identifier-ACK	The acknowledgement value used by the Identifier to set JOBSTATUS. The default is 131.
Identifier-Error	The error value used by the Identifier to set JOBSTATUS. The default is 141.
Assembler-ACK	The acknowledgement value used by the Assembler to set TRNSTATUS. The default is 231.
Assembler-Error	The error value used by the Assembler to set TRNSTATUS. The default is 241.
Distributor-ACK	The acknowledgement value used by the Distributor to set TRNSTATUS. The default is 331.
Distributor-Error	The error value used by the Distributor to set TRNSTATUS. The default is 341.
Presenter-ACK	The acknowledgement value used by the Presenter to set TRNSTATUS and BCHSTATUS columns. The default is 431.
Presenter-Error	The error value used by the Presenter to set TRNSTATUS and BCHSTATUS columns. The default is 441.
Archiver-ACK	The acknowledgement value used by the Archiver to set BCHSTATUS column. The default is 531.
Archiver-Error	The error value used by the Archiver to set BCHSTATUS column. The default is 541.
Publisher-ACK	The acknowledgement value used by the Publisher to set BCHSTATUS column. The default is 631.
Publisher-Error	The error value used by the Publisher to set BCHSTATUS column. The default is 641.
PubNotifier-ACK	The acknowledgement value used by the PubNotifier to set BCHSTATUS column. The default is 731.
PubNotifier-Error	The error value used by the PubNotifier to set BCHSTATUS column. The default is 741.

Here is an example:

Option	Value
Identifier-ACK	131
Identifier-Error	141
Assembler-ACK	231
Assembler-Error	241
Distributor-ACK	331
Distributor-Error	341
Presenter-ACK	431
Presenter-Error	441
Archiver-ACK	531
Archiver-Error	541

Option	Value
Publisher-ACK	631
Publisher-Error	641
PubNotifier-ACK	731
PubNotifier-Error	741

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *HouseKeeper*:

Note The status values are shown in sequence to provide a frame of reference in terms of processing and when errors occur. To configure Document Factory, you can use numeric values/status values in between the defaults provided to interject another step in the assembly line process.

Option	Description
FetchSize	How many records to query at one time. The default is 5.
TransactionTimeoutMillis	This is the expiration time, in milliseconds, that determines when the Housekeeper thread changes an ACK status code to an *Error* status code for a TRNS or BCHS record. The Housekeeper thread compares this value to the Trns.MODIFYTIME and Bchs.BCHMODIFYTIME column values to determine if a record with an ACK status code should be updated to an Error status code. For example, if the Housekeeper thread finds a TRNS record with a status code of Identifier- ACK (131) and the value for this option is set to 360000 milliseconds, but the MODIFYTIME column value for the TRNS record indicates the last time the record was modified with the ACK status has exceeded the timeout value for this option, then The HouseKeeper thread will set the status code to Identifier-Error (141). The default is 360000 milliseconds.

Here is an example:

Option	Value
FetchSize	5
TransactionTimeoutMillis	360000

CONFIGURING SCHEDULER WORKER THREADS

Setting up the various Scheduler worker threads include:

- *Configuring the NotifyArchiver Thread* on page 116
- *Configuring the NotifyAssembler Thread* on page 118
- *Configuring the NotifyDistributor Thread* on page 119

Configuring the NotifyArchiver Thread

The NotifyArchiver thread reads configuration information from deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Archiver-Ready	This is the status code that lets the NotifyArchiver thread know a recipient batch record or row in the BCHS table is ready for processing. The default is 511.
Archiver-InProgress	This is the status code that indicates a recipient batch record or row in the BCHS table has been sent to the Archiver. The default is 521.
Archiver-Error	This is the status code that indicates a recipient batch record or row in the BCHS table was processed by the NotifyArchiver thread but it encountered an error. The default is 541.
Publisher-Ready	This is the status code that indicates a recipient batch record or row in the BCHS table is ready to be processed by the NotifyPublsher thread. The default is 611.

Here is an example:

Option	Value
Archiver-Ready	511
Archiver-InProgress	521
Archiver-Error	541
Publisher-Ready	611

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
ArchiverQueue	The name of the queue used to notify the Archiver.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf
ArchiverQueue	jms/archiver_requestq
TimeoutSeconds	5

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *NotifyArchiver*:

Option	Description
IntervalMillis	How often, in milliseconds, the NotifyArchiver thread should perform its work. The default is the value provided in WorkerIntervalMillis option for the Scheduler (main) thread.
StartDelayMillis	How long, in milliseconds, the NotifyArchiver thread should wait after startup and before performing any work. The default is the value provided in WorkerStartDelayMillis option for the Scheduler (main) thread.
FetchSize	How many records to query at one time. The default is 5.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5

Configuring the NotifyAssembler Thread

The NotifyAssembler thread reads configuration information from deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Assembler-Ready	This is the status code that lets the NotifyAssembler thread know a transaction is ready for processing. The default is 211.
Assembler-InProgress	This is the status code that indicates a transaction has been sent to the Assembler. The default is 221.
Assembler-Error	This is the status code that indicates a transaction was processed by the NotifyAssembler thread but it encountered an error. The default is 241.

Here is an example:

Option	Value
Assembler-Ready	211
Assembler-InProgress	221
Assembler-Error	241

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
AssemblerQueue	The name of the queue used to notify the Assembler.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf

Option	Value
AssemblerQueue	jms/assembler_requestq
TimeoutSeconds	5

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
IntervalMillis	How often, in milliseconds, the NotifyAssembler thread should perform its work. The default is the value provided in WorkerIntervalMillis option for the Scheduler (main) thread.
StartDelayMillis	How long, in milliseconds, the NotifyAssembler thread should wait after startup and before performing any work. The default is the value provided in WorkerStartDelayMillis option for the Scheduler (main) thread.
FetchSize	How many records to query at one time. The default is 5.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5

Configuring the NotifyDistributor Thread

The NotifyDistributor thread reads configuration information from the deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Distributor-Ready	This is the status code that lets the NotifyDistributor thread know a transaction is ready for processing. The default is 311.
Distributor-InProgress	This is the status code that indicates a transaction has been sent to the Distributor. The default is 321.
Distributor-Error	This is the status code that indicates a transaction was processed by the NotifyDistributor thread but it encountered an error. The default is 341.

Here is an example:

Option	Value
Distributor-Ready	311
Distributor-InProgress	321
Distributor-Error	341

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
DistributorQueue	The name of the queue used to notify the Distributor.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf
DistributorQueue	jms/distributor_requestq
TimeoutSeconds	5

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *NotifyDistributor*:

Option	Description
IntervalMillis	How often, in milliseconds, the NotifyDistributor thread should perform its work. The default is the value provided in WorkerIntervalMillis option for the Scheduler (main) thread.
StartDelayMillis	How long, in milliseconds, the NotifyDistributor thread should wait after startup and before performing any work. The default is the value provided in WorkerStartDelayMillis option for the Scheduler (main) thread.
FetchSize	How many records to query at one time. The default is 5.

Option	Description
StatusCode	The status code tells the system a transaction is ready to be sent to the Distributor. This is the StatusCode column in the TRNS table. It is used in addition to the TRNSTATUS value of 311 to indicate that the transaction is ready for distribution processing. The default is B.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5
StatusCode	B

Configuring the NotifyIdentifier Thread

The NotifyIdentifier thread reads configuration information from deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Identifier-Ready	This is the status code that lets the NotifyIdentifier thread know a transaction is ready for processing. The default is 111.
Identifier-InProgress	This is the status that indicates a transaction has been sent to the Identifier. The default is 121.
Identifier-Error	This is the status code that indicates a transaction was processed by the NotifyIdentifier thread but it encountered an error. The default is 141.

Here is an example:

Option	Value
Identifier-Ready	111
Identifier-InProgress	121
Identifier-Error	141

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
IdentifierQueue	The name of the queue used to notify the Identifier.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf
IdentifierQueue	jms/identifier_requestq
TimeoutSeconds	5

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *NotifyIdentifier*:

Option	Description
IntervalMillis	How often, in milliseconds, the NotifyIdentifier thread should perform its work. The default is the value provided in WorkerIntervalMillis option for the Scheduler (main) thread.
StartDelayMillis	How long, in milliseconds, the NotifyIdentifier thread should wait after startup and before performing any work. The default is the value provided in WorkerStartDelayMillis option for the Scheduler (main) thread.
FetchSize	The number of records to retrieve. The default is 5.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5

Configuring the NotifyPresenterImmediate Thread

The NotifyPresenterImmediate thread reads configuration information from deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Presenter-InProgress	This is the status code that indicates a transaction has been sent to the Presenter. The default is 421.
Batcher-InProgress	This is the status code that indicates a transaction has been processed by the Batcher and it is ready for processing by the NotifyPresenterImmediate thread. The default is 415.
Presenter-Error	This is the status code that indicates a transaction has been processed by the NotifyPresenterImmediate thread but an error was encountered. The default is 441.

Here is an example:

Option	Value
Presenter-InProgress	421
Batcher-InProgress	415
Presenter-Error	441

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
PresenterQueue	The name of the queue used to notify the Presenter.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001

Option	Value
jms.qcf.name	jms/qcf
PresenterQueue	jms/presenter_requestq
TimeoutSeconds	5

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *NotifyPresenterImmediate*:

Option	Description
IntervalMillis	How often, in milliseconds, the NotifyPresenterImmediate thread should perform its work. The default is the value provided in the WorkerIntervalMillis option for the Scheduler (main) thread.
StartDelayMillis	How long, in milliseconds, the NotifyPresenterImmediate thread should wait after startup and before performing any work. The default is the value provided in the WorkerStartDelayMillis option for the Scheduler (main) thread.
FetchSize	How many records to query at one time. The default is 5.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5

Configuring the NotifyPresenterScheduled Thread

The NotifyPresenterScheduled thread reads configuration information from deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Presenter-Ready	This is the status code that lets the NotifyPresenterScheduled thread know a transaction is ready for processing. The default is 411.
Presenter-InProgress	This is the status code that indicates a transaction has been sent to the Presenter. The default is 421.
Presenter-Error	This is the status code that indicates a transaction has been processed by the NotifyPresenterScheduled thread but an error was encountered. The default is 441.

Here is an example:

Option	Value
Presenter-Ready	411
Presenter-InProgress	421
Presenter-Error	441

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
PresenterQueue	The name of the queue used to notify the Presenter.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf
PresenterQueue	jms/presenter_requestq
TimeoutSeconds	5

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *NotifyPresenterScheduled*:

Option	Description
IntervalMillis	How often the NotifyPresenterScheduled thread should perform its work. The default is the value provided in WorkerIntervalMillis configuration option for the Scheduler (main) thread.
StartDelayMillis	How long the NotifyPresenterScheduled thread should wait after startup and before performing any work. The default is the value provided in WorkerStartDelayMillis configuration option for the Scheduler (main) thread.
FetchSize	How many records to query at one time. The default is 5.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5

Configuring the NotifyPublisher Thread

The NotifyPublisher thread reads configuration information from deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Publisher-Ready	This is the status code that lets the NotifyPublisher thread know a transaction is ready for processing. The default is 611.
Publisher-InProgress	This is the status code that indicates a transaction has been sent to the Publisher. The default is 621.
Publisher-Error	This is the status code that indicates a transaction was processed by the NotifyPublisher thread but it encountered an error. The default is 641.
PubNotifier-Ready	This is the status code that lets the NotifyPubNotifier thread know a transaction is ready for processing. The default is 711.

Here is an example:

Option	Value
Publisher-Ready	611
Publisher-InProgress	621
Publisher-Error	641
PubNotifier-Ready	711

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
PublisherQueue	The name of the queue used to notify the Publisher.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf
PublisherQueue	jms/publisher_requestq
TimeoutSeconds	5

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *NotifyPublisher*:

Option	Description
IntervalMillis	How often, in milliseconds, the NotifyPublisher thread should perform its work. The default is the value provided in WorkerIntervalMillis option for the Scheduler (main) thread.
StartDelayMillis	How long, in milliseconds, the NotifyPublisher thread should wait after startup and before performing any work. The default is the value provided in WorkerStartDelayMillis option for the Scheduler (main) thread.
FetchSize	How many records to query at one time. The default is 5.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5

Configuring the NotifyPubNotifier Thread

The NotifyPubNotifier thread reads configuration information from deploy.properties file and the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
PubNotifier-Ready	This is the status code that lets the NotifyPubNotifier thread know a transaction is ready for processing. The default is 711.
PubNotifier-InProgress	This is the status code that indicates a transaction has been sent to the PubNotifier. The default is 721.
PubNotifier-Error	This is the status code that indicates a transaction was processed by the NotifyPubNotifier thread but it encountered an error. The default is 741.
Processing-Complete	This is the status code that indicates a transaction is complete. The default is 999.

Here is an example:

Option	Value
PubNotifier-Ready	711
PubNotifier-InProgress	721
PubNotifier-Error	741
Processing-Complete	999

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
PubNotifierQueue	The name of the queue used to notify the PubNotifier.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf

Option	Value
PubNotifierQueue	jms/pubnotifier_requestq
TimeoutSeconds	5

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *NotifyPubNotifier*:

Option	Description
IntervalMillis	How often, in milliseconds, the NotifyPubNotifier thread should perform its work. The default is the value provided in WorkerIntervalMillis option for the Scheduler (main) thread.
StartDelayMillis	How long, in milliseconds, the NotifyPubNotifier thread should wait after startup and before performing any work. The default is the value provided in WorkerStartDelayMillis option for the Scheduler (main) thread.
FetchSize	How many records to query at one time. The default is 5.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5

SCHEDULER STATUS CODES

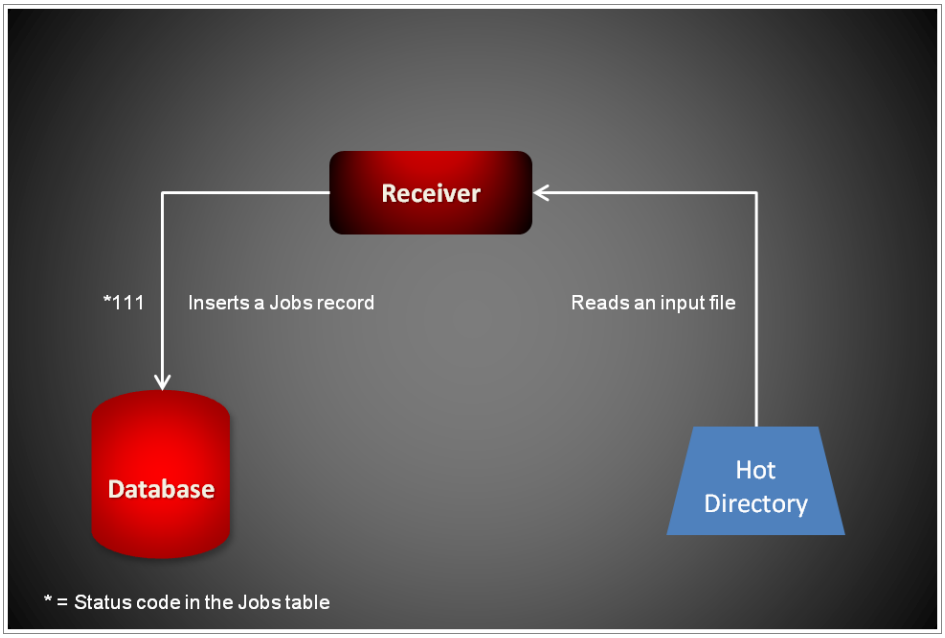
These codes are read from the Status GROUP_NAME column in the ALCONFIGCONTEXT table.

Option	Value	Updated by	Read by
Identifier-Ready	111	Receiver	Scheduler-NotifyIdentifier
Identifier-InProgress	121	Scheduler-NotifyIdentifier	
Identifier-ACK	131	Identifier	Scheduler-HouseKeeper
Identifier-Error	141	Identifier	
Assembler-Ready	211	Identifier	Scheduler-NotifyAssembler
Assembler-InProgress	221	Scheduler-NotifyAssembler	
Assembler-ACK	231	Assembler	Scheduler-HouseKeeper
Assembler-Error	241	Assembler	
Distributor-Ready	311	Assembler	Scheduler-NotifyDistributor
Distributor-InProgress	321	Scheduler-NotifyDistributor	
Distributor-ACK	331	Distributor	Scheduler-HouseKeeper
Distributor-Error	341	Distributor	
Presenter-Ready	411	Distributor	Batcher, Scheduler-NotifyPresenterScheduled
Batcher-InProgress	415	Batcher	Scheduler-NotifyPresenterImmediate
Presenter-InProgress	421	Scheduler-NotifyPresenterScheduled, Scheduler-NotifyPresenterImmediate	
Presenter-ACK	431	Presenter	Scheduler-HouseKeeper
Presenter-Error	441	Presenter	
Archiver-Ready	511	Presenter	Scheduler-NotifyArchiver
Archiver-InProgress	521	Scheduler-NotifyArchiver	
Archiver-ACK	531	Archiver	Scheduler-HouseKeeper
Archiver-Error	541	Archiver	
Publisher-Ready	611	Archiver, Scheduler-NotifyArchiver	Scheduler-NotifyPublisher
Publisher-InProgress	621	Scheduler-NotifyPublisher	
Publisher-ACK	631	Publisher	Scheduler-HouseKeeper
Publisher-Error	641	Publisher	

Option	Value	Updated by	Read by
Publisher-Transferring	651	Publisher	
Publisher-Publishing	661	Publisher	
Publisher-Cancelled	671	Publisher	
Publisher-Unknown	681	Publisher	
Publisher-Success	690	Publisher	
PubNotifier-Ready	711	Publisher, Scheduler-NotifyPublisher	Scheduler-NotifyPubNotifier
PubNotifier-InProgress	721	Scheduler-NotifyPubNotifier	
PubNotifier-ACK	731	PubNotifier	Scheduler-HouseKeeper
PubNotifier-Error	741	PubNotifier	
PubNotifier-Next	999	PubNotifier	
ProcessingError	941		
ProcessingComplete	999	Scheduler-NotifyPubNotifier	

CONFIGURING THE RECEIVER

The Receiver process reads the extract data for a transaction from an input file and creates a Jobs table record. The Supervisor process deploys and manages the Receiver process. The Receiver process monitors the hot directories you define for input files to process. The Receiver process is the first process in the assembly line to process a transaction.

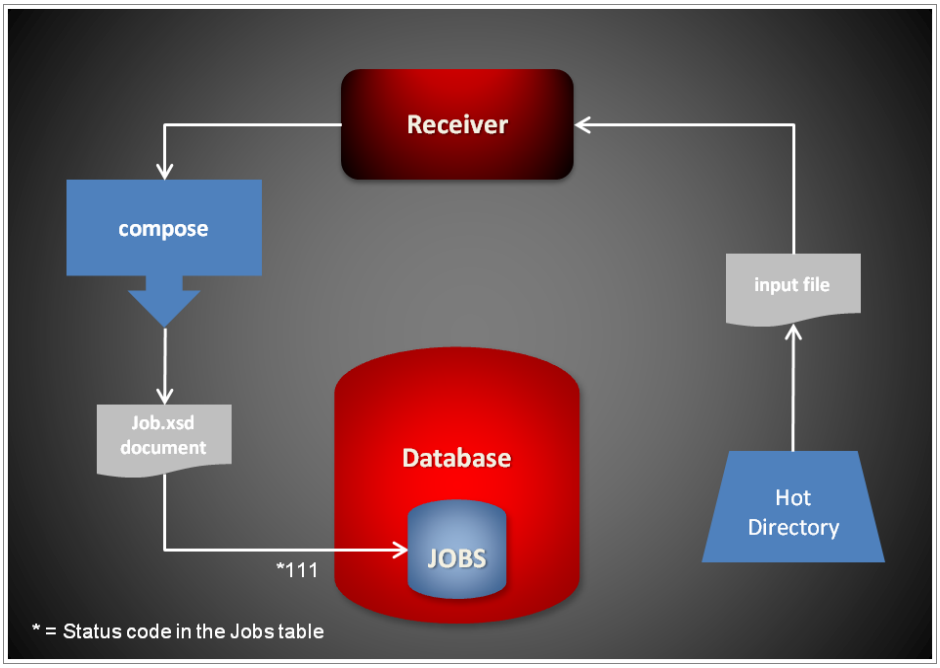


Note The Receiver can also accept jobs from the doPublishFromImport web service. See *Using Publishing Services* on page 443 for more information.

The Receiver reads an input file from the hot directory and converts it into a Job.xsd-compliant XML file that contains the extract data. It then inserts this file and its extract data into a new record the Jobs table.

Note See *Input Formats* on page 142 for more information.

The Receiver then updates the status code for the Jobs record so the Scheduler process can notify the next process in the assembly line. This value comes from the ALCONFIGCONTEXT table, Identifier - Ready status value (111).



STARTING AND STOPPING THE RECEIVER

To	Then
Verify the Receiver is running.	Verify there is a running process with the name docfactory_receiver.
Start the Receiver	Place the receiver.jar file in the deploy directory of Document Factory.
Stop the Receiver	Remove the receiver.jar file from the deploy directory of Document Factory.

Note The receiver.jar configuration file is uncompressed and deployed to the temp\receiver directory. This directory becomes the working directory for the Receiver. All output, including Log4J output, uses this directory as the starting directory.

USING RECEIVER CONFIGURATION RESOURCES

The configuration information for the Receiver is stored in these resources:

Resource	Contains the
receiver.jar file	minimal startup configuration information.
.bindings file	Java Naming and Directory Interface (JNDI) data sources.
APPCONFIGCONTEXT table	Configuration options.
ALCONFIGCONTEXT table	Configuration options for the Receiver status codes.

receiver.jar File

The receiver.jar file is located in the \deploy subdirectory of the Document Factory. It contains these configuration resources:

Component	Description
deploy.properties	Contains the minimal startup configuration information.
log4j.xml	Used to capture Log4J diagnostic and error output during start up. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j/
log4j.dtd	Used by the log4j.xml file.

deploy.properties File

The deploy.properties file is extracted and placed in the temp\receiver working directory. This file contains the minimal startup configuration options used to read the configuration for the Receiver from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables:

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Receiver configuration.
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Receiver configuration.
application.id	The value of APP_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Receiver configuration.
config	The configuration name for the Receiver. The default is Receiver. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Receiver configuration.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.

Option	Description
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=3
config=Receiver
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

log4j.xml File

The log4j.xml file is extracted and placed in the temp/receiver working directory. The log4j.xml file contains loggers that are used during start up of the Receiver, prior to the Receiver loading the Log4J configuration from the APPCONFIGCONTEXT table. See the Log4J configuration options in the *APPCONFIGCONTEXT Table* on page 137 for more information.

.bindings File

The .bindings file is located in the config\context subdirectory of the Document Factory. It contains the Java Naming and Directory Interface (JNDI) data sources used by the Receiver. Each JNDI data source contains these configuration options:

Option	Description
ClassName	The fully-qualified class name for the data source. Use the javax.sql.DataSource value.
FactoryName	The data source factory fully-qualified class name. Use the org.apache.commons.dbcp.BasicDataSourceFactory value. The BasicDataSourceFactory class supports connection pooling.
driverClassName	The Java Database Connectivity (JDBC) driver class name.
url	The JDBC URL.
maxOpenPreparedStatements	The maximum number of prepared statements to cache in the connection pool. Use the value -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	How often the idle object evictor thread should run and perform a clean up of the stale connection handles. Use the value -1 to disable the idle object evictor thread.
validationQuery	A validation query that should be run when borrowing objects from the connection pool.

Option	Description
username	The JDBC user name.
password	The JDBC password.
testOnBorrow	Set this option to Yes if validationQuery should be used when borrowing an object from the connection pool. The default is No.
initialSize	The initial connection pool size.
maxActive	The maximum number of active connections in the pool.
maxIdle	The maximum number of idle connections in the pool.
minIdle	The minimum number of idle connections in the pool.
maxWait	The maximum time (in milliseconds) to wait for a connection object to be retrieved from the pool before issuing an error.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type= maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
```

```
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
DMKRConfig/RefAddr/9/Type=maxWait
#Unix friendly Documaker Doc. Factory JNDI DataSource
DMKRFactory/ClassName=javax.sql.DataSource
DMKRFactory/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRFactory/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRFactory/RefAddr/0/Encoding=String
DMKRFactory/RefAddr/0/Type=driverClassName
DMKRFactory/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRFactory/RefAddr/1/Encoding=String
DMKRFactory/RefAddr/1/Type=url
DMKRFactory/RefAddr/10/Content=-1
DMKRFactory/RefAddr/10/Encoding=String
DMKRFactory/RefAddr/10/Type=maxOpenPreparedStatements
DMKRFactory/RefAddr/11/Content=-1
DMKRFactory/RefAddr/11/Encoding=String
DMKRFactory/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRFactory/RefAddr/12/Content=select 1 from dual
DMKRFactory/RefAddr/12/Encoding=String
DMKRFactory/RefAddr/12/Type=validationQuery
DMKRFactory/RefAddr/2/Content=dmkr_asline
DMKRFactory/RefAddr/2/Encoding=String
DMKRFactory/RefAddr/2/Type=username
DMKRFactory/RefAddr/3/Content=oracle12
DMKRFactory/RefAddr/3/Encoding=String
DMKRFactory/RefAddr/3/Type=password
DMKRFactory/RefAddr/4/Content=true
DMKRFactory/RefAddr/4/Encoding=String
DMKRFactory/RefAddr/4/Type=testOnBorrow
DMKRFactory/RefAddr/5/Content=1
DMKRFactory/RefAddr/5/Encoding=String
DMKRFactory/RefAddr/5/Type=initialSize
DMKRFactory/RefAddr/6/Content=8
DMKRFactory/RefAddr/6/Encoding=String
DMKRFactory/RefAddr/6/Type= maxActive
DMKRFactory/RefAddr/7/Content=8
DMKRFactory/RefAddr/7/Encoding=String
DMKRFactory/RefAddr/7/Type=maxIdle
DMKRFactory/RefAddr/8/Content=0
DMKRFactory/RefAddr/8/Encoding=String
DMKRFactory/RefAddr/8/Type=minIdle
DMKRFactory/RefAddr/9/Content=60000
DMKRFactory/RefAddr/9/Encoding=String
DMKRFactory/RefAddr/9/Type=maxWait
```

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Receiver*:

Option	Description
StartCommand	The start command. This value is used by the Supervisor to start the class specified in JavaClass configuration option. The default is Java.
StartArguments	The start arguments for JavaClass. There is no default.

Option	Description
JavaClass	The Java class used to start the worker class specified in WorkerClass configuration option. Use the oracle.documaker.process.ProcessShell value. ProcessShell class is a process shell that provides all functionality needed to communicate with the Supervisor process and to start and manage the worker class specified in WorkerClass configuration option.
JVMOptions	Any JVM options the Supervisor process uses to start JavaClass. There is no default.
Instances	(Optional) The number of instances the Supervisor should start for a process configuration. The default is two (2).
UseLoadBalancing	(Optional) This option controls wether the Supervisor checks the idle time of a process's instances that are running and starts additional ones when all of them are busy. Instances are considered busy when their idle time is less than the value provided in the MinIdleTimeSeconds option. The Supervisor uses the value provided in the IdleTimeChecks option to determine the number of idle time checks to run before it starts additional instances. When additional instances are started for load balancing purposes, they are shut down by the Supervisor if their idle time exceeds the value in the MaxIdleTimeSeconds option. The maximum number of instances running is the value for the MaxInstances option (including the instances configured in the Instances option). The Supervisor checks the idle time of the current instances at the interval specified in the IdleTimeCheckIntervalSeconds and if all are busy, it starts an additional number of instances equal to the value provided in the IncrementCount option. Please note that the Supervisor does not start checking the busy time of the current instances until the time provided in the IdleTimeCheckDelaySeconds option elapses. Make sure the value for the delay is ample enough to provide for all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. You can enter Yes or No. The default is Yes.
MaxInstances	(Optional) This option controls the maximum number of instances that can run when the UseLoadBalancing option is enabled. The default is the number of processors times two.
IncrementCount	(Optional) This option controls how many additional instances are started during the current check when all instances running are busy and the UseLoadBalancing option is enabled. The default is two (2).
IdleTimeCheckIntervalSeconds	(Optional) This option controls how often the Supervisor checks the idle time of the instances that are running to determine if they are busy so it can start additional ones when the UseLoadBalancing option is enabled. The default is 10 seconds.
IdleTimeCheckDelaySeconds	(Optional) This option controls the initial delay before the first idle time check is performed by the Supervisor when the UseLoadBalancing option is enabled. This time should be ample enough to allow all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. The default is 120 seconds.

Option	Description
IdleTimeChecks	(Optional) This option defines the number of consecutive idle time checks that must fail, meaning all instances were busy during each check, before more instances are started when the UseLoadBalancing option is enabled. Each check takes place at the IdleTimeCheckIntervalSeconds interval. The default is 12.
MinIdleTimeSeconds	(Optional) This option controls the minimum idle time for each instance. The idle time represents how long it has been since an instance processed the last request. If the Supervisor detects an instance has an idle time less than the value provided for this option, it considers it busy for the purpose of load balancing. The default is 5 seconds.
MaxIdleTimeSeconds	(Optional) This option controls the maximum idle time for an additional instance. The idle time represents how long it has been since an instance performed processing. If the Supervisor detects an instance, which was started for the purpose of load balancing, has reached an idle time greater than the value provided for this option, it sends the instance a shutdown request. The default is 120 seconds.
MaxTransactions	(Optional) This option controls the maximum number of transactions an instance can process before it is restarted by the Supervisor. The default is -1, which disables this option.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before it is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
OrderedRestartIntervalSeconds	(Optional) This option controls the interval used for restarting each process instance in a sequential/ordered manner when the MaxTransactions or MaxUpTime options are used. The Supervisor restarts one instance at a time and waits for an amount of time equal to the value specified for this option before it restarts the next one and so on until it has restarted all of them. If you set this option to less than 60 seconds, you can negatively affect performance. The default is 60 seconds.
WatchList	A comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted.
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is 5.
WorkerClass	The class that extends the oracle.documaker.process.worker.Worker class and is started by the class specified in JavaClass configuration option. Use the oracle.documaker.receiver.Receiver value.
WorkerThreads	How many threads of WorkerClass should be created by JavaClass. The default is 1.
WorkerIntervalMillis	How often, in milliseconds, each WorkerClass thread should perform its work. The default is 5000.

Option	Description
WorkerStartDelayMillis	How long, in milliseconds, each WorkerClass thread should wait before performing any work. The default is 10000.
ShutdownHookClass	The class that extends the oracle.documaker.process.shutdown.ShutdownHook class. Use the oracle.documaker.receiver.shutdown.ReceiverShutdownHook value.
IPCIntervalMillis	How often, in milliseconds, the inter-process communication (IPC) thread should perform its work. This option is used by JavaClass to report back to the Supervisor process. The default is 1000.
IPCStartDelayMillis	How long, in milliseconds, the inter-process communication (IPC) thread should wait before performing any work. This option is used by JavaClass to report back to the Supervisor process. The default is 10000.
Log4jIntervalMillis	How often, in milliseconds, the Log4J resource monitor thread should perform its work. The system uses this option to monitor the log4j.xml file deployed under temp\receiver working directory and reload it when a change is detected. The default is 1000.
Log4jStartDelayMillis	How long, in milliseconds, the Log4J resource monitor thread should wait before performing any work. The system uses this option to monitor the log4j.xml file deployed under temp\receiver working directory and reload it when a change is detected. The default is 10000.
HotDirectories	A comma-delimited list of directories that should be monitored for job import files by the Receiver thread.
TextDelimiter	A string and offset that serves as the delimiter to use for new transactions in flat files. Here is an example: HEADERREC,10 There is no default.
XMLDelimiter	The string expression and offset that serves as the delimiter to use for new transactions in a stacked XML file. Here is an example: <Diamond>,0 There is no default.
XPathDelimiter	The XPath expression that serves as the delimiter to use for new transactions in a valid XML file. Here is an example: //DOCUMENT There is no default.
OmitPI	This flag indicates whether the XML processing instruction should be omitted during output generation for each transaction XML. The default is No.
Indent	This flag indicates whether the XML should be indented during output generation for each transaction XML. The default is No.
StripWhiteSpace	This flag indicates whether white space should be removed from the XML during output generation for each transaction XML. The default is No.

Here is an example:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/jre/bin/docfactory_receiver
JavaClass	oracle.documaker.process.ProcessShell
JVMOptions	-Xmx128m -Duser.name=oracle
Instances	1
UseLoadBalancing	No
WorkerClass	oracle.documaker.receiver.Receiver
WorkerThreads	4
WorkerIntervalMillis	1000
WorkerStartDelayMillis	5000
ShutdownHookClass	oracle.documaker.receiver.shutdown.ReceiverShutdownHook
IPCIntervalMillis	1000
IPCStartDelayMillis	10000
Log4jIntervalMillis	5000
Log4jStartDelayMillis	10000
HotDirectories	/oracle_home/*InstallationLocation*/hotdirectory
TextDelimiter	HEADERREC,10
XMLDelimiter	<?xml,0
XPathDelimiter	//DOCUMENT
OmitPI	No
Indent	No
StripWhiteSpace	No

InstallationLocation = The installation location where you installed Document Factory.

Log4J configuration options

For specific information on the Log4J configuration options, see *Defining Log4J Configuration Options* on page 277.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Identifier-Ready	This is the status code value that tells the Scheduler process that transactions are ready for the Identifier. The default is 111.

Here is an example:

Option	Value
Identifier-Ready	111

INPUT FORMATS

These input formats are supported:

Input Type	Output Type	Description
Plain XML	Job.xsd based XML *	A plain XML file that contains one or more transactions. When multiple transactions are included, the file is parsed via xPath using the xPathDelimiter option. The XML for each transaction is extracted and added as the content of a new transaction element in the job object derived from Job.xsd.
Stacked XML	Job.xsd based XML *	A file that contains multiple Documaker XML files. Each Documaker XML file included in this file contains its own processing instruction and root element. This type of file is known as a Documaker stacked XML file.
Flat Files	Job.xsd based XML *	A flat extract file containing one or more transactions. The content for each transaction is parsed using the TextDelimiter option and added as the content of a new transaction element in the job object derived from Job.xsd.
Job.xsd based XML	Job.xsd based XML *	The XML is unmarshalled into a Job.xsd object.
JobRequest.xsd based XML	Job.xsd based XML *	The XML is unmarshalled into a JobRequest.xsd object and the job object is extracted.

* = All output is converted into a Job.xsd file based object using Java Architecture for XML Binding (JAXB) and then marshalled to XML, which is inserted in the JOBPAYLOADXML column. The actual content of each transaction in the job object is also inserted into a HashMap object which is then serialized and inserted into the JOBATTACHMENTS column.

Job Schema

This is the job object that is inserted in the Jobs table.

Element	Description	Type/Count
JOBUNIQUE_ID	The unique identifier for the new job.	string (0...1)

Element	Description	Type/Count
JOBPRIORITY	The job priority. Acceptable values are: 0 = immediate/highest priority 10 = normal/regular priority 20 = lowest priority The default is 10.	int (0...1)
Payload	The content of the new job.	Payload (0...1)

Payload

The payload contains the content of the new job.

Element	Description	Type/Count
(choice)	One of these: Transaction Extract (Data type)	choice (1)

Transaction

A transaction for the job.

Element	Description	Type/Count
UNIQUE_ID	The unique ID for the transaction.	string (0...1)
STATUSCODE	The status code for the transaction.	string (0...1)
ORIGUSER	The original user for the transaction.	string (0...1)
CURRUSER	The current user for the transaction.	string (0...1)
CURRGROUP	The current group for the transaction.	string (0...1)
CURRROLE	The current role for the transaction.	string (0...1)
CURRSUPER	The current super user for the transaction.	string (0...1)
TRANCODE	The transaction code for the transaction.	string (0...1)
APPROVALSTATE	The approval state for the transaction.	string (0...1)
DESCR	The description for the transaction.	string (0...1)
PROCESSNAME	The name of the process that created the transaction.	string (0...1)
RETENTION	How long the transaction should be retained.	dateTime (0...1)
Data	The data for the transaction.	Data (1)
TRNDATATYPE	The type of data in the Data element. Acceptable values are: 0=data in XML data type 1=data in BLOB	int (0...1)

Data

The data for a transaction.

Element	Description	Type/Count
Name	The name of the data (can be a file name).	string (0...1)
Content	The content of the data.	Content (1)

Content

Represents the content of a file attachment.

Element	Description	Type/Count
URI *	A file URI.	string(1)
Binary *	The binary content of the file attachment.	base64Binary (1)

* = URI and Binary elements are mutually exclusive.

Example job.xsd XML File

Here is an example of a job.xsd XML file. This XML file conforms to the job.xsd file. This means the job.xsd schema dictates how these XML files are generated and these XML files adhere to the job.xsd schema rules.

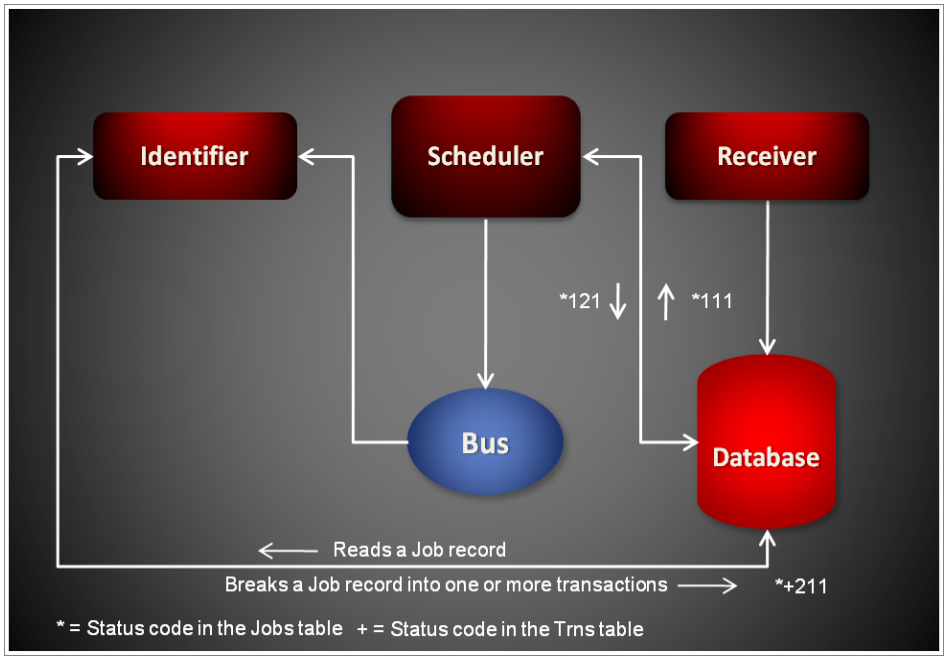
```
<?xml version="1.0" encoding="utf-8"?>
<Job
xmlns="oracle/documaker/schema/tables/jobs"
xmlns:trns="oracle/documaker/schema/tables/trns">
<Payload>
<Transaction>
<trns:Data>
<trns:Content>
<trns:Binary>
PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4KPERvY3VtZW50UmVx
dWVzdD4K
ICAgIDxQYWNrYWdlSW5mbz4KICAgICAgICA8S2V5MT5DRU5UUkFMPC9LZXkxPgogICAg
ICAgIDxL
ZXkyPkFDQ09VTlRfU1RBVFVTPC9LZXkyPgogICAgICAgIDxLZXlJRD4wMDAwMDAwPC9L
ZXlJRD4K
ICAgICAgICA8UnVuRGF0ZT4yMDA1MDgzMDwvUnVuRGF0ZT4KICAgICAgICA8VHJhbkNv
ZGU+UTwv
VHJhbkNvZGU+CiAgICAgICAgPFByb2R1Y3Q+Rm91bmRhdGlvbiBMaWZlPC9Qcm9kdWN0
PgogICAg
ICAgIDxQb2xpY3lOdW1iZXI+PC9Qb2xpY3lOdW1iZXI+CiAgICAgICAgPFBvbGljeUlz
c3VlRGF0
ZT4yMDA1MDIwMzwvUG9saWN5SXNzdWVEYXRlPgogICAgICAgIDxSZXRyb2FjdGl2ZURh
dGU+MjAw
NTAyMDM8L1JldHJvYWN0aXZlRGF0ZT4KICAgICAgICA8RWZmRGF0ZT4yMDA1MDUwMTwv
RWZmRGF0
ZT4KICAgICAgICA8RXhwRGF0ZT4yMDA2MDUwMTwvRXhwRGF0ZT4KICAgICAgICA8Q3Jl
YXRldGlt
ZT4wNi8zMC8yMDA5IDEyOjAxOjAzPC9DcmVhdGV0aW1lPgogICAgICAgIDxNb2RpZnl0
aW1lPjA3
LzAyLzIwMDkgMTI6NTU6MDk8L01vZGlmeXRpbWU+CiAgICAgICAgPElzc3VlU3RhdGVD
b2RlPkdB
```

PC9Jc3N1ZVN0YXRlQ29kZT4KICAgICAgICA8V2lwUmVhc29uPjwvV2lwUmVhc29uPgoJ
CTxVc2Vy
R3JvdXA+MTwvVXNlckdyb3VwPgoJCTxEZXNjcmlwdGlvbj5XZWxjb21lIFBhY2tldDwv
RGVzY3Jp
CiAgICAgICAgPC9BZGRyZXNzZWVEYXRhPgogICAgPC9Eb2N1bWVudFJlcXVlc3Q+Cg==
</trns:Binary>
</trns:Content>
</trns:Data>
</Transaction>
</Payload>
</Job>

CONFIGURING THE IDENTIFIER

The Identifier process reads a job and breaks it into one or more transactions. It is deployed and managed by the Supervisor process and it monitors an input queue and waits for notification messages from the Scheduler process that transactions are ready for processing.

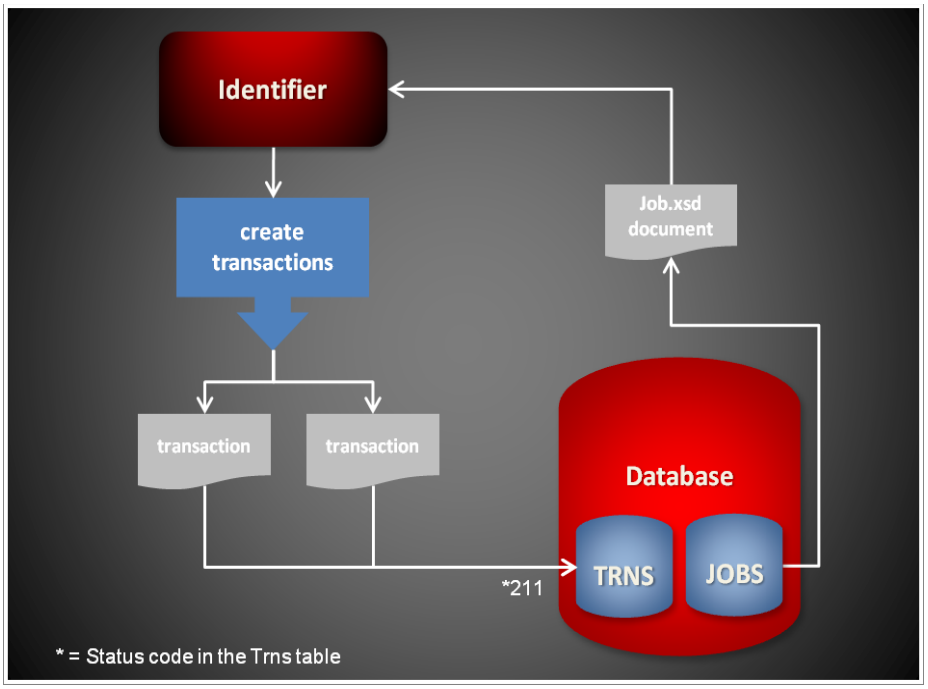
Once a notification message is received, the Identifier retrieves a record from the Jobs table and breaks it into one or more transactions as TRNS table records. The Identifier process typically runs after the Receiver process and reads input from Jobs records generated by the Receiver.



Each notification message received by the Identifier provides the job ID for a job in the Jobs table that needs identifying. Here is an example of a message:

```
<?xml version="1.0" encoding="UTF-8"?>
<JobTicket
    xmlns="oracle/documaker/schema/tables/jobs"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<JOB_ID>101</JOB_ID>
</JobTicket>
```

The Identifier reads an input record from the Jobs table. It then breaks apart the job record into one or more transactions as new records in the TRNS table. The Identifier then updates the status code for the Jobs record and the new TRNS records so the Scheduler process can notify the next process in the assembly line.



STARTING AND STOPPING THE IDENTIFIER

To	Then
Verify the Identifier is running.	Verify there is a running process with the name docfactory_identifier.
Start the Identifier	Place the identifier.jar file in the deploy directory of Document Factory.
Stop the Identifier	Remove the identifier.jar file from the deploy directory of Document Factory.

Note The identifier.jar configuration file is uncompressed and deployed to the temp\identifier directory. This directory becomes the working directory for the Identifier. All output, including Log4J output, uses this directory as the starting directory.

USING IDENTIFIER CONFIGURATION RESOURCES

The configuration information for the Identifier is stored in these resources:

Resource	Contains the
identifier.jar file	Minimal startup configuration information.
.bindings file	Java Naming and Directory Interface (JNDI) data sources.
APPCONFIGCONTEXT table	Configuration options.
ALCONFIGCONTEXT table	Configuration options for the Identifier status codes and message bus.

identifier.jar File

The identifier.jar file is located in the \deploy subdirectory of the Document Factory. It contains these configuration resources:

Component	Description
deploy.properties	Contains the minimal startup configuration information.
log4j.xml	Used to capture Log4J diagnostic and error output during start up. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j/
log4j.dtd	Used by log4j.xml file.

deploy.properties File

The deploy.properties file is extracted and placed in the temp\identifier working directory. This file contains the minimal startup configuration options used to read the configuration for the Identifier from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables:

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Identifier configuration.
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Identifier configuration.
application.id	The value of APP_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Identifier configuration.
config	The configuration name for the Identifier. The default is Identifier. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Identifier configuration.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.

Option	Description
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=4
config=Identifier
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

log4j.xml File

The log4j.xml file is extracted and placed in the temp/identifier working directory. The log4j.xml file contains loggers used during start up of the Identifier, prior to the Identifier loading the Log4J configuration from the APPCONFIGCONTEXT table. See the Log4J configuration options in the *APPCONFIGCONTEXT Table* on page 151 for more information.

.bindings File

The .bindings file is located in the config\context subdirectory of the Document Factory. It contains the Java Naming and Directory Interface (JNDI) data sources used by the Identifier. Each JNDI data source contains these configuration options:

Option	Description
ClassName	The data source fully-qualified class name. Use the javax.sql.DataSource value.
FactoryName	The data source factory fully-qualified class name. Use the org.apache.commons.dbcp.BasicDataSourceFactory value. The BasicDataSourceFactory class supports connection pooling.
driverClassName	The Java Database Connectivity (JDBC) driver class name.
url	The JDBC URL.
maxOpenPreparedStatements	The maximum number of prepared statements to cache in the connection pool. Use the value -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	How often the idle object evictor thread should run and perform clean up of the stale connection handles. Use the value -1 to disable the idle object evictor thread.
validationQuery	A validation query that should be run when borrowing objects from the connection pool.

Option	Description
username	The JDBC user name.
password	The JDBC password.
testOnBorrow	Set this option to Yes if validationQuery should be used when borrowing an object from the connection pool. The default is No.
initialSize	The initial connection pool size.
maxActive	The maximum number of active connections in the pool.
maxIdle	The maximum number of idle connections in the pool.
minIdle	The minimum number of idle connections in the pool.
maxWait	The maximum time (in milliseconds) to wait for a connection object to be retrieved from the pool before issuing an error.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type= maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
```

```
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
DMKRConfig/RefAddr/9/Type=maxWait
#Unix friendly Documaker Doc. Factory JNDI DataSource
DMKRFactory/ClassName=javax.sql.DataSource
DMKRFactory/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRFactory/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRFactory/RefAddr/0/Encoding=String
DMKRFactory/RefAddr/0/Type=driverClassName
DMKRFactory/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRFactory/RefAddr/1/Encoding=String
DMKRFactory/RefAddr/1/Type=url
DMKRFactory/RefAddr/10/Content=-1
DMKRFactory/RefAddr/10/Encoding=String
DMKRFactory/RefAddr/10/Type=maxOpenPreparedStatements
DMKRFactory/RefAddr/11/Content=-1
DMKRFactory/RefAddr/11/Encoding=String
DMKRFactory/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRFactory/RefAddr/12/Content=select 1 from dual
DMKRFactory/RefAddr/12/Encoding=String
DMKRFactory/RefAddr/12/Type=validationQuery
DMKRFactory/RefAddr/2/Content=dmkr_asline
DMKRFactory/RefAddr/2/Encoding=String
DMKRFactory/RefAddr/2/Type=username
DMKRFactory/RefAddr/3/Content=oracle12
DMKRFactory/RefAddr/3/Encoding=String
DMKRFactory/RefAddr/3/Type=password
DMKRFactory/RefAddr/4/Content=true
DMKRFactory/RefAddr/4/Encoding=String
DMKRFactory/RefAddr/4/Type=testOnBorrow
DMKRFactory/RefAddr/5/Content=1
DMKRFactory/RefAddr/5/Encoding=String
DMKRFactory/RefAddr/5/Type=initialSize
DMKRFactory/RefAddr/6/Content=8
DMKRFactory/RefAddr/6/Encoding=String
DMKRFactory/RefAddr/6/Type= maxActive
DMKRFactory/RefAddr/7/Content=8
DMKRFactory/RefAddr/7/Encoding=String
DMKRFactory/RefAddr/7/Type=maxIdle
DMKRFactory/RefAddr/8/Content=0
DMKRFactory/RefAddr/8/Encoding=String
DMKRFactory/RefAddr/8/Type=minIdle
DMKRFactory/RefAddr/9/Content=60000
DMKRFactory/RefAddr/9/Encoding=String
DMKRFactory/RefAddr/9/Type=maxWait
```

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Identifier*:

Option	Description
StartCommand	Defines the command used to start the Identifier. This is used by the Supervisor to start the class specified in JavaClass configuration option. The default is Java.
StartArguments	Defines the initialization arguments used to start the Identifier.

Option	Description
JavaClass	The Java class that is used to start the worker class specified in WorkerClass configuration option. Use the oracle.documaker.process.ProcessShell value. ProcessShell class is a process shell that provides all functionality needed to communicate with the Supervisor process and to start and manage the worker class specified in WorkerClass configuration option.
JVMOptions	Any JVM options the Supervisor process uses to start JavaClass. There is no default.
Instances	(Optional) The number of instances the Supervisor should start for a process configuration. The default is two (2).
UseLoadBalancing	(Optional) This option controls wether the Supervisor checks the idle time of a processes' instances that are running and starts additional ones when all of them are busy. Instances are considered busy when their idle time is less than the value provided in the MinIdleTimeSeconds option. The Supervisor uses the value provided in the IdleTimeChecks option to determine the number of idle time checks to run before it starts additional instances. When additional instances are started for load balancing purposes, they are shut down by the Supervisor if their idle time exceeds the value in the MaxIdleTimeSeconds option. The maximum number of instances running is the value for the MaxInstances option (including the instances configured in the Instances option). The Supervisor checks the idle time of the current instances at the interval specified in the IdleTimeCheckIntervalSeconds and if all are busy, it starts an additional number of instances equal to the value provided in the IncrementCount option. Please note that the Supervisor does not start checking the busy time of the current instances until the time provided in the IdleTimeCheckDelaySeconds option elapses. Make sure the value for the delay is ample enough to provide for all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. You can enter Yes or No. The default is Yes.
MaxInstances	(Optional) This option controls the maximum number of instances that can run when the UseLoadBalancing option is enabled. The default is the number of processors times two.
IncrementCount	(Optional) This option controls how many additional instances are started during the current check when all instances running are busy and the UseLoadBalancing option is enabled. The default is two (2).
IdleTimeCheckIntervalSeconds	(Optional) This option controls how often the Supervisor checks the idle time of the instances that are running to determine if they are busy so it can start additional ones when the UseLoadBalancing option is enabled. The default is 10 seconds.
IdleTimeCheckDelaySeconds	(Optional) This option controls the initial delay before the first idle time check is performed by the Supervisor when the UseLoadBalancing option is enabled. This time should be ample enough to allow all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. The default is 120 seconds.
IdleTimeChecks	(Optional) This option defines the number of consecutive idle time checks that must fail, meaning all instances were busy during each check, before more instances are started when the UseLoadBalancing option is enabled. Each check takes place at the IdleTimeCheckIntervalSeconds interval. The default is 12.

Option	Description
MinIdleTimeSeconds	(Optional) This option controls the minimum idle time for each instance. The idle time represents how long it has been since an instance processed the last request. If the Supervisor detects an instance has an idle time less than the value provided for this option, it considers it busy for the purpose of load balancing. The default is 5 seconds.
MaxIdleTimeSeconds	(Optional) This option controls the maximum idle time for an additional instance. The idle time represents how long it has been since an instance performed processing. If the Supervisor detects an instance, which was started for the purpose of load balancing, has reached an idle time greater than the value provided for this option, it sends the instance a shutdown request. The default is 120 seconds.
MaxTransactions	(Optional) This option controls the maximum number of transactions an instance can process before it is restarted by the Supervisor. The default is -1, which disables this option.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before it is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
OrderedRestartIntervalSeconds	(Optional) This option controls the interval used for restarting each process instance in a sequential/ordered manner when the MaxTransactions or MaxUpTime options are used. The Supervisor restarts one instance at a time and waits for an amount of time equal to the value specified for this option before it restarts the next one and so on until it has restarted all of them. If you set this option to less than 60 seconds, you can negatively affect performance. The default is 60 seconds.
WatchList	A comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted.
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is 5.
WorkerClass	The class that extends the oracle.documaker.process.worker.Worker Thread class and is started by the class specified in JavaClass configuration option. Use the oracle.documaker.identifier.Identifier value.
WorkerThreads	How many threads of WorkerClass should be created by JavaClass. You can use the value 1. The default is one (1).
WorkerIntervalMillis	How often, in milliseconds, each WorkerClass thread should perform its work. The default is 5000.
WorkerStartDelayMillis	How long, in milliseconds, each WorkerClass thread should wait after startup and before performing any work. The default is 10000.
ShutdownHookClass	The class that extends the oracle.documaker.process.shutdown.ShutdownHook class. Use the oracle.documaker.identifier.shutdown.IdentifierShutdownHook value.

Option	Description
IPCIntervalMillis	How often, in milliseconds, the inter-process communication (IPC) thread should perform its work. This option is used by JavaClass to report back to the Supervisor process. The default is 1000.
IPCStartDelayMillis	How long, in milliseconds, the inter-process communication (IPC) thread should wait after startup and before performing any work. This option is used by JavaClass to report back to the Supervisor process. The default is 10000.
Log4jIntervalMillis	How often, in milliseconds, the Log4J resource monitor thread should perform its work. This option is used to monitor log4j.xml file deployed under temp\identifier working directory and reload it when a change is detected. The default is 1000.
Log4jStartDelayMillis	How long, in milliseconds, the Log4J resource monitor thread should wait after startup and before performing any work. This option is used to monitor log4j.xml file deployed under temp\identifier working directory and reload it when a change is detected. The default is 10000.

Here is an example:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/jre/bin/docfactory_identifier
JavaClass	oracle.documaker.process.ProcessShell
JVMOptions	-Xmx128m -Duser.name=oracle
Instances	1
UseLoadBalancing	No
WorkerClass	oracle.documaker.identifier.Identifier
WorkerThreads	4
WorkerIntervalMillis	1000
WorkerStartDelayMillis	5000
ShutdownHookClass	oracle.documaker.identifier.shutdown.IdentifierShutdownHook
IPCIntervalMillis	1000
IPCStartDelayMillis	10000
Log4jIntervalMillis	5000
Log4jStartDelayMillis	10000

Log4J configuration options

For specific information on the Log4J configuration options, see *Defining Log4J Configuration Options* on page 277.

ALCONFIGCONTEXT Table

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Status*:

Option	Description
Identifier-Ready	This is the status code that indicates a transaction is ready to be sent to the Identifier. The default is 111.
Identifier-ACK	This is the status code that indicates a transaction has been received and it is being processed by Identifier. The default is 131.
Identifier-Error	This is the status code that indicates the Identifier process failed to process a transaction. The default is 141.
Assembler-Ready	This is the status code that indicates the Identifier process successfully processed a transaction and it is now ready for the Assembler process. The default is 211.

Here is an example:

Option	Value
Identifier-Ready	111
Identifier-ACK	131
Identifier-Error	141
Assembler-Ready	211

These options are read from this table when the GROUP_NAME column value is *Bus*:

Option	Description
IdentifierQueue	The name of the queue the Identifier uses to receive notifications from the Scheduler process.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

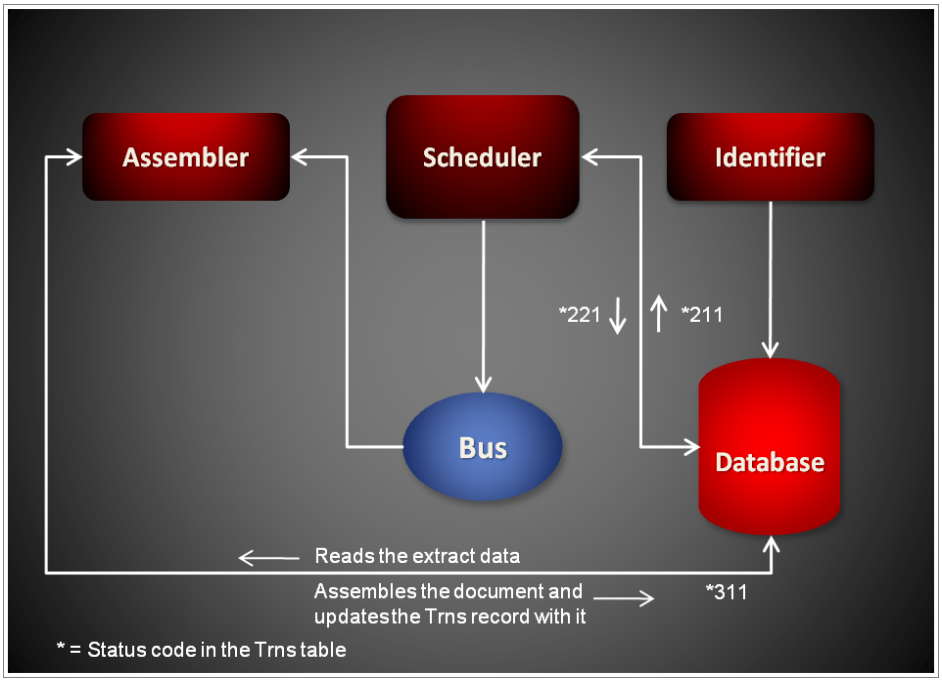
Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001

Option	Value
jms.qcf.name	jms/qcf
IdentifierQueue	jms/identifier_requestq
TimeoutSeconds	5

CONFIGURING THE ASSEMBLER

The Assembler process reads the extract data for a transaction and assembles a document from it. It is deployed and managed by the Supervisor process and it monitors an input queue and waits for notification messages from the Scheduler process.

Once a notification message is received, the Assembler retrieves the extract data for a transaction from a record in the TRNS table and assembles the document. The Assembler process typically runs after the Identifier process and reads input from TRNS records generated by the Identifier process.

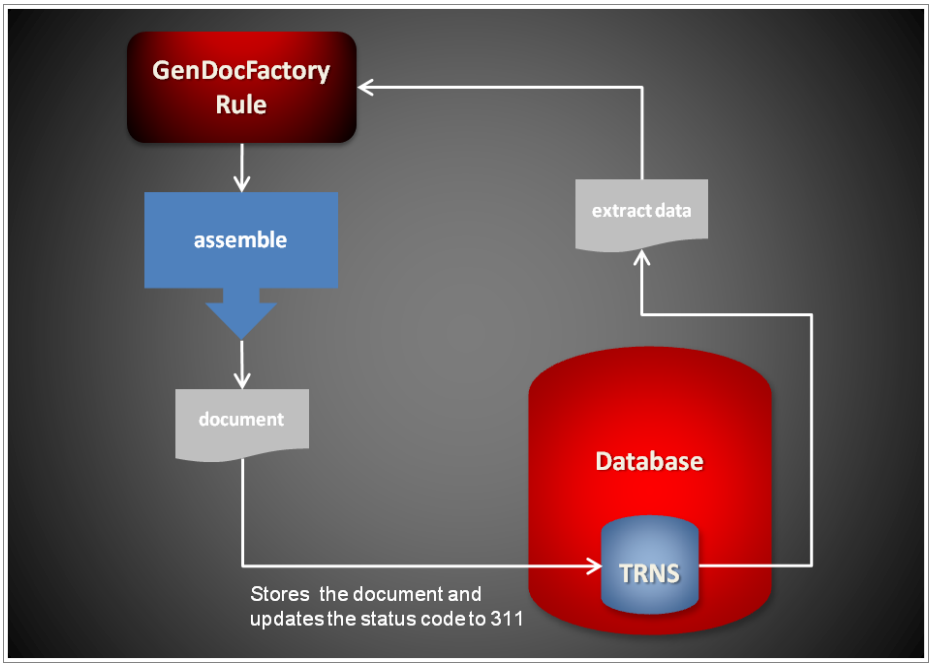


Each notification message received by the Assembler provides the transaction ID for a transaction in TRNS table that needs assembling. Here is an example of a message:

```
<?xml version="1.0" encoding="UTF-8"?>
<TransactionTicket
    xmlns="oracle/documaker/schema/tables/trns"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<TRN_ID>101</TRN_ID>
</TransactionTicket>
```

USING THE GENDOCFACTORY RULE

The Assembler uses the GenDocFactory rule to perform basic transaction processing and housekeeping. The rule loads the extract data from a TRNS record, assembles the document, unloads the NA\POL file data back to the TRNS record, and updates the status code for it at the completion of the assembly process so the Scheduler process can notify the next process in the assembly line.



Here is an overview of what the GenDocFactory rule does:

Initialization	Loads the transaction status.
PreProc	• Reads the transaction record from the TRNS table by the TRN_ID. • Validates the transaction status is set to *Assembler Start*. • Updates the transaction status to *Assembler Processing*. • Sets the NA\POL unload option (NAPOLTYPE) to indicate the format of the input data. Acceptable values are: 0=XML, 1=BLOB, or a URI such as file://c/docfactory/tempdata/. • Loads the extract data from the transaction record.
PostProc	If the form set has zero forms, the rule: • Sets the transaction status to an error state. • Issues an error message. Otherwise, the rule: • Unloads the form set data to the transaction table record. • Updates the transaction status to indicate *Assembler End*.

Starting and Stopping the Assembler

To	Then
Verify the Assembler is running.	Verify there is a running process with the name docfactory_assembler.
Start the Assembler	Place the assembler.jar file in the deploy directory of Document Factory.
Stop the Assembler	Remove the assembler.jar file from the deploy directory of Document Factory.

Note The assembler.jar configuration file is uncompressed and deployed to the temp\assembler directory. This directory becomes the working directory for the Assembler. All output, including Log4J output, uses this directory as the starting directory.

USING ASSEMBLER CONFIGURATION RESOURCES

The configuration information for the Assembler is stored in these resources:

Resource	Contains the
assembler.jar file	Minimal startup configuration information.
.bindings file	Java Naming and Directory Interface (JNDI) data sources.
APPCONFIGCONTEXT table	Configuration options.
ALCONFIGCONTEXT table	Configuration options for the Assembler status codes and message bus.
fsiuser_1.ini file	INI options specific to the Assembler process.
fsisys.ini file	INI options common to the Assembler, Distributor and Presenter processes.
afgjob_1.jdt file	Documaker rules run by the Assembler process.

assembler.jar File

The assembler.jar file is located in the \deploy subdirectory of the Document Factory. It contains these configuration resources:

Component	Description
deploy.properties	Contains the minimal startup configuration information.
log4j.xml	Used to capture Log4J diagnostic and error output during start up. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j/
log4j.dtd	Used by the log4j.xml file.

deploy.properties File

The deploy.properties file is extracted and placed in the temp\assembler working directory. This file contains the minimal startup configuration options used to read the configuration for the Assembler from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables:

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Assembler configuration.
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Assembler configuration.

Option	Description
application.id	The value of APP_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Assembler configuration.
config	The configuration name for the Assembler. The default is Assembler. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Assembler configuration.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=5
config=Assembler
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

log4j.xml File

The log4j.xml file is extracted and placed in the temp/assembler working directory. The log4j.xml file contains loggers used during start up of the Assembler, prior to the Assembler loading the Log4J configuration from the APPCONFIGCONTEXT table. See the Log4J configuration options in the *APPCONFIGCONTEXT Table* on page 163 for more information.

.bindings File

The .bindings file is located in the config\context subdirectory of the Document Factory. It contains the Java Naming and Directory Interface (JNDI) data sources used by the Assembler. Each JNDI data source contains these configuration options:

Option	Description
ClassName	The fully-qualified class name for the data source. Use the javax.sql.DataSource value.
FactoryName	The fully-qualified class name for the data source factory. Use the org.apache.commons.dbcp.BasicDataSourceFactory value. The BasicDataSourceFactory class supports connection pooling.

Option	Description
driverClassName	The Java Database Connectivity (JDBC) driver class name.
url	The JDBC URL.
maxOpenPreparedStatements	The maximum number of prepared statements to cache in the connection pool. Use the value -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	How often the idle object evictor thread should run and perform a clean up of the stale connection handles. Use the value -1 to disable the idle object evictor thread.
validationQuery	A validation query that should be run when borrowing objects from the connection pool.
username	The JDBC user name.
password	The JDBC password.
testOnBorrow	Set this option to Yes if validationQuery should be used when borrowing an object from the connection pool. The default is No.
initialSize	The initial connection pool size.
maxActive	The maximum number of active connections in the pool.
maxIdle	The maximum number of idle connections in the pool.
minIdle	The minimum number of idle connections in the pool.
maxWait	The maximum time (in milliseconds) to wait for a connection object to be retrieved from the pool before issuing an error.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
```

```
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type= maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
DMKRConfig/RefAddr/9/Type=maxWait
#Unix friendly Documaker Doc. Factory JNDI DataSource
DMKRFactory/ClassName=javax.sql.DataSource
DMKRFactory/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRFactory/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRFactory/RefAddr/0/Encoding=String
DMKRFactory/RefAddr/0/Type=driverClassName
DMKRFactory/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRFactory/RefAddr/1/Encoding=String
DMKRFactory/RefAddr/1/Type=url
DMKRFactory/RefAddr/10/Content=-1
DMKRFactory/RefAddr/10/Encoding=String
DMKRFactory/RefAddr/10/Type=maxOpenPreparedStatements
DMKRFactory/RefAddr/11/Content=-1
DMKRFactory/RefAddr/11/Encoding=String
DMKRFactory/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRFactory/RefAddr/12/Content=select 1 from dual
DMKRFactory/RefAddr/12/Encoding=String
DMKRFactory/RefAddr/12/Type=validationQuery
DMKRFactory/RefAddr/2/Content=dmkr_asline
DMKRFactory/RefAddr/2/Encoding=String
DMKRFactory/RefAddr/2/Type=username
DMKRFactory/RefAddr/3/Content=oracle12
DMKRFactory/RefAddr/3/Encoding=String
DMKRFactory/RefAddr/3/Type=password
DMKRFactory/RefAddr/4/Content=true
DMKRFactory/RefAddr/4/Encoding=String
DMKRFactory/RefAddr/4/Type=testOnBorrow
DMKRFactory/RefAddr/5/Content=1
DMKRFactory/RefAddr/5/Encoding=String
DMKRFactory/RefAddr/5/Type=initialSize
DMKRFactory/RefAddr/6/Content=8
DMKRFactory/RefAddr/6/Encoding=String
DMKRFactory/RefAddr/6/Type= maxActive
DMKRFactory/RefAddr/7/Content=8
DMKRFactory/RefAddr/7/Encoding=String
DMKRFactory/RefAddr/7/Type=maxIdle
DMKRFactory/RefAddr/8/Content=0
DMKRFactory/RefAddr/8/Encoding=String
DMKRFactory/RefAddr/8/Type=minIdle
DMKRFactory/RefAddr/9/Content=60000
DMKRFactory/RefAddr/9/Encoding=String
DMKRFactory/RefAddr/9/Type=maxWait
```

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Assembler*:

Option	Description
StartCommand	The process name to start. It should be the full path and executable name.
StartArguments	These should be the arguments the StartCommand executable expects.
env.mode.*	The environment variables the process expects to run. The Supervisor creates an environment variable for each env.mode.* configuration option it encounters. The naming convention is `env.mode.name` Where *mode* can be either zero (0), meaning prepend, one (1), meaning append, or two (2), meaning overwrite, and *name* is the name of the environment variable. When the mode is not defined, the default is two (2). An example of an env.mode.* variable would be env.0.PATH or env.ORACLE_HOME. Notice the second example uses the default overwrite mode.
StartDirectory	This should be the start directory.
Instances	(Optional) The number of instances the Supervisor should start for a process configuration. The default is two (2).
UseLoadBalancing	(Optional) This option controls wether the Supervisor checks the idle time of a process's instances that are running and starts additional ones when all of them are busy. Instances are considered busy when their idle time is less than the value provided in the MinIdleTimeSeconds option. The Supervisor uses the value provided in the IdleTimeChecks option to determine the number of idle time checks to run before it starts additional instances. When additional instances are started for load balancing purposes, they are shut down by the Supervisor if their idle time exceeds the value in the MaxIdleTimeSeconds option. The maximum number of instances running is the value for the MaxInstances option (including the instances configured in the Instances option). The Supervisor checks the idle time of the current instances at the interval specified in the IdleTimeCheckIntervalSeconds and if all are busy, it starts an additional number of instances equal to the value provided in the IncrementCount option. Please note that the Supervisor does not start checking the busy time of the current instances until the time provided in the IdleTimeCheckDelaySeconds option elapses. Make sure the value for the delay is ample enough to provide for all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. You can enter Yes or No. The default is Yes.
MaxInstances	(Optional) This option controls the maximum number of instances that can run when the UseLoadBalancing option is enabled. The default is the number of processors times two.
IncrementCount	(Optional) This option controls how many additional instances are started during the current check when all instances running are busy and the UseLoadBalancing option is enabled. The default is two (2).

Option	Description
IdleTimeCheckIntervalSeconds	(Optional) This option controls how often the Supervisor checks the idle time of the instances that are running to determine if they are busy so it can start additional ones when the UseLoadBalancing option is enabled. The default is 10 seconds.
IdleTimeCheckDelaySeconds	(Optional) This option controls the initial delay before the first idle time check is performed by the Supervisor when the UseLoadBalancing option is enabled. This time should be ample enough to allow all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. The default is 120 seconds.
IdleTimeChecks	(Optional) This option defines the number of consecutive idle time checks that must fail, meaning all instances were busy during each check, before more instances are started when the UseLoadBalancing option is enabled. Each check takes place at the IdleTimeCheckIntervalSeconds interval. The default is 12.
MinIdleTimeSeconds	(Optional) This option controls the minimum idle time for each instance. The idle time represents how long it has been since an instance processed the last request. If the Supervisor detects an instance has an idle time less than the value provided for this option, it considers it busy for the purpose of load balancing. The default is 5 seconds.
MaxIdleTimeSeconds	(Optional) This option controls the maximum idle time for an additional instance. The idle time represents how long it has been since an instance performed processing. If the Supervisor detects an instance, which was started for the purpose of load balancing, has reached an idle time greater than the value provided for this option, it sends the instance a shutdown request. The default is 120 seconds.
MaxTransactions	(Optional) This option controls the maximum number of transactions an instance can process before it is restarted by the Supervisor. The default is -1, which disables this option.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before it is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
OrderedRestartIntervalSeconds	(Optional) This option controls the interval used for restarting each process instance in a sequential/ordered manner when the MaxTransactions or MaxUpTime options are used. The Supervisor restarts one instance at a time and waits for an amount of time equal to the value specified for this option before it restarts the next one and so on until it has restarted all of them. If you set this option to less than 60 seconds, you can negatively affect performance. The default is 60 seconds.
WatchList	A comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted.
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is 5.

Here is an example:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/bin/docfactory_assembler
StartArguments	/ini=fsiuser_1.ini /debug=0 /phase=1
env.0.PATH	/oracle_home/*InstallationLocation*/oracle_instantclient_11_2,/oracle_home/*InstallationLocation*/jre/bin,/oracle_home/*InstallationLocation*/jre/bin/client,/oracle_home/*InstallationLocation*/bin
env.ORACLE_HOME	/oracle_home/*InstallationLocation*/bin
env.NLS_LANG	AMERICAN_AMERICA.AL32UTF8
env.TNS_ADMIN	/oracle_home/*InstallationLocation*/oracle_instantclient_11_2/NETWORK/ADMIN
env.JVM_OPTIONS	-Xmx256m,-Duser.name=oracle,-Dlog4j.configuration=/oracle_home/*InstallationLocation*/docfactory/temp/assembler/log4j.xml,-Dlog4j.file=/oracle_home/*InstallationLocation*/docfactory/temp/assembler/logs/log4j.log,-Djndi.context=/oracle_home/*InstallationLocation*/docfactory/config/context,-Dfactory.jndi.name=DMKRFactory,-Dconfig.jndi.name=DMKRConfig,-Dschema=DMKR_ASLINE
StartDirectory	/oracle_home/*InstallationLocation*/dmres/correspondence
Instances	2
UseLoadBalancing	No
MaxInstances	8
IncrementCount	1
IdleTimeCheckIntervalSeconds	15
IdleTimeCheckDelaySeconds	240
IdleTimeChecks	5
MinIdleTimeSeconds	5
MaxIdleTimeSeconds	120
MaxTransactions	-1
MaxReportIntervalSeconds	180
MaxUpTimeSeconds	-1
WaitForShutdownSeconds	60
OrderedRestartIntervalSeconds	60
WatchList	/oracle_home/*InstallationLocation*/mstrres/correspondence/fsiuser_1.ini,/oracle_home/*InstallationLocation*/mstrres/correspondence/fsisys.ini

InstallationLocation = The installation location where you installed Document Factory.

Option	Value
MaxRestarts	5

InstallationLocation = The installation location where you installed Document Factory.

Log4J configuration options

For specific information on the Log4J configuration options, see *Defining Log4J Configuration Options* on page 277.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Assembler-Ready	This is the status code that indicates a transaction is ready to be sent to the Assembler. The default is 211.
Assembler-ACK	This is the status code that indicates a transaction has been received and it is being processed by Assembler. The default is 231.
Assembler-Error	This is the status code that indicates the Assembler process failed to process a transaction. The default is 241.

Here is an example:

Option	Value
Assembler-Ready	211
Assembler-ACK	231
Assembler-Error	241

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
AssemblerQueue	The name of the queue the Assembler uses to receive notifications from the Scheduler process.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf
AssemblerQueue	jms/assembler_requestq
TimeoutSeconds	5

FSIUSER_1.INI File

This file provides the INI options required to run the Assembler process under the Document Factory. You can find this file in the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section.

Database Handler Definition

These options are read from the DBHandler:ODBC_DMKR_ASLINE INI control group:

Option	Description
Class	The class name for the ODBC handler
Server	The ODBC DSN name.
UserID	The user ID for the ODBC DSN.
Password	The password for the ODBC DSN.
Debug	Set this option to Yes if diagnostic information should be generated for ODBC operations. You can enter Yes or No.

Here is an example:

```
< DBHandler:ODBC_DMKR_ASLINE >
   Class  = ODBC
   Passwd = ********
   Server = dmkr_asline
   UserID = dmkr_asline
   Debug  = No
```

WIP Index Table Definition

These options are read from the DBTable:WIP INI control group and controls the database handler the Assembler uses for communication with the TRNS table:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:WIP >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = FORMSETID
```

WIP Data Table Definition

These options are read from the DBTable:WIPData INI control group. They specify the database table location of the form set data where the Assembler should get the TRNS content:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:WIPData >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = FORMSETID
```

Extract Table Definition

These options are read from the DBTable:EXTR INI control group. These options specify the location of the extract data used as input during the Assembler process:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:EXTR >
     DBHandler = ODBC_DMKR_ASLINE
     UniqueTag = TRN_ID
```

Transaction Status Table Definition

These options are read from the DBTable:TRNSTATUS INI control group. These options identify the location of the database table that should be updated when the Assembler process is running and has completed:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:TRNSTATUS >
     DBHandler = ODBC_DMKR_ASLINE
     UniqueTag = TRN_ID
```

DFD Definitions

These options are read from the WIPData INI control group:

Option	Description
DatabaseWIP	Set this option to Yes if you want to store WIP in a database. The default is No.
File	The internal name of the WIP table.
WIPDFDFile	The name of the WIP index DFD file.

WIPDataDFD	The name of the WIP data DFD file for XML NA/POL. This is used by default or when you have the following INI setting:
	`< DocFactory >`
	`NAPOLTYPE =`
	This is the default for the NAPOLTYPE setting and it tells you the NA/POL information is stored in XML format.

WIPDsDataDFD	(Optional) The name of the WIP data DFD file for combined NA/POL. This is only used when you have the following INI setting:
	`< DocFactory >`
	`NAPOLTYPE = 1`
	This setting uses the native NA/POL format for storing form set data. With this option enabled, you cannot use standard xPath syntax to query NA/POL data.

TRNExtrDFD	The name of the extract DFD file. The default is TRNS.DFD — or same name as the WIPDFDFile.
TRNStatusDFD	(Optional) The name of the transaction status DFD file.
Jobs	(Optional) The name of the Jobs table.
JobsDFD	The name of the Jobs index DFD file.

Here is an example:

```
< WIPData >
   DatabaseWIP= Yes
   File      = WIP
   Path      = <CONFIG:CORRESPONDENCE> WIPPath =
   WIPData   = WIPData
   Jobs      = JOBS
   JobsDFD   =
c:\oracle_home\InstallationLocation\mstrres\dmres\deflib\jobs.dfd
   BCHS      = BCHS
   BCHSDFD   =
c:\oracle_home\InstallationLocation\mstrres\dmres\deflib\BCHS.dfd
   WIPDFDFile=
c:\oracle_home\InstallationLocation\mstrres\dmres\deflib\trnsdf.dfd
   WIPDataDFD=
c:\oracle_home\InstallationLocation\mstrres\dmres\deflib\docdata.dfd
   WIPDsDataDFD=
c:\oracle_home\InstallationLocation\mstrres\dmres\deflib\dsdata.dfd
   DocFactory= Yes
   WriteFiles= No
```

ODBC File Conversion

These options are read from the ODBC_FileConvert INI control group:

Option	Description
WIP	The actual name of the database table that corresponds to the internal WIP table name. This is the WIP index table.
WIPData	The actual name of the database table that corresponds to the internal WIPData table name. This is the WIP data table.
EXTR	The actual name of the database table that corresponds to the internal EXTR table name. This is the extract data table.

Here is an example:

```
< ODBC_FileConvert >
    WIP       = TRNS
    WIPData   = TRNS
    EXTR      = TRNS
```

Configuring Document Factory Options

These options are read from the DocFactory INI control group:

Option	Description
Assembler_StatusCode	This is the status code the Assembler sets when it finishes processing if the transaction did not have errors and was not marked for interactive editing (WIP). The default is B.
Assembler_ApprovalState	This is the approval state the Assembler sets when it finishes processing if the transaction did not have errors and was not marked for interactive editing (WIP). The default is 10.
ASMManual_StatusCode	This is the status code the Assembler sets for transactions flagged as manual. The default is W.
ASMManual_ApprovalState	This is the approval state the Assembler sets for transactions flagged as manual. The default is 30.
ASMError_StatusCode	This is the status code the Assembler sets for transactions flagged with errors. The default is W.
ASMError_ApprovalState	This is the approval state the Assembler sets for transactions flagged with errors. The default is 40.
Assembler_Start	The Assembler start status code. The default is 221.
Assembler_Processing	The Assembler ACK status code. The default is 231.
Assembler_Error	The Assembler error status code. The default is 241.
Assembler_End	The Assembler end status code. The default is 311.
NAPOLType	Indicates the type of transaction data. The Assembler accepts the following: • 0 = XML • 1 = BLOB (also known as native) • A URI such as *file://c/docfactory/tempdata/* The default is zero (0).

Option	Description
Bindings	The path location for the Java Naming and Directory Interface (JNDI) .bindings file containing the data source information for JNI code. The default is /docfactory/config/context/.
UpdateBCHS_RCPS	Set this option to No if you do not want the system to write the PUBS_ID to the BCHS_RCPS record during Presenter processing. The default is Yes.

Here is an example:

```
< DocFactory >
   Assembler_StatusCode  = B
   Assembler_ApprovalState = 10
   ASMManual_StatusCode = W
   ASMManual_ApprovalState = 30
   ASMError_StatusCode = W
   ASMError_ApprovalState = 40
   Assembler_Start = 221
   Assembler_Processing = 231
   Assembler_Error = 241
   Assembler_End = 311
   NAPOLType = 0
   Bindings = /oracle_home/InstallationLocation/docfactory/config/
context
   UpdateBCHS_RCPS = Yes
```

Logging Messages to the Database

These options are read from the Environment INI control group:

Option	Description
JLOG_Enabled	Set this option to Yes to redirect warning and error messages to the LOGS and ERRS tables instead of being written to the trace file. The default is No.

Here is an example:

```
< Environment >
    JLOG_Enabled = Yes
```

Controlling Log Output

These options are read from the DocFactory_Assembler:JLog INI control group:

Option	Description
LogLogger	The name of the Log4J logger used to log warning messages to the LOGS table. This name should match the Log4J logger name in log4j.xml file.
ErrorLogger	The name of the Log4J logger used to log error messages to the ERRS table. This name should match the Log4J logger name in log4j.xml file.
ColumnNames	A comma-delimited list of table column names to GVM mappings. Is used by the loggers to capture the GVM values and set them as the column values. The format for each comma-delimited token can be ColumnName=GVMName or just ColumnName.
BufferSize	The maximum buffer size for messages. This value should match the length of the LOGMESSAGE and ERRMESSAGE columns.

Option	Description
Debug	Set this option to Yes if you want diagnostic output generated for the Logger. The default is No.
LogError	Set this option to No if you want the system to suppress all error messages. The default is Yes, which tells the system to issue error messages.
LogWarning	Set this option to Yes if you want the system to issue warning messages. The default is No, which suppresses all warning messages.

Here is an example:

```
< DocFactory_Assembler:JLog >
    LogLogger = LogLogger
    ErrorLogger = ErrorLogger
    BufferSize = 2000
    Debug = No
    LogError = Yes
    LogWarning = No
    ColumnNames = JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,
BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_ID
```

FSISYS.INI File

This file can be found in the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section. It provides INI options required to run the Assembler process under the Document Factory.

Enabling Document Factory

These options are read from the RunMode INI control group:

Option	Description
DocFactory	Must be set to Yes if you are using Document Factory. To facilitate legacy Documaker Server processing, this option defaults to No.

Here is an example:

```
< RunMode >
    DocFactory = Yes
```

Enabling Debug options

These options are read from the Debug_Switches INI control group. All of these options default to No, to reduce processing overhead.

Option	Description
Show_Debug_Options	Set this option to Yes to show all debug options. The default is No.
Enable_Debug_Options	Set this option to Yes to enable all debug options. The default is No.
DBLib	Set this option to Yes to generate diagnostic information for the DBLIB library. The default is No.
WIPLib	Set this option to Yes to generate diagnostic information for the WIPLIB library. The default is No.

Option	Description
ARCLib	Set this option to Yes to generate diagnostic information for the ARCLIB library. The default is No.
SQLib	Set this option to Yes to generate diagnostic information for the SQLib library. The default is No.
DocFactory	Set this option to Yes to generate diagnostic information for the Document Factory. The default is No.
DXMLib	Set this option to Yes to generate diagnostic information for the DXMLIB library. The default is No.

Here is an example:

```
< Debug_Switches >
    Show_Debug_Options      = No
    Enable_Debug_Options    = No
    DBLib                   = No
    WIPLib                  = No
    ARCLib                  = No
    SQLib                   = No
    DocFactory              = No
    DXMLib                  = No
```

AFGJOB_1.JDT File

This file provides the Documaker rules to run for the Assembler process under the Document Factory. You can find this file in the \deflib subdirectory under the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section. Here is an example:

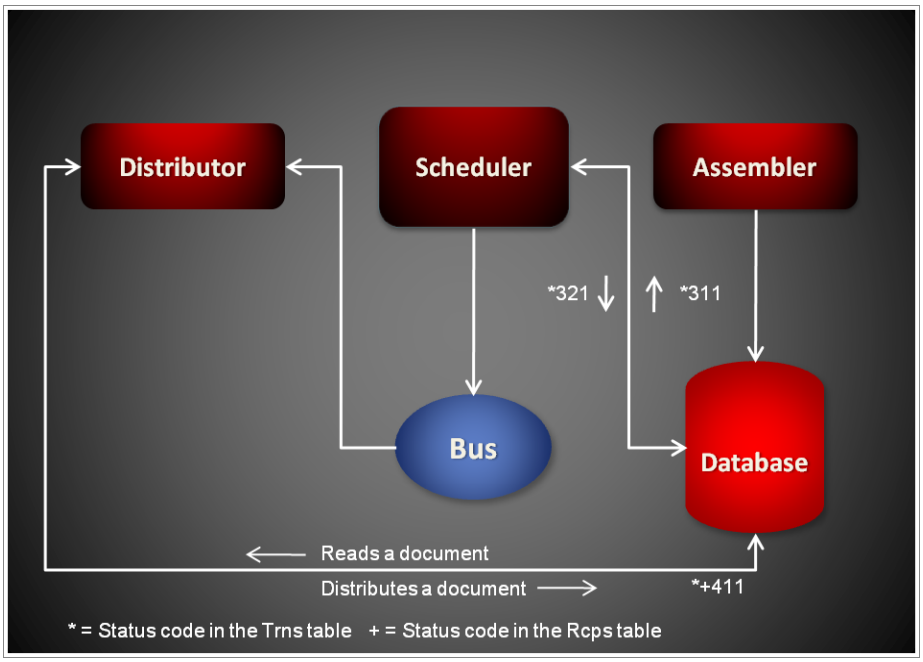
```
/* JDT Rules for Single-Step Processing Batching By Recipient. */
;RULStandardJobProc;1;Always the first job level rule;
...
;BuildMasterFormList;1;4;
/*Added to Allow WIP and Archive from Documaker*/
/* Every form set in this base uses these rules. */
;GenDocFactory;2;DocFactory Phase 1;
;RunTriggers;2;;
;ResetOvFlw;2;;
;ProcessQueue;2;PostPaginationQueue;
;PaginateAndPropogate;2;;
;Ext2GVM;;!/DocumentRequest/PackageInfo/TranName, 1,100,TRNNAME,S;
;RequiredFieldCheck;;;
/* Every image in this base uses these rules. */
;StandardImageProc;3;Always the first image level rule;
/* Every field in this base uses these rules. */
;StandardFieldProc;4;Always the first field level rule;
```

CONFIGURING THE DISTRIBUTOR

The Distributor process determines who should get the published documents. It is deployed and managed by the Supervisor process and it monitors an input queue and waits for notification messages from the Scheduler process that there are transactions ready for processing.

Once a notification message is received, the Distributor retrieves the NA/POL document data for a transaction from a record in the TRNS table and creates associated recipient records in the RCPS table.

The Distributor process typically runs after the Assembler process and reads input from TRNS records updated by the Assembler.



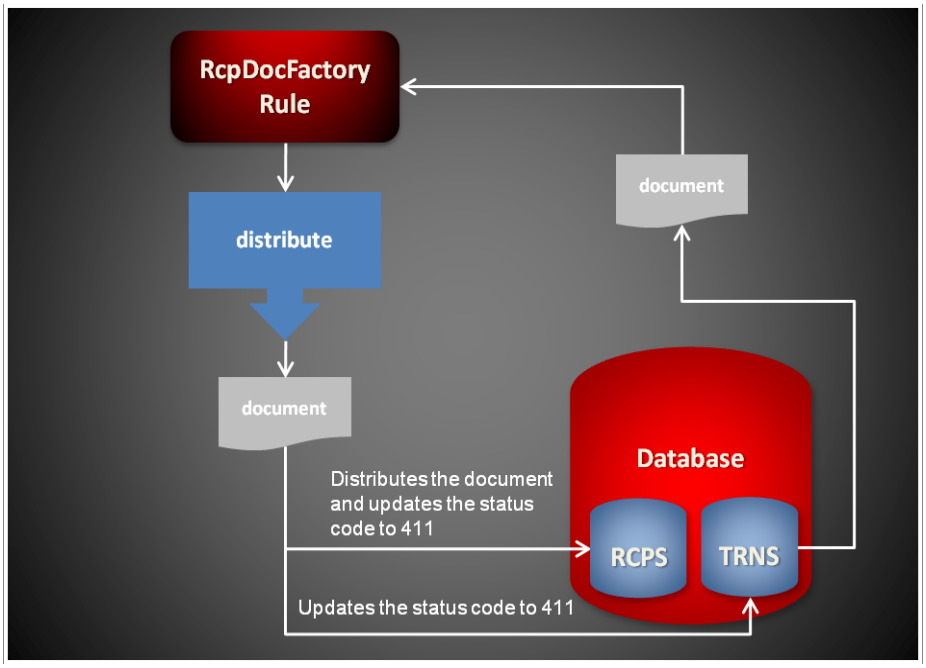
Each notification message received by the Distributor provides the transaction ID for a transaction in TRNS table that needs distributing. Here is an example of a message:

```
<?xml version="1.0" encoding="UTF-8"?>
<TransactionTicket
    xmlns="oracle/documaker/schema/tables/trns"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<TRN_ID>101</TRN_ID>
</TransactionTicket>
```

USING THE RCPDOCFACTORY RULE

The Distributor uses the RcpDocFactory rule to perform basic transaction processing and housekeeping. This rule performs these tasks:

- Loads the NA/POL document data from a TRNS record
- Creates the applicable recipient records in the RCPS table
- Updates the status code for the transaction in the TRNS and RCPS records so the Scheduler process can notify the next process in the assembly line



Here is an overview of what the RcpDocFactory rule does:

Initialization	Loads the transaction status.
PreProc	• Reads the transaction record from the TRNS table by the TRN_ID. • Validates the transaction status is set to *Distributor Start*. • Updates the transaction status to *Distributor Processing*. • Loads the extract data from the transaction record. • Sets the NA/POL data type based on the TRNNAPOLTYPE column in the transaction record. • Loads the form set from data in the transaction record.
PostProc	• Writes the recipient records to the RCPS table. • Sets the transaction status to *Distributor End*.

Note By default, if the ADR_SELECTED value for the recipient is set to zero, the recipient record will not be written. If, however, you set the RCBCheckSelected option in the FSISYS.INI file to No, the recipient is written to the RCPS table.

Starting and Stopping the Distributor

To	Then
Verify the Distributor is running	Verify there is a running process with the name docfactory_distributor.
Start the Distributor	Place the distributor.jar file in the deploy directory of Document Factory.
Stop the Distributor	Remove the distributor.jar file from the deploy directory of Document Factory.

Note The distributor.jar configuration file is uncompressed and deployed to the temp\distributor directory. This directory becomes the working directory for the Distributor. All output, including Log4J output, uses this directory as the starting directory.

Using Distributor Configuration Resources

The configuration information for the Distributor is stored in these resources:

Resource	Contains the
distributor.jar file	Minimal startup configuration information.
.bindings file	Java Naming and Directory Interface (JNDI) data sources.
APPCONFIGCONTEXT table	Configuration options.
ALCONFIGCONTEXT table	Configuration options for the Distributor status codes and message bus.
fsiuser_2.ini file	INI options specific to the Distributor process.
fsisys.ini file	INI options that are common to the Assembler, Distributor, and Presenter processes.
afgjob_2.jdt file	Documaker rules run by the Distributor process.

distributor.jar

The distributor.jar file is located in the \deploy subdirectory of the Document Factory. It contains these configuration resources:

Component	Description
deploy.properties	Contains the minimal startup configuration information.
log4j.xml	Used to capture Log4J diagnostic and error output during start up. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j/
log4j.dtd	Used by the log4j.xml file.

deploy.properties File

The deploy.properties file is extracted and placed in the temp\distributor working directory. This file contains the minimal startup configuration options used to read the configuration for the Distributor from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables:

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Distributor configuration.

Option	Description
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Distributor configuration.
application.id	The value of APP_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Distributor configuration.
config	The configuration name for the Distributor. The default is Distributor. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Distributor configuration.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=6
config=Distributor
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

log4j.xml File

The log4j.xml file is extracted and placed in the temp/distributor working directory. The log4j.xml file contains loggers used during start up of the Distributor, prior to the Distributor loading the Log4J configuration from the APPCONFIGCONTEXT table. See the Log4J configuration options in the *APPCONFIGCONTEXT Table* on page 180 for more information.

.bindings File

The .bindings file is located in the config\context subdirectory of the Document Factory. It contains the Java Naming and Directory Interface (JNDI) data sources used by the Distributor. Each JNDI data source contains these configuration options:

Option	Description
ClassName	The data source fully-qualified class name. Use the javax.sql.DataSource value.

Option	Description
FactoryName	The data source factory fully-qualified class name. Use the org.apache.commons.dbcp.BasicDataSourceFactory value. The BasicDataSourceFactory class supports connection pooling.
driverClassName	The Java Database Connectivity (JDBC) driver class name.
url	The JDBC URL.
maxOpenPreparedStatements	The maximum number of prepared statements to cache in the connection pool. Use the value -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	How often the idle object evictor thread should run and perform clean up of the stale connection handles. Use the value -1 to disable the idle object evictor thread.
validationQuery	A validation query that should be run when borrowing objects from the connection pool.
username	The JDBC user name.
password	The JDBC password.
testOnBorrow	Set this option to Yes if validationQuery should be used when borrowing an object from the connection pool. The default is No.
initialSize	The initial connection pool size.
maxActive	The maximum number of active connections in the pool.
maxIdle	The maximum number of idle connections in the pool.
minIdle	The minimum number of idle connections in the pool.
maxWait	The maximum time (in milliseconds) to wait for a connection object to be retrieved from the pool before issuing an error.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
```

```
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type= maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
DMKRConfig/RefAddr/9/Type=maxWait
#Unix friendly Documaker Doc. Factory JNDI DataSource
DMKRFactory/ClassName=javax.sql.DataSource
DMKRFactory/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRFactory/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRFactory/RefAddr/0/Encoding=String
DMKRFactory/RefAddr/0/Type=driverClassName
DMKRFactory/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRFactory/RefAddr/1/Encoding=String
DMKRFactory/RefAddr/1/Type=url
DMKRFactory/RefAddr/10/Content=-1
DMKRFactory/RefAddr/10/Encoding=String
DMKRFactory/RefAddr/10/Type=maxOpenPreparedStatements
DMKRFactory/RefAddr/11/Content=-1
DMKRFactory/RefAddr/11/Encoding=String
DMKRFactory/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRFactory/RefAddr/12/Content=select 1 from dual
DMKRFactory/RefAddr/12/Encoding=String
DMKRFactory/RefAddr/12/Type=validationQuery
DMKRFactory/RefAddr/2/Content=dmkr_asline
DMKRFactory/RefAddr/2/Encoding=String
DMKRFactory/RefAddr/2/Type=username
DMKRFactory/RefAddr/3/Content=oracle12
DMKRFactory/RefAddr/3/Encoding=String
DMKRFactory/RefAddr/3/Type=password
DMKRFactory/RefAddr/4/Content=true
DMKRFactory/RefAddr/4/Encoding=String
DMKRFactory/RefAddr/4/Type=testOnBorrow
DMKRFactory/RefAddr/5/Content=1
DMKRFactory/RefAddr/5/Encoding=String
DMKRFactory/RefAddr/5/Type=initialSize
DMKRFactory/RefAddr/6/Content=8
DMKRFactory/RefAddr/6/Encoding=String
DMKRFactory/RefAddr/6/Type= maxActive
DMKRFactory/RefAddr/7/Content=8
DMKRFactory/RefAddr/7/Encoding=String
DMKRFactory/RefAddr/7/Type=maxIdle
DMKRFactory/RefAddr/8/Content=0
DMKRFactory/RefAddr/8/Encoding=String
DMKRFactory/RefAddr/8/Type=minIdle
DMKRFactory/RefAddr/9/Content=60000
```

```
DMKRFactory/RefAddr/9/Encoding=String
DMKRFactory/RefAddr/9/Type=maxWait
```

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Distributor*:

Option	Description
StartCommand	Defines the command to use to start the Distributor. Include the full path.
StartArguments	Defines the initialization arguments used to start the Distributor. Here is an example: /ini=fsiuser_2.ini /debug=0 /phase=2
env.mode.*	The environment variables the process expects to run. The Supervisor creates an environment variable for each env.mode.* configuration option it encounters. The naming convention is env.*mode*.*name*. Where *mode* can be either zero (0), meaning prepend, one (1), meaning append, or two (2), meaning overwrite, and *name* is the name of the environment variable. When the mode is not defined, the default is two (2). An example of an env.mode.* variable would be env.0.PATH or env.ORACLE_HOME. Notice the second example uses the default overwrite mode.
StartDirectory	Defines the start up directory. Here is an example: c:/oracle/oracle_insurance_1/documaker/mstrres/dmres
Instances	(Optional) The number of instances the Supervisor should start for a process configuration. The default is two (2).
UseLoadBalancing	(Optional) This option controls wether the Supervisor checks the idle time of a process's instances that are running and starts additional ones when all of them are busy. Instances are considered busy when their idle time is less than the value provided in the MinIdleTimeSeconds option. The Supervisor uses the value provided in the IdleTimeChecks option to determine the number of idle time checks to run before it starts additional instances. When additional instances are started for load balancing purposes, they are shut down by the Supervisor if their idle time exceeds the value in the MaxIdleTimeSeconds option. The maximum number of instances running is the value for the MaxInstances option (including the instances configured in the Instances option). The Supervisor checks the idle time of the current instances at the interval specified in the IdleTimeCheckIntervalSeconds and if all are busy, it starts an additional number of instances equal to the value provided in the IncrementCount option. Please note that the Supervisor does not start checking the busy time of the current instances until the time provided in the IdleTimeCheckDelaySeconds option elapses. Make sure the value for the delay is ample enough to provide for all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. You can enter Yes or No. The default is Yes.
MaxInstances	(Optional) This option controls the maximum number of instances that can run when the UseLoadBalancing option is enabled. The default is the number of processors times two.
IncrementCount	(Optional) This option controls how many additional instances are started during the current check when all instances running are busy and the UseLoadBalancing option is enabled. The default is two (2).

Option	Description
IdleTimeCheckIntervalSeconds	(Optional) This option controls how often the Supervisor checks the idle time of the instances that are running to determine if they are busy so it can start additional ones when the UseLoadBalancing option is enabled. The default is 10 seconds.
IdleTimeCheckDelaySeconds	(Optional) This option controls the initial delay before the first idle time check is performed by the Supervisor when the UseLoadBalancing option is enabled. This time should be ample enough to allow all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. The default is 120 seconds.
IdleTimeChecks	(Optional) This option defines the number of consecutive idle time checks that must fail, meaning all instances were busy during each check, before more instances are started when the UseLoadBalancing option is enabled. Each check takes place at the IdleTimeCheckIntervalSeconds interval. The default is 12.
MinIdleTimeSeconds	(Optional) This option controls the minimum idle time for each instance. The idle time represents how long it has been since an instance processed the last request. If the Supervisor detects an instance has an idle time less than the value provided for this option, it considers it busy for the purpose of load balancing. The default is 5 seconds.
MaxIdleTimeSeconds	(Optional) This option controls the maximum idle time for an additional instance. The idle time represents how long it has been since an instance performed processing. If the Supervisor detects an instance, which was started for the purpose of load balancing, has reached an idle time greater than the value provided for this option, it sends the instance a shutdown request. The default is 120 seconds.
MaxTransactions	(Optional) This option controls the maximum number of transactions an instance can process before it is restarted by the Supervisor. The default is -1, which disables this option.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before it is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
OrderedRestartIntervalSeconds	(Optional) This option controls the interval used for restarting each process instance in a sequential/ordered manner when the MaxTransactions or MaxUpTime options are used. The Supervisor restarts one instance at a time and waits for an amount of time equal to the value specified for this option before it restarts the next one and so on until it has restarted all of them. If you set this option to less than 60 seconds, you can negatively affect performance. The default is 60 seconds.
WatchList	A comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted.
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is 5.

Here is an example:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/bin/docfactory_distributor
StartArguments	/ini=fsiuser_2.ini /debug=0 /phase=2
env.0.PATH	/oracle_home/*InstallationLocation*/oracle_instantclient_11_2,/oracle_home/*InstallationLocation*/jre/bin,/oracle_home/*InstallationLocation*/jre/bin/client,/oracle_home/*InstallationLocation*/bin
env.ORACLE_HOME	/oracle_home/*InstallationLocation*/bin
env.NLS_LANG	AMERICAN_AMERICA.AL32UTF8
env.TNS_ADMIN	/oracle_home/*InstallationLocation*/oracle_instantclient_11_2/NETWORK/ADMIN
env.JVM_OPTIONS	-Xmx256m,-Duser.name=oracle,-Dlog4j.configuration=/oracle_home/*InstallationLocation*/docfactory/temp/distributor/log4j.xml,-Dlog4j.file=/oracle_home/*InstallationLocation*/docfactory/temp/distributor/logs/log4j.log,-Djndi.context=/oracle_home/*InstallationLocation*/docfactory/config/context,-Dfactory.jndi.name=DMKRFactory,-Dconfig.jndi.name=DMKRConfig,-Dschema=DMKR_ASLINE
StartDirectory	/oracle_home/*InstallationLocation*/mstrres/correspondence
Instances	2
UseLoadBalancing	No
MaxInstances	8
IncrementCount	1
IdleTimeCheckIntervalSeconds	15
IdleTimeCheckDelaySeconds	240
IdleTimeChecks	5
MinIdleTimeSeconds	5
MaxIdleTimeSeconds	120
MaxTransactions	-1
MaxReportIntervalSeconds	180
MaxUpTimeSeconds	-1
WaitForShutdownSeconds	60
OrderedRestartIntervalSeconds	60

InstallationLocation = The installation location where you installed Document Factory.

Option	Value
WatchList	/oracle_home/*InstallationLocation*/mstrres/correspondence/fsiuser_2.ini,/ oracle_home/*InstallationLocation*/mstrres/correspondence/fsisys.ini
MaxRestarts	5

InstallationLocation = The installation location where you installed Document Factory.

Log4J configuration options

For specific information on the Log4J configuration options, see *Defining Log4J Configuration Options* on page 277.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Distributor-Ready	This is the status code that indicates a transaction is ready to be sent to the Distributor. The default is 311.
Distributor-ACK	This is the status code that indicates a transaction has been received and it is being processed by Distributor. The default is 331.
Distributor-Error	This is the status code that indicates the Distributor process failed to process a transaction. The default is 341.

Here is an example:

Option	Value
Distributor-Ready	311
Distributor-ACK	331
Distributor-Error	341

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
DistributorQueue	The name of the queue the Distributor uses to receive notifications from the Scheduler process.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf
DistributorQueue	jms/distributor_requestq
TimeoutSeconds	5

FSIUSER_2.INI File

You can find this file in the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section. It provides INI options required to run the Distributor process under the Document Factory.

Database Handler Definition

These options are read from the DBHandler:WIP_ODBC_ORA INI control group:

Option	Description
Class	The class name for the ODBC handler
Server	The ODBC DSN name.
UserID	The user ID for the ODBC DSN.
Password	The password for the ODBC DSN.
Debug	Set this option to Yes if you want diagnostic information generated for ODBC operations.

Here is an example:

```
< DBHandler:WIP_ODBC_ORA >
    Class  = ODBC
    Passwd = ********
    Server = DMKR_ASLINE
    UserID = dmkr_asline
    Debug  = No
```

RCPS Table Definition

These options are read from the DBTable:RCPS INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:RCPS >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = RCP_ID
```

WIP Index Table Definition

These options are read from the DBTable:WIP INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:WIP >
    DBHandler = WIP_ODBC_ORA
    UniqueTag = FORMSETID
```

WIP Data Table Definition

These options are read from the DBTable:WIPData INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:WIPData >
   DBHandler = WIP_ODBC_ORA
   UniqueTag = FORMSETID
```

Extract Table Definition

These options are read from the DBTable:EXTR INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:EXTR >
   DBHandler = WIP_ODBC_ORA
   UniqueTag = TRN_ID
```

Jobs Table Definition

These options are read from the DBTable:JOBS INI control group:

Option	Description
DBHandler	The name of the database handler.

Option	Description
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:JOBS >
    DBHandler = WIP_ODBC_ORA
    UniqueTag = JOBUNIQUE_ID
```

DFD Definitions

These options are read from the WIPData INI control group:

Option	Description
DatabaseWIP	Set this option to Yes if you want to store WIP in a database. The default is No.
File	The internal name of the WIP table.
WIPDFDFile	The name of the WIP index DFD file.
WIPDataDFD	The name of the WIP data DFD file for XML NA/POL.
WIPDsDataDFD	The name of the WIP data DFD file for combined NA/POL.
JOBS	The name of the Jobs table.
JOBSDFD	The name of the Jobs index DFD file.
BCHS	The name of the batch table.
BCHSDFD	The name of the batch DFD file.

Here is an example:

```
< WIPData >
    DatabaseWIP = Yes
    File = WIP
    WIPDFDFile = .\deflib\trnsdf.dfd
    WIPDataDFD = .\deflib\docdata.dfd
    WIPDsDataDFD = .\deflib\dsdata.dfd
    JOBS  = JOBS
    JOBSDFD = .\deflib\jobs.dfd
    BCHS = BCHS
    BCHSDFD = .\deflib\BCHS.dfd
```

This option is read from the Data INI control group:

Option	Description
RCBDFDFile	The name of the RCPS DFD file.

Here is an example:

```
< Data >
    RCBDFDFile = .\deflib\rcbdocf.dfd
```

ODBC File Conversion

These options are read from the ODBC_FileConvert INI control group:

Option	Description
WIP	The actual name of the database table that corresponds to the internal WIP table name. This is the WIP index table.
WIPData	The actual name of the database table that corresponds to the internal WIPData table name. This is the WIP data table.
EXTR	The actual name of the database table that corresponds to the internal EXTR table name. This is the extract data table.

Here is an example:

```
< ODBC_FileConvert >
    WIP = TRNS
    WIPData = TRNS
    EXTR = TRNS
```

Document Factory Options

These options are read from the DocFactory INI control group:

Option	Description
AddresseeType	The recipient addressee type. The default is ADDRESSEE.
StandardType	The recipient standard type. The default is STANDARD.
Distributor_StatusCode	The ready for processing status code for the Distributor. The default is P.
DistError_StatusCode	The error status code for the Distributor. The default is E.
DistManual_StatusCode	The status code for transactions flagged for manual processing. The default is W.
DistManual_ApprovalState	The approval state code for transactions flagged for manual processing. The default is 30.
Distributor_Start	The Distributor start status code. The default is 321.
Distributor_Processing	The Distributor ACK status code. The default is 331.
Distributor_Error	The Distributor error status code. The default is 341.
Distributor_End	The Distributor end status code. The default is 411.
Bindings	The path location for the Java Naming and Directory Interface (JNDI) .bindings file containing the data source information for JNI code. The default is /docfactory/config/context/.

Here is an example:

```
< DocFactory >
   AddresseeType = ADDRESSEE
   StandardType = STANDARD
   Distributor_StatusCode = P
   DistError_StatusCode = E
   DistManual_StatusCode = W
   DistManual_ApprovalState = 30
   Distributor_Start = 321
   Distributor_Processing = 331
```

```
   Distributor_Error = 341
   Distributor_End = 411
   Bindings = /oracle_home/InstallationLocation/docfactory/config/
context
```

Logging messages to the database

This option is read from the Environment INI control group:

Option	Description
JLOG_Enabled	Set this option to Yes to redirect warning and error messages to the LOGS and ERRS tables instead of being written to the trace file. The default is No.

Here is an example:

```
< Environment >
    JLOG_Enabled = Yes
```

Controlling log output

These options are read from the DocFactory_Distributor:JLog INI control group:

Option	Description
LogLogger	The name of the Log4J logger used to log warning messages to the LOGS table. This name should match the Log4J logger name in log4j.xml file.
ErrorLogger	The name of the Log4J logger used to log error messages to the ERRS table. This name should match the Log4J logger name in log4j.xml file.
ColumnNames	A comma-delimited list of table column names to GVM mappings. Is used by the loggers to capture the GVM values and set them as the column values. The format for each comma-delimited token can be ColumnName=GVMName or just ColumnName.
BufferSize	The maximum buffer size for messages. This value should match the length of the LOGMESSAGE and ERRMESSAGE columns.
Debug	Set this option to Yes if you want diagnostic output generated for the Logger. The default is No.
LogError	Set this option to No if you want the system to suppress all error messages. The default is Yes, which tells the system to issue error messages.
LogWarning	Set this option to Yes if you want the system to issue warning messages. The default is No, which suppresses all warning messages.

Here is an example:

```
< DocFactory_Distributor:JLog >
    LogLogger = LogLogger
    ErrorLogger = ErrorLogger
    BufferSize = 2000
    Debug = No
    LogError = Yes
    LogWarning = No
    ColumnNames =
JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_
ID
```

FSISYS.INI File

This file provides the INI options required to run the Distributor process under Document Factory. You can find this file in the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section.

Enabling Document Factory code

These options are read from the RunMode INI control group:

Option	Description
DocFactory	Must be set to Yes if you are using Document Factory. To facilitate legacy Documaker Server processing, this option defaults to No.
RCBCheckSelected	This option turns on or off the recipient batch filter where RCPS records are not written if the record count returned is zero (0). The default is Yes.

Here is an example:

```
< RunMode >
    DocFactory = Yes
    RCBCheckSelected = Yes
```

Enabling Debug options

These options are read from the Debug_Switches INI control group:

Option	Description
Show_Debug_Options	Set this option to Yes to show all debug options. The default is No.
Enable_Debug_Options	Set this option to Yes to enable all debug options. The default is No.
DBLib	Set this option to Yes to generate diagnostic information for the DBLIB library. The default is No.
WIPLib	Set this option to Yes to generate diagnostic information for the WIPLIB library. The default is No.
ARCLib	Set this option to Yes to generate diagnostic information for the ARCLIB library. The default is No.
SQLib	Set this option to Yes to generate diagnostic information for the SQLib library. The default is No.
DocFactory	Set this option to Yes to generate diagnostic information for the Document Factory. The default is No.
DXMLib	Set this option to Yes to generate diagnostic information for the DXMLIB library. The default is No.

Here is an example:

```
< Debug_Switches >
    Show_Debug_Options      = No
    Enable_Debug_Options    = No
    DBLib                   = No
    WIPLib                  = No
    ARCLib                  = No
    SQLib                   = No
```

```
DocFactory              = No
DXMLib                  = No
```

AFGJOB_2.JDT File

This file provides the Documaker Server rules to run for the Distributor process under Document Factory. You can find this file in the \deflib subdirectory under the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section. Here is an example:

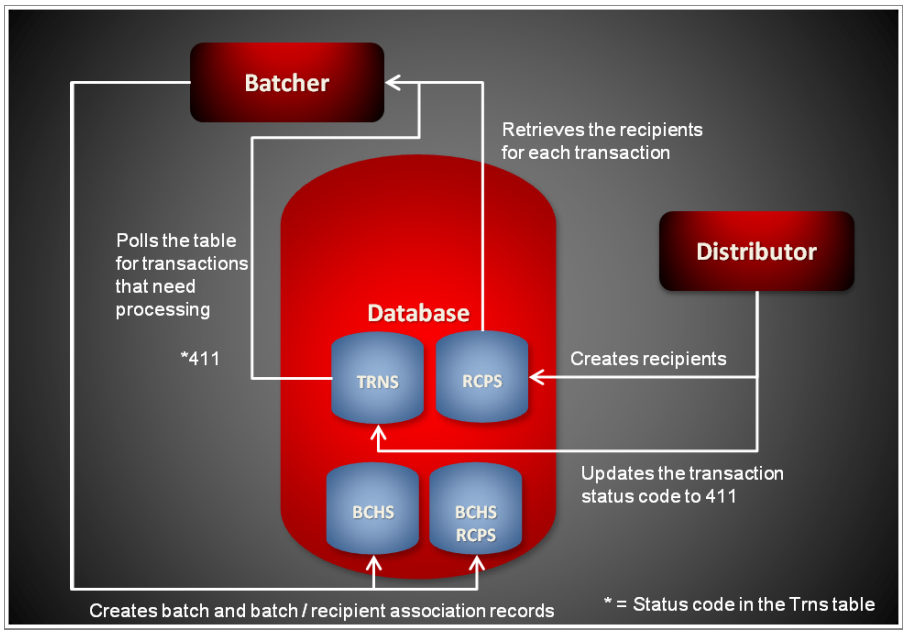
```
/*  JDT Rules for Single-Step Processing Batching By Recipient. */
;RULStandardJobProc;1;Always the first job level rule;
;SetErrHdr;1;*:;
...
;SetErrHdr;1;*:-----------------------------------------------;
;JobInit1;1;;
/* Every form set in this base uses these rules. */
;RcpDocFactory;2;DocFactory Phase 2;
;BatchingByPageCountPerRecipINI;;;
;BatchingByRecipINI;2;;
;RequiredFieldCheck;;;
/* Every image in this base uses these rules. */
;WIPImageProc;3;Always the first image level rule;
/* Every field in this base uses these rules. */
;WIPImageProc;4;Always the first field level rule;
```

CONFIGURING THE BATCHER

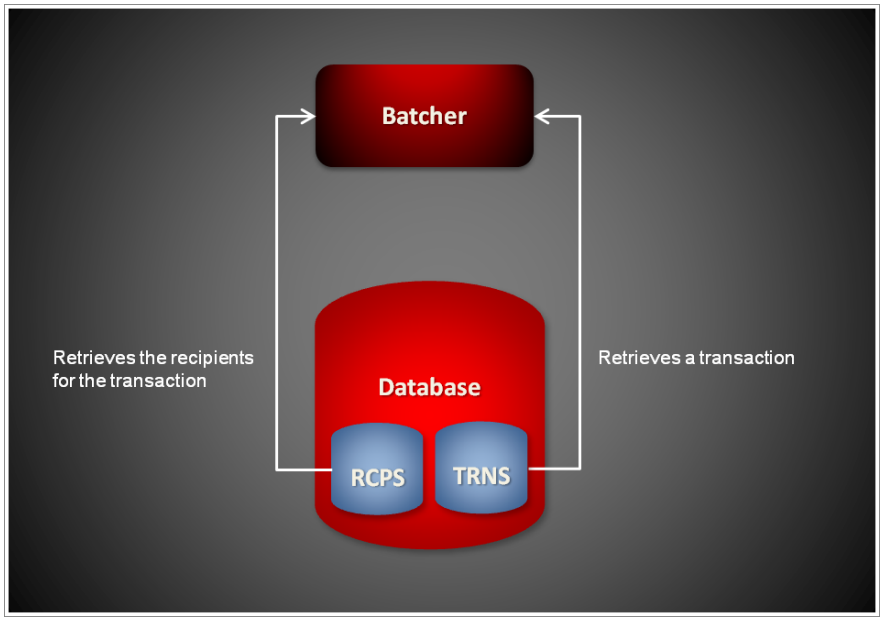
The Batcher process creates and associates batches with recipients. It is deployed and managed by the Supervisor process and it monitors the TRNS table for transactions with a status code of *Presenter-Ready* (411). The Batcher then retrieves a transaction record, looks up the recipients for it, and generates batch and batch-to-recipient association records for it.

The Batcher process typically runs after the Distributor process and reads input from TRNS and RCPS table records created by the Distributor.

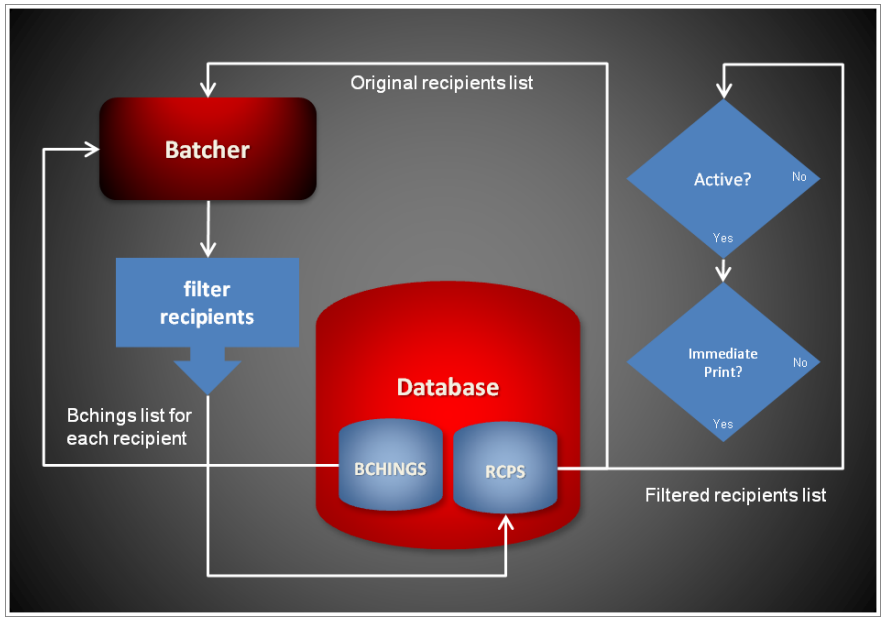
Note The batcher process creates the Batch and Batch to Recipient records. The Scheduler tells the Archiver, PubNotifier and Publisher when to process a transaction.



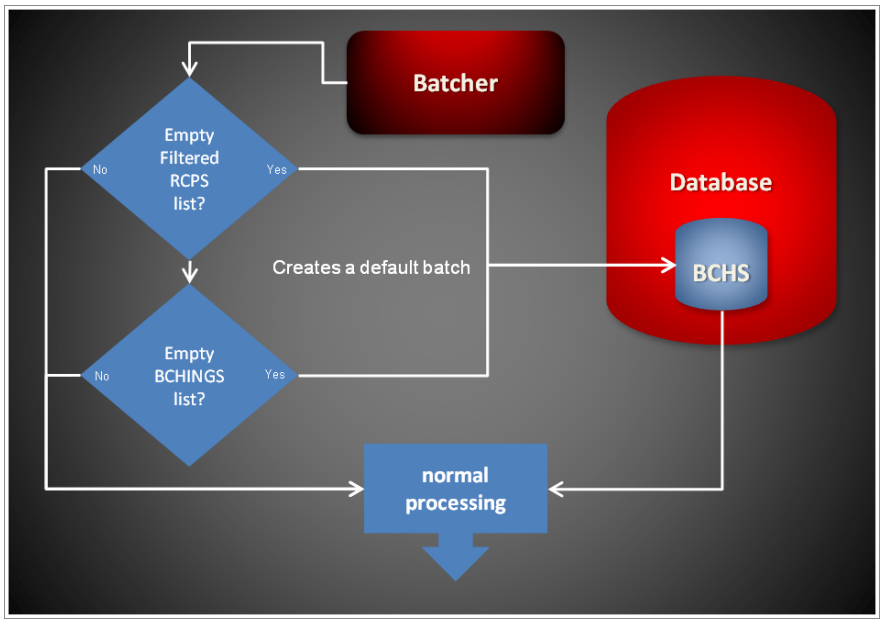
The Batcher reads an input record from the TRNS table. It then looks up the matching RCPS records for the TRNS record.



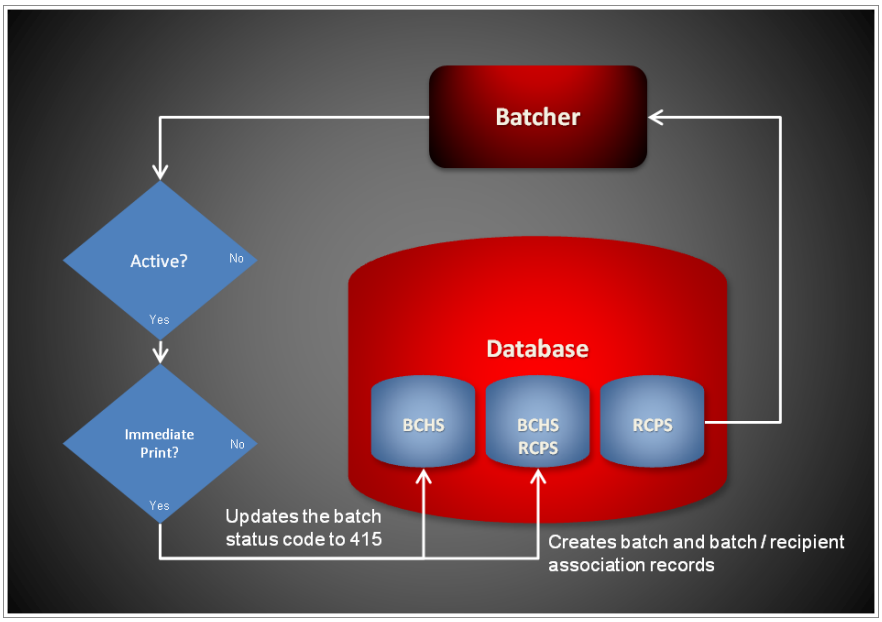
The Batcher also looks up BCHINGS records for each RCPS record and applies any RCPS filtering logic defined in BCHINGSELECTRULE column to the list of RCPS records, otherwise it leaves the list alone. If there are any RCPS records in the final recipients list, the Batcher also uses the information in the BCHINGS records to determine if the batch is active and if it is defined as an immediate or scheduled print batch.



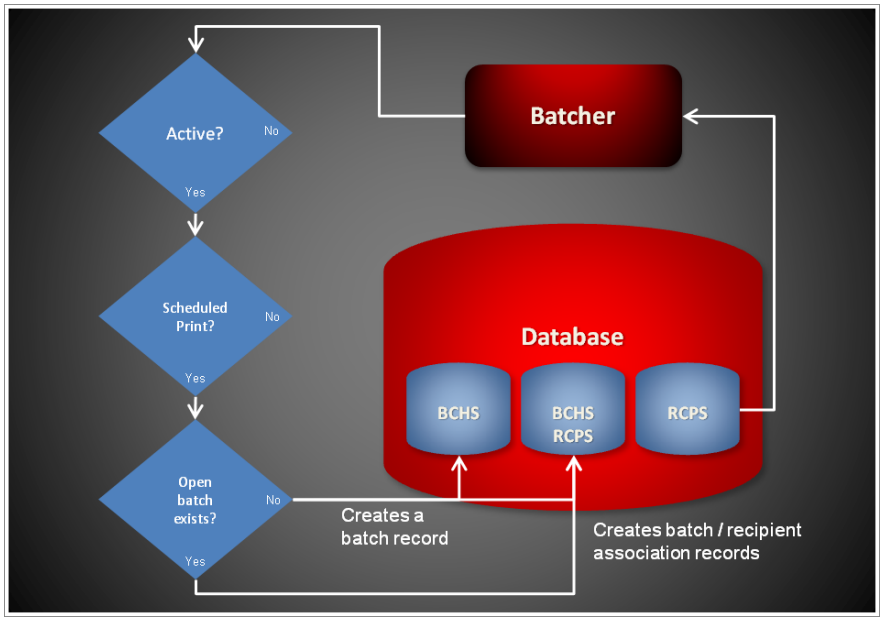
If there are no records returned in the BCHINGS lookup or there are no RCPS records in the list after applying the RCPS filtering logic the Batcher creates a new default batch and assigns the original RCPS records to it.



If there are records in the RCPS list and a batch is defined as active and for immediate print the Batcher creates a new BCHS record and new BCHS_RCPS records and sets the appropriate status code for the batch record so the Scheduler can notify the Presenter.

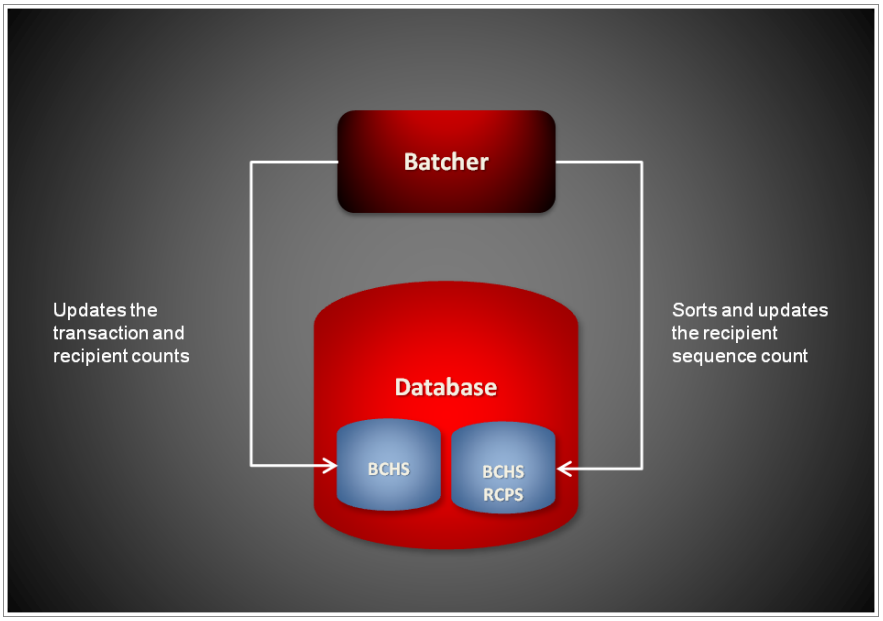


If there are records in the RCPS list and a batch is defined as active and for scheduled print, the Batcher first checks if an open scheduled BCHS record exists. If one exists, the Batcher uses that record instead of creating a new BCHS record. The Batcher then creates new BCHS_RCPS records and associates them with the batch record. The Scheduler process can then check the batch and determine when it needs to be processed and closed.



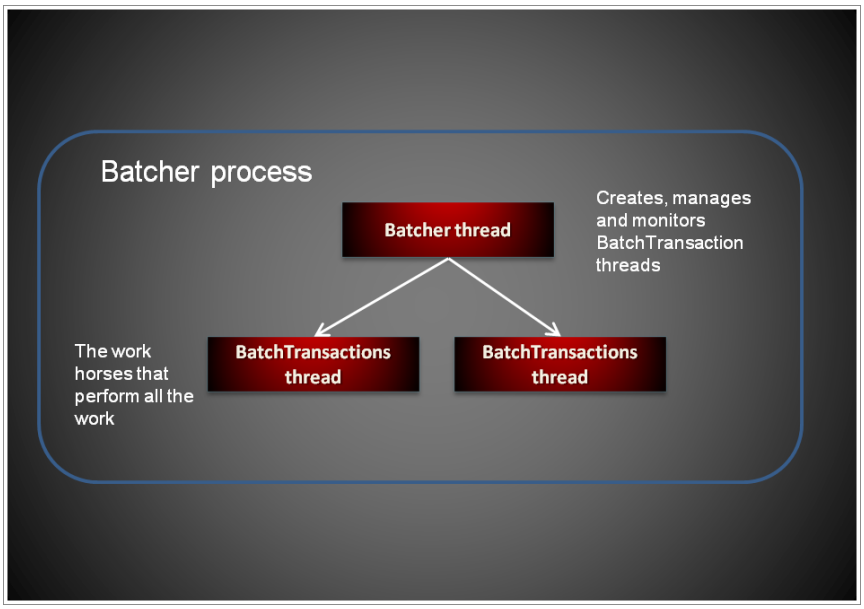
Note A scheduled batch contains a date/time stamp value in BCHSTARTINGTIME column that indicates when the batch should be processed and closed. This is how the Batcher and Scheduler processes determine if a batch is open or closed.

Finally, the Batcher updates the transaction count and recipient count in the BCHTRNCOUNT and BCHRCPCOUNT columns for the batch record and sorts the batch/recipient association records in the BCHS_RCPS table and updates their RCPSEQ column. Sorting logic can also be included in the BCHINGSORTRULE column in the BCHINGS table.



Multi-threaded Architecture

The Batcher process is multi-threaded. The Batcher thread is the main thread and it is responsible for creating, starting, and monitoring the subordinate BatchTransactions threads that perform all the batching work. The Batcher thread can create more than one instance of the BatchTransactions thread. The BatchTransactions thread instance count is controlled by the WorkerThreads configuration option in the Batcher configuration section. All batching logic is performed by the BatchTransactions threads.



STARTING AND STOPPING THE BATCHER

To	Then
Verify the Batcher is running.	Verify there is a running process with the name docfactory_batcher.
Start the Batcher	Place the batcher.jar file in the deploy directory of Document Factory.
Stop the Batcher	Remove the batcher.jar file from the deploy directory of Document Factory.

Note The batcher.jar configuration file is uncompressed and deployed to the temp\batcher directory. This directory becomes the working directory for the Batcher. All output, including Log4J output, uses this directory as the starting directory.

USING BATCHER CONFIGURATION RESOURCES

The configuration information for the Batcher is stored in these resources:

Resource	Description
batcher.jar file	Contains the minimal startup configuration information.
.bindings file	Contains the Java Naming and Directory Interface (JNDI) data sources.
APPCONFIGCONTEXT table	Contains configuration options.
ALCONFIGCONTEXT table	Contains configuration options for the Batcher status codes and message bus.

batcher.jar File

The batcher.jar file is located in the \deploy subdirectory of the Document Factory. It contains these configuration resources:

Component	Description
deploy.properties	Contains the minimal startup configuration information.
log4j.xml	Used to capture Log4J diagnostic and error output during start up. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j/
log4j.dtd	Used by the log4j.xml file.

deploy.properties File

The deploy.properties file contains the minimal startup configuration options used to read the configuration for the Batcher from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables. It is extracted and placed in the temp\batcher working directory.

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Batcher configuration.
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Batcher configuration.
application.id	The value of APP_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Batcher configuration.
config	The configuration name for the Batcher. The default is Batcher. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Batcher configuration.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.

Option	Description
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=8
config=Batcher
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

log4j.xml File

The log4j.xml file is extracted and placed in the temp/batcher working directory. The log4j.xml file contains loggers used during start up of the Batcher, prior to the Batcher loading the Log4J configuration from the APPCONFIGCONTEXT table. See the Log4J configuration options in the *APPCONFIGCONTEXT Table* on page 199 for more information.

.bindings File

The .bindings file contains the Java Naming and Directory Interface (JNDI) data sources used by the Batcher. Each JNDI data source contains these configuration options. It is located in the config\context subdirectory of Document Factory.

Option	Description
ClassName	The data source fully-qualified class name. Use the javax.sql.DataSource value.
FactoryName	The data source factory fully-qualified class name. Use the org.apache.commons.dbcp.BasicDataSourceFactory value. The BasicDataSourceFactory class supports connection pooling.
driverClassName	The Java Database Connectivity (JDBC) driver class name.
url	The JDBC URL.
maxOpenPreparedStatements	The maximum number of prepared statements to cache in the connection pool. Use the value -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	How often the idle object evictor thread should run and perform clean up of the stale connection handles. Use the value -1 to disable the idle object evictor thread.
validationQuery	A validation query that should be run when borrowing objects from the connection pool.

Option	Description
username	The JDBC user name.
password	The JDBC password.
testOnBorrow	Set this option to Yes if validationQuery should be used when borrowing an object from the connection pool. The default is No.
initialSize	The initial connection pool size.
maxActive	The maximum number of active connections in the pool.
maxIdle	The maximum number of idle connections in the pool.
minIdle	The minimum number of idle connections in the pool.
maxWait	The maximum time (in milliseconds) to wait for a connection object to be retrieved from the pool before issuing an error.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type= maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
```

```
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
DMKRConfig/RefAddr/9/Type=maxWait
#Unix friendly Documaker Doc. Factory JNDI DataSource
DMKRFactory/ClassName=javax.sql.DataSource
DMKRFactory/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRFactory/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRFactory/RefAddr/0/Encoding=String
DMKRFactory/RefAddr/0/Type=driverClassName
DMKRFactory/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRFactory/RefAddr/1/Encoding=String
DMKRFactory/RefAddr/1/Type=url
DMKRFactory/RefAddr/10/Content=-1
DMKRFactory/RefAddr/10/Encoding=String
DMKRFactory/RefAddr/10/Type=maxOpenPreparedStatements
DMKRFactory/RefAddr/11/Content=-1
DMKRFactory/RefAddr/11/Encoding=String
DMKRFactory/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRFactory/RefAddr/12/Content=select 1 from dual
DMKRFactory/RefAddr/12/Encoding=String
DMKRFactory/RefAddr/12/Type=validationQuery
DMKRFactory/RefAddr/2/Content=dmkr_asline
DMKRFactory/RefAddr/2/Encoding=String
DMKRFactory/RefAddr/2/Type=username
DMKRFactory/RefAddr/3/Content=oracle12
DMKRFactory/RefAddr/3/Encoding=String
DMKRFactory/RefAddr/3/Type=password
DMKRFactory/RefAddr/4/Content=true
DMKRFactory/RefAddr/4/Encoding=String
DMKRFactory/RefAddr/4/Type=testOnBorrow
DMKRFactory/RefAddr/5/Content=1
DMKRFactory/RefAddr/5/Encoding=String
DMKRFactory/RefAddr/5/Type=initialSize
DMKRFactory/RefAddr/6/Content=8
DMKRFactory/RefAddr/6/Encoding=String
DMKRFactory/RefAddr/6/Type= maxActive
DMKRFactory/RefAddr/7/Content=8
DMKRFactory/RefAddr/7/Encoding=String
DMKRFactory/RefAddr/7/Type=maxIdle
DMKRFactory/RefAddr/8/Content=0
DMKRFactory/RefAddr/8/Encoding=String
DMKRFactory/RefAddr/8/Type=minIdle
DMKRFactory/RefAddr/9/Content=60000
DMKRFactory/RefAddr/9/Encoding=String
DMKRFactory/RefAddr/9/Type=maxWait
```

CONFIGURING THE MAIN BATCHER THREAD

The Batcher thread reads configuration information from deploy.properties file and APPCONFIGCONTEXT table.

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Batcher*:

Option	Description
StartCommand	Defines the command to use to start the Batcher. This value is used by the Supervisor to start the class specified in JavaClass configuration option.
StartArguments	Defines the initialization arguments used to start the Batcher.
JavaClass	The Java class that is used to start the worker class specified in WorkerClass configuration option. Use the oracle.documaker.process.ProcessShell value. ProcessShell class is a process shell that provides all functionality needed to communicate with the Supervisor process and to start and manage the worker class specified in WorkerClass configuration option.
JVMOptions	Any JVM options the Supervisor process uses to start the JavaClass. Here is an example: -Xmx128m -Duser.name=oracle -Djava.library.path=c:/oracle/oracle_insurance_1/documaker/bin
Instances	(Optional) The number of instances the Supervisor should start for a process configuration. The default is two (2).
UseLoadBalancing	(Optional) This option controls wether the Supervisor checks the idle time of a process's instances that are running and starts additional ones when all of them are busy. Instances are considered busy when their idle time is less than the value provided in the MinIdleTimeSeconds option. The Supervisor uses the value provided in the IdleTimeChecks option to determine the number of idle time checks to run before it starts additional instances. When additional instances are started for load balancing purposes, they are shut down by the Supervisor if their idle time exceeds the value in the MaxIdleTimeSeconds option. The maximum number of instances running is the value for the MaxInstances option (including the instances configured in the Instances option). The Supervisor checks the idle time of the current instances at the interval specified in the IdleTimeCheckIntervalSeconds and if all are busy, it starts an additional number of instances equal to the value provided in the IncrementCount option. Please note that the Supervisor does not start checking the busy time of the current instances until the time provided in the IdleTimeCheckDelaySeconds option elapses. Make sure the value for the delay is ample enough to provide for all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. You can enter Yes or No. The default is Yes.
MaxInstances	(Optional) This option controls the maximum number of instances that can run when the UseLoadBalancing option is enabled. The default is the number of processors times two.
IncrementCount	(Optional) This option controls how many additional instances are started during the current check when all instances running are busy and the UseLoadBalancing option is enabled. The default is two (2).
IdleTimeCheckIntervalSeconds	(Optional) This option controls how often the Supervisor checks the idle time of the instances that are running to determine if they are busy so it can start additional ones when the UseLoadBalancing option is enabled. The default is 10 seconds.
IdleTimeCheckDelaySeconds	(Optional) This option controls the initial delay before the first idle time check is performed by the Supervisor when the UseLoadBalancing option is enabled. This time should be ample enough to allow all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. The default is 120 seconds.
IdleTimeChecks	(Optional) This option defines the number of consecutive idle time checks that must fail, meaning all instances were busy during each check, before more instances are started when the UseLoadBalancing option is enabled. Each check takes place at the IdleTimeCheckIntervalSeconds interval. The default is 12.

Option	Description
MinIdleTimeSeconds	(Optional) This option controls the minimum idle time for each instance. The idle time represents how long it has been since an instance processed the last request. If the Supervisor detects an instance has an idle time less than the value provided for this option, it considers it busy for the purpose of load balancing. The default is 5 seconds.
MaxIdleTimeSeconds	(Optional) This option controls the maximum idle time for an additional instance. The idle time represents how long it has been since an instance performed processing. If the Supervisor detects an instance, which was started for the purpose of load balancing, has reached an idle time greater than the value provided for this option, it sends the instance a shutdown request. The default is 120 seconds.
MaxTransactions	(Optional) This option controls the maximum number of transactions an instance can process before it is restarted by the Supervisor. The default is -1, which disables this option.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before it is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
OrderedRestartIntervalSeconds	(Optional) This option controls the interval used for restarting each process instance in a sequential/ordered manner when the MaxTransactions or MaxUpTime options are used. The Supervisor restarts one instance at a time and waits for an amount of time equal to the value specified for this option before it restarts the next one and so on until it has restarted all of them. If you set this option to less than 60 seconds, you can negatively affect performance. The default is 60 seconds.
WatchList	A comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted.
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is 5.
WorkerClass	The class that extends the oracle.documaker.process.worker.Worker Thread class and is started by the class specified in JavaClass configuration option. Use the oracle.documaker.batch.Batcher value.
WorkerThreads	How many threads of WorkerClass should be created by JavaClass. You can use the value 1. The default is 1.
WorkerIntervalMillis	How often each WorkerClass thread should perform its work. The default is 5000 milliseconds.
WorkerStartDelayMillis	How long each WorkerClass thread should wait after startup and before performing any work. The default is 10000 milliseconds.
ShutdownHookClass	The class that extends the oracle.documaker.process.shutdown.ShutdownHook class. Use the oracle.documaker.batch.shutdown.BatcherShutdownHook value.
HouseKeeperClass	The class that extends the oracle.documaker.process.housekeeping.HouseKeeper class. Use the oracle.documaker.batch.housekeeping.BatcherHouseKeeper value.

Option	Description
HouseKeeperIntervalMillis	How often, in milliseconds, the HouseKeeperClass thread should perform its work. The default is 15000.
HouseKeeperStartDelayMillis	How long, in milliseconds, the HouseKeeperClass thread should wait after startup and before performing any work. The default is 30000.
IPCIntervalMillis	How often, in milliseconds, the inter-process communication (IPC) thread should perform its work. This option is used by JavaClass to report back to the Supervisor process. The default is 1000.
IPCStartDelayMillis	How long, in milliseconds, the inter-process communication (IPC) thread should wait after startup and before performing any work. This option is used by JavaClass to report back to the Supervisor process. The default is 10000.
Log4jIntervalMillis	How often, in milliseconds, the Log4J resource monitor thread should perform its work. This option is used to monitor log4j.xml file deployed under temp\batcher working directory and reload it when a change is detected. The default is 1000.
Log4jStartDelayMillis	How long, in milliseconds, the Log4J resource monitor thread should wait after startup and before performing any work. This option is used to monitor log4j.xml file deployed under temp\batcher working directory and reload it when a change is detected. The default is 10000.

Here is an example:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/jre/bin/docfactory_batcher
JavaClass	oracle.documaker.process.ProcessShell
JVMOptions	-Xmx128m -Duser.name=oracle
Instances	1
UseLoadBalancing	No
WorkerClass	oracle.documaker.batch.Batcher
WorkerThreads	4
WorkerIntervalMillis	1000
WorkerStartDelayMillis	5000
ShutdownHookClass	oracle.documaker.batch.shutdown.BatcherShutdownHook
HouseKeeperClass	oracle.documaker.batch.housekeeping.BatcherHouseKeeper
HouseKeeperIntervalMillis	3000
HouseKeeperStartDelayMillis	10000
IPCIntervalMillis	1000
IPCStartDelayMillis	10000

InstallationLocation = The installation location where you installed Document Factory.

Option	Value
Log4jIntervalMillis	5000
Log4jStartDelayMillis	10000

InstallationLocation = The installation location where you installed Document Factory.

Log4J configuration options

For specific information on the Log4J configuration options, see *Defining Log4J Configuration Options* on page 277.

CONFIGURING BATCHTRANSACTIONS THREADS

The BatchTransactions thread reads configuration information from deploy.properties file and ALCONFIGCONTEXT, APPCONFIGCONTEXT, and BCHINGS tables.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Presenter-Ready	This is the status code that lets the BatchTransactions thread know a transaction is ready for processing. The default is 411.
Batcher-InProgress	This is the status code that lets other Document Factory threads/processes know a transaction is being processed by a BatchTransactions thread. The default is 415.
Presenter-Error	This is the status code that indicates a transaction had an error. The default is 441.

Here is an example:

Option	Value
Presenter-Ready	411
Batcher-InProgress	415
Presenter-Error	441

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *BatchTransactions*:

Option	Description
IntervalMillis	How often, in milliseconds, the BatchTransactions thread should perform its work.

Option	Description
StartDelayMillis	How long, in milliseconds, the BatchTransactions thread should wait after startup and before performing any work.
FetchSize	How many records to query at one time from the TRNS table. The default is 5.

Here is an example:

Option	Value
IntervalMillis	1000
StartDelayMillis	10000
FetchSize	5

BCHINGS Table

These options are read from BCHINGS table where the BCHINGS.BATCHNAME and BCHINGS.BATCHTYPE column values match the values provided by an RCPS record via the RCPS.BATCHNAME and RCPS.BATCHTYPE columns.

The BatchTransactions thread retrieves TRNS records that have a status of *Presenter-Ready*. The thread then retrieves the matching RCPS records using the TRN_ID column value from a TRNS record.

Next, the Batcher matches those RCPS records to BCHINGS records for additional configuration options before creating the BCHS and BCHS_RCPS records.

Here are the configuration options:

Option	Description
BATCHNAME	The batch name associated with the RCPS.BATCHNAME column. Used in the lookup of BCHINGS records. This value is also passed to the new batch record via BCHS.BCHBATCHNAME column.
BCHINGNAME	The value that is used for BCHS.BCHNAME column when the new BCHS record is created.
BCHINGTYPE	The batch type. Acceptable values are: 0=Immediate Batch (default), 1=Scheduled Batch. This value is passed to the new batch record via BCHS.BCHTYPE column.
BCHINGPRTTYPE	The output type for a batch (PDF, PS, AFP, XER, TXT, HTML, EPT (HTML email), VPP, and so on). This value is passed to the new batch record via BCHS.BCHPRTTYPE column.
BCHINGACTIVE	A boolean value that indicates if a BCHINGS record is active. Inactive records are skipped in the lookup. Acceptable values are: 0=inactive, 1=active.

* = Indicates this option is used to calculate the start time for a scheduled batch when BCHINGTYPE=1. The closest next starting time is selected when multiple start times can be generated based on the options selected.

Option	Description
BCHINGARCHIVE	A boolean value that indicates if a batch should be archived. Acceptable values are: 0=don't archive, 1=archive. Batches that have a value of 1 are sent to the Archiver process. This value is passed to the new batch record via BCHS.BCHARCHIVE column.
BCHINGENABLENTF	A boolean value that indicates if a batch can have Notification ability (SMS or EMAIL). Acceptable values are: 0=Do not notify, 1=Notify. Batches that have a value of one (1) are sent to the PubNofifier process. This value is passed to the new batch record via the BCHS.BCHENABLENTF column.
BCHINGDOY *	The day of year for a scheduled batch. Null or zero (0) means not used.
BCHINGMOY *	The month of year for a scheduled batch. Null or zero (0) means not used. Normally used with BCHINGDOM option.
BCHINGDOM *	The day of month for a scheduled batch. Null or zero (0) means not used. Normally used with BCHINGMOY option.
BCHINGSTARTYEAR *	The starting year for a scheduled batch. Null or zero (0) means not used. Normally used with BCHINGSTARTMONTH and BCHINGSTARTDAY options.
BCHINGSTARTMONTH *	The starting month for a scheduled batch. Null or zero (0) means not used. Normally used with BCHINGSTARTYEAR and BCHINGSTARTDAY options.
BCHINGSTARTDAY *	The starting day for a scheduled batch. Null or zero (0) means not used. Normally used with BCHINGSTARTYEAR and BCHINGSTARTMONTH options.
BCHINGSTARTHOURS *	The starting hours for a scheduled batch,. Null or zero (0) means not used. Normally used with BCHINGSTARTDAY, BCHINGSTARTMINUTES, and BCHINGSTARTSECONDS options.
BCHINGSTARTMINUTES *	The starting minutes for a scheduled batch. Null or zero (0) means not used. Normally used with BCHINGSTARTDAY, BCHINGSTARTHOURS, and BCHINGSTARTSECONDS options.
BCHINGSTARTSECONDS *	The starting seconds for a scheduled batch. Null or zero (0) means not used. Normally used with BCHINGSTARTDAY, BCHINGSTARTMINUTES, and BCHINGSTARTHOURS options.
BCHINGMON *	Indicates a scheduled batch should start on Monday. Null or zero (0) means not used.
BCHINGTUE *	Indicates a scheduled batch should start on Tuesday. Null or zero (0) means not used.
BCHINGWEN *	Indicates a scheduled batch should start on Wednesday. Null or zero (0) means not used.
BCHINGTHUR *	Indicates a scheduled batch should start on Thursday,. Null or zero (0) means not used.

* = Indicates this option is used to calculate the start time for a scheduled batch when BCHINGTYPE=1. The closest next starting time is selected when multiple start times can be generated based on the options selected.

Option	Description
BCHINGFRI *	Indicates a scheduled batch should start on Friday. Null or zero (0) means not used.
BCHINGSAT *	Indicates a scheduled batch should start on Saturday. Null or zero (0) means not used.
BCHINGSUN *	Indicates a scheduled batch should start on Sunday. Null or zero (0) means not used.
BCHINGSELECTRULE	Additional selection criteria for batch. Added as part of a WHERE clause on the RCPS table. Column names need to be prefixed with table names. The value for this option is added to the WHERE clause by using keyword AND, followed by (x), where x is the value for this option. This value is also passed to the new batch record via BCHS.BCHSELECTRULE column.
BCHINGPRTTYPERULE	Additional selection criteria of the PRTTYPE for a batch record. This option is not used by the Batcher. It is passed to the new batch record via BCHS.BCHPRTTYPERULE column so it can be used by other processes.
BCHINGSORTRULE	Additional sort criteria to sequence the BCHS_RCPS records the Batcher creates. This value is also passed to the new batch record via BCHS.BCHSORTRULE column.
BATCHTYPE	The batch type associated with the RCPS.BATCHTYPE column. Used during lookup of BCHINGS records. This value is also passed to the new batch record via BCHS.BCHBATCHTYPE column.
CALLBACK	A print callback function for the batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.CALLBACK column so it can be used by other processes.
BATCHBANNERBEGINSCRIPT	A batch banner begin DAL script to run upon printing a batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.BATCHBANNERBEGINSCRIPT column so it can be used by other processes.
BATCHBANNERENDSCRIPT	A batch banner end DAL script to run upon printing a batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.BATCHBANNERENDSCRIPT column so it can be used by other processes.
BATCHBANNERBEGINFORM	A batch banner start form to use for printing a batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.BATCHBANNERBEGINFORM column so it can be used by other processes.
BATCHBANNERENDFORM	A batch banner start end to use for printing a batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.BATCHBANNERENDFORM column so it can be used by other processes.
TRANSBANNERBEGINSCRIPT	A batch transaction banner begin DAL script to run upon printing a batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.TRANSBANNERBEGINSCRIPT column so it can be used by other processes.

* = Indicates this option is used to calculate the start time for a scheduled batch when BCHINGTYPE=1. The closest next starting time is selected when multiple start times can be generated based on the options selected.

Option	Description
TRANSBANNERENDSCRIPT	A batch transaction banner end DAL script to run upon printing a batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.TRANSBANNERENDSCRIPT column so it can be used by other processes.
TRANSBANNERBEGINFORM	A batch transaction banner start form to use upon a printing a batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.TRANSBANNERBEGINFORM column so it can be used by other processes.
TRANSBANNERENDFORM	A batch transaction banner end form to use upon a printing a batch. This option is not used by the Batcher. It is passed to the new batch record via BCHS.TRANSBANNERENDFORM column so it can be used by other processes.
BCHINGNTFRULE	Notification rule to run when the BCHINGENABLENTF=1 is set by the Publisher for notifications. This option is not used by the Batcher. It is passed to the new batch record via BCHS.BCHNTFRULE column so it can be used by other processes.
BCHINGLANGRULE	Rule to be run by the Publisher for notification to determine the language of the notification when one is selected. This option is not used by the Batcher. It is passed to the new batch record via BCHS.BCHLANGRULE column so it can be used by other processes.
BCHINGPUBLISH	A boolean value that indicates if a batch should be published. Acceptable values are 0=do not publish, 1=publish. A value of one (1) means a batch is sent to the Publisher process. This value is passed to the new batch record via BCHS.BCHPUBLISH column.
BCHINGMIMETYPE	The MIME type indicator for the batch that gets propagated to the Pubs table rows to define the MIME type of the print spool or other output stored in the row. Normally used for printing by the output Publisher process for print device routing. This option is not used by the Batcher. It is passed to the new batch record via BCHS.BCHMIMETYPE column so it can be used by other processes.
BCHINGBREAKTYPE	An indicator of how to break a batch. Acceptable values are: 0 or null = None 1 = by sheet count 2 = by rcp count 3 = by page count 4 = by transaction count 5 = by custom script This option is not used by the Batcher. It is passed to the new batch record via BCHS.BCHBREAKTYPE column so it can be used by other processes.

* = Indicates this option is used to calculate the start time for a scheduled batch when BCHINGTYPE=1. The closest next starting time is selected when multiple start times can be generated based on the options selected.

Option	Description
BCHINGBREAKVALUE	And indicator that breaks a batch to new Pubs table row. Acceptable values are: 0 or null = not applicable 1 = Use this sheet count static value, GVM variable, or DAL value to compare against the running sheet count to break the batch 2 = Use this rcp count static value,GVM variable, or DAL value to compare against the running rcp count to break the batch 3 = Use this page count static value, GVM variable, or DAL value to compare against the running page count to break the batch 4 = Use this transaction count static value, GVM variable, or DAL value to compare against the running transaction count to break the batch transaction count value to break the batch 5 = Use a script and when the return value is True, break the batch This option is not used by the Batcher. It is passed to the new batch record via the BCHS.BCHBREAKVALUE column so it can be used by other processes.

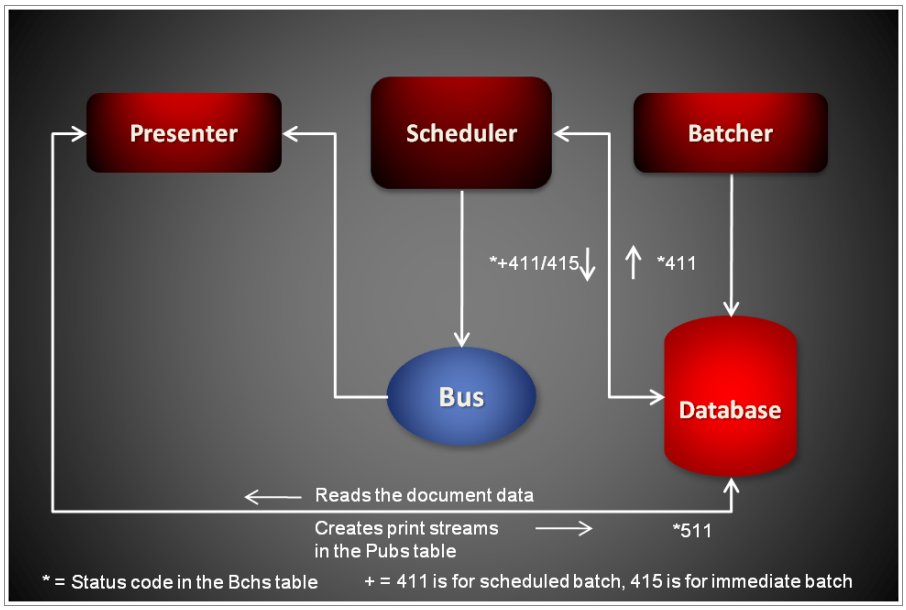
***** = Indicates this option is used to calculate the start time for a scheduled batch when BCHINGTYPE=1. The closest next starting time is selected when multiple start times can be generated based on the options selected.

CONFIGURING THE PRESENTER

The Presenter process generates print streams for the Document Factory. It is deployed and managed by the Supervisor process. The Presenter monitors an input queue and waits for notification messages from the Scheduler process that there are transactions ready for processing.

Once a notification message is received, the Presenter retrieves the document data from records in the BCHS, BCHS_RCPS, RCPS and TRNS tables and creates one or more print streams to the Pubs table.

The Presenter process typically runs after the Distributor and Batcher processes and reads input from RCPS records generated by the Distributor and BCHS and BCHS_RCPS records created by the Batcher. The Presenter also reads the document data from the TRNS table to create the print streams.

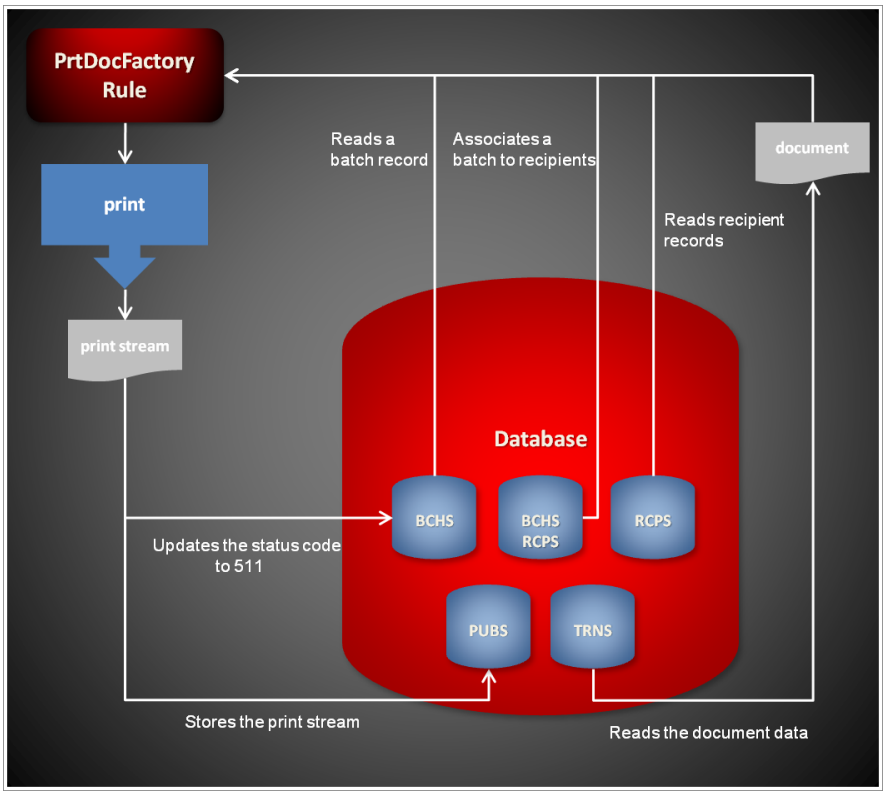


Each notification message received by the Presenter provides the batch ID for a record in BCHS table that needs presenting. Here is an example of a message:

```
<?xml version="1.0" encoding="UTF-8"?>
<BatchTicket
    xmlns="oracle/documaker/schema/tables/bchs"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<BCH_ID>101</BCH_ID>
</BatchTicket>
```

USING THE PRTDOCFACTORY RULE

The Presenter uses the PrtDocFactory rule to perform basic transaction processing and housekeeping. It retrieves a record from the BCHS table, cross-references the record BCH_ID in the BCHS_RCPS table to match the RCPS records to the BCHS record, and then retrieves the matching records from the RCPS table. The Presenter then retrieves the document data from the TRNS table and generates one or more prints streams, depending on the split options, for the RCPS records. The rule then stores the print streams in the Pubs table and updates the status code for the transaction in the BCHS record so the Scheduler process can notify the next process in the assembly line.



BATCH PROCESSING LOGIC

Each execution of the Presenter is for a specific batch ID (BCH_ID). The batch record details the processing options for the batch. These options would include:

- Immediate or batch print
- Output type (PrtType)
- Batch split criteria
- Callback function.

The batch table record has a one-to-many relationship with the batch recipients table (BCHS_RCPS). The batch recipients table provides a list of the recipient table records (RCPS) to be included in the batch processing.

For each record in the batch recipients table with a corresponding BCH_ID the Presenter...

- Loads the recipient record
- Loads the recipient's transaction record
- Loads the transaction's form set
- Creates the recipient's copy of the output in the format specified by the batch's PrtType column

SPLIT OPTIONS

Optionally, the output batches for batch print can be split in to logical output batches in the Pubs table based on following criteria:

- DAL script
- Sheet count
- Page count
- Recipient record count
- Unique transaction count

For the DAL Script option, the DAL script is called at the conclusion of processing for each recipient record. If the DAL script returns value greater that zero the batch is split. For the count options, the batch is split when the desired count exceeds the break value. The break value can be specified as a number or as a supported tilde (~) function. If a tilde function is used, the function must return a numeric value. For immediate batches, the output is always split by recipient.

Here is an overview of what the PrtDocFactory rule does:

Initialization	Loads the transaction status.
PreProc	• Reads a batch table record. • Validates the batch status. • Updates the batch status to *Presenter Start*. • Initializes the print environment. • Installs the callback functions. • Creates the page and sheet count GMVs.
PostProc	• Sets the batch break condition. • Installs the batch callback (if specified). • Initializes the RCPS, RCPS_BCHS and Pubs tables. • Sets the print output function. • Initializes the printer type. • Opens the print stream. • Executes the Batch Begin Banner Function.

For each record in the BCHS_RCPS table, the system:

- Looks up the corresponding recipient record (RCPS).

- Queues up the print recipient.
- Looks up the corresponding transaction record (TRNS).
- Loads the form set data from the transaction record.
- Executes the Transaction Banner Begin function.
- Prints the form set.
- Executes the Transaction Banner End function.
- Splits all immediate print batches by recipient.

For non-immediate print, the batches can be split using these options:

- DAL script
- Sheet count
- Page count
- Recipient record count
- Unique transaction count
- Tilde (~) function (to specify break count).

If the batch is to be split, the system...

- Executes the Batch Banner End function.
- Closes the stream to the current Pubs table record.
- Opens the new stream Pubs table record.
- Executes the Batch Banner Begin function.

After all recipient records have been processed, the system...

- Executes the Batch Banner end function.
- Closes the stream to the final Pubs table record.
- Updates the batch record status to *Presenter End*.
- Terminates the print environment.

Supported Output Types

These output types are supported:

Processing option	Output type
Immediate	

*=PXL, PST, and XER require the disk print option (See INI options - DocFactoryDiskPrint).

Processing option	Output type
	PDF (Portable Document Format) PCL (Printer Command Language) PXL (Printer Command Language-6)* XMP (XML output) RTF (Rich Text Format) MPM (Inline HTML) PST (PostScript)*
Scheduled	
	PCL (Printer Command Language) PXL (Printer Command Language-6)* XMP (XML output) MPM (Inline HTML) AFP (IBM Advanced Function Printing) XER (Xerox Metacode)* VPP (Versatile Printing and Plotting) PST (PostScript)*

*=PXL, PST, and XER require the disk print option (See INI options - DocFactoryDiskPrint).

Starting and Stopping the Presenter

To	Then
Verify the Presenter is running.	Verify there is a running process with the name docfactory_presenter.
Start the Presenter	Place the presenter.jar file in the deploy directory of Document Factory.
Stop the Presenter	Remove the presenter.jar file from the deploy directory of Document Factory.

Note The presenter.jar configuration file is uncompressed and deployed to the temp\presenter directory. This directory becomes the working directory for the Presenter. All output, including Log4J output, uses this directory as the starting directory.

Using Presenter Configuration Resources

The configuration information for the Presenter is stored in these resources:

Resource	Contains the
presenter.jar file	Minimal startup configuration information.
.bindings file	Java Naming and Directory Interface (JNDI) data sources.
APPCONFIGCONTEXT table	Configuration options.
ALCONFIGCONTEXT table	Configuration options for the Presenter status codes and message bus.
fsiuser_3.ini file	INI options specific to the Presenter process.
fsisys.ini file	INI options that are common to the Assembler, Distributor, and Presenter processes.
afgjob_3.jdt file	Documaker rules run by the Presenter process.

presenter.jar

The presenter.jar file is located in the \deploy subdirectory of the Document Factory. It contains these configuration resources:

Component	Description
deploy.properties	Contains the minimal startup configuration information.
log4j.xml	Used to capture Log4J diagnostic and error output during start up. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j/
log4j.dtd	Used by the log4j.xml file.

deploy.properties File

The deploy.properties file is extracted and placed in the temp\presenter working directory. This file contains the minimal startup configuration options used to read the configuration for the Presenter from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables:

Option	Description
system.id	The value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Presenter configuration.
assemblyline.id	The value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Presenter configuration.
application.id	The value of APP_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Presenter configuration.

Option	Description
config	The configuration name for the Presenter. The default is Presenter. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Presenter configuration.
config.jndi.name	The Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.
config.schema	The database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.
factory.jndi.name	The JNDI name for the data source that contains the assembly line tables.
factory.schema	The database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=7
config=Presenter
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

log4j.xml File

The log4j.xml file is extracted and placed in the temp/presenter working directory. The log4j.xml file contains loggers used during start up of the Presenter, prior to the Presenter loading the Log4J configuration from the APPCONFIGCONTEXT table. See the Log4J configuration options in the *APPCONFIGCONTEXT Table* on page 217 for more information.

.bindings File

The .bindings file is located in the config\context subdirectory of the Document Factory. It contains the Java Naming and Directory Interface (JNDI) data sources used by the Presenter. Each JNDI data source contains these configuration options:

Option	Description
ClassName	The data source fully-qualified class name. Use the javax.sql.DataSource value.
FactoryName	The data source factory fully-qualified class name. Use the org.apache.commons.dbcp.BasicDataSourceFactory value. The BasicDataSourceFactory class supports connection pooling.
driverClassName	The Java Database Connectivity (JDBC) driver class name.
url	The JDBC URL.

Option	Description
maxOpenPreparedStatements	The maximum number of prepared statements to cache in the connection pool. Use the value -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	How often the idle object evictor thread should run and perform clean up of the stale connection handles. Use the value -1 to disable the idle object evictor thread.
validationQuery	A validation query that should be run when borrowing objects from the connection pool.
username	The JDBC user name.
password	The JDBC password.
testOnBorrow	Set this option to Yes if validationQuery should be used when borrowing an object from the connection pool. The default is No.
initialSize	The initial connection pool size.
maxActive	The maximum number of active connections in the pool.
maxIdle	The maximum number of idle connections in the pool.
minIdle	The minimum number of idle connections in the pool.
maxWait	The maximum time (in milliseconds) to wait for a connection object to be retrieved from the pool before issuing an error.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
```

```
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type= maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
DMKRConfig/RefAddr/9/Type=maxWait
#Unix friendly Documaker Doc. Factory JNDI DataSource
DMKRFactory/ClassName=javax.sql.DataSource
DMKRFactory/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRFactory/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRFactory/RefAddr/0/Encoding=String
DMKRFactory/RefAddr/0/Type=driverClassName
DMKRFactory/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRFactory/RefAddr/1/Encoding=String
DMKRFactory/RefAddr/1/Type=url
DMKRFactory/RefAddr/10/Content=-1
DMKRFactory/RefAddr/10/Encoding=String
DMKRFactory/RefAddr/10/Type=maxOpenPreparedStatements
DMKRFactory/RefAddr/11/Content=-1
DMKRFactory/RefAddr/11/Encoding=String
DMKRFactory/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRFactory/RefAddr/12/Content=select 1 from dual
DMKRFactory/RefAddr/12/Encoding=String
DMKRFactory/RefAddr/12/Type=validationQuery
DMKRFactory/RefAddr/2/Content=dmkr_asline
DMKRFactory/RefAddr/2/Encoding=String
DMKRFactory/RefAddr/2/Type=username
DMKRFactory/RefAddr/3/Content=oracle12
DMKRFactory/RefAddr/3/Encoding=String
DMKRFactory/RefAddr/3/Type=password
DMKRFactory/RefAddr/4/Content=true
DMKRFactory/RefAddr/4/Encoding=String
DMKRFactory/RefAddr/4/Type=testOnBorrow
DMKRFactory/RefAddr/5/Content=1
DMKRFactory/RefAddr/5/Encoding=String
DMKRFactory/RefAddr/5/Type=initialSize
DMKRFactory/RefAddr/6/Content=8
DMKRFactory/RefAddr/6/Encoding=String
DMKRFactory/RefAddr/6/Type= maxActive
DMKRFactory/RefAddr/7/Content=8
DMKRFactory/RefAddr/7/Encoding=String
DMKRFactory/RefAddr/7/Type=maxIdle
DMKRFactory/RefAddr/8/Content=0
DMKRFactory/RefAddr/8/Encoding=String
DMKRFactory/RefAddr/8/Type=minIdle
DMKRFactory/RefAddr/9/Content=60000
DMKRFactory/RefAddr/9/Encoding=String
DMKRFactory/RefAddr/9/Type=maxWait
```

APPCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Presenter*:

Option	Description
StartCommand	Defines the command to use to start the Presenter. Include the full path.
StartArguments	Defines the initialization arguments used to start the Presenter. Here is an example: /ini=fsiuser_3.ini /debug=0 /phase=3
env.mode.*	The environment variables the process expects to run. The Supervisor creates an environment variable for each env.mode.* configuration option it encounters. The naming convention is `env.mode.name` Where *mode* can be either zero (0), meaning prepend, one (1), meaning append, or two (2), meaning overwrite, and *name* is the name of the environment variable. When the mode is not defined, the default is two (2). An example of an env.mode.* variable would be env.0.PATH or env.ORACLE_HOME. Notice the second example uses the default overwrite mode.
StartDirectory	Defines the start up directory. Here is an example: c:/oracle/oracle_insurance_1/documaker/mstrres/dmres
Instances	(Optional) The number of instances the Supervisor should start for a process configuration. The default is two (2).
UseLoadBalancing	(Optional) This option controls wether the Supervisor checks the idle time of a process's instances that are running and starts additional ones when all of them are busy. Instances are considered busy when their idle time is less than the value provided in the MinIdleTimeSeconds option. The Supervisor uses the value provided in the IdleTimeChecks option to determine the number of idle time checks to run before it starts additional instances. When additional instances are started for load balancing purposes, they are shut down by the Supervisor if their idle time exceeds the value in the MaxIdleTimeSeconds option. The maximum number of instances running is the value for the MaxInstances option (including the instances configured in the Instances option). The Supervisor checks the idle time of the current instances at the interval specified in the IdleTimeCheckIntervalSeconds and if all are busy, it starts an additional number of instances equal to the value provided in the IncrementCount option. Please note that the Supervisor does not start checking the busy time of the current instances until the time provided in the IdleTimeCheckDelaySeconds option elapses. Make sure the value for the delay is ample enough to provide for all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. You can enter Yes or No. The default is Yes.
MaxInstances	(Optional) This option controls the maximum number of instances that can run when the UseLoadBalancing option is enabled. The default is the number of processors times two.
IncrementCount	(Optional) This option controls how many additional instances are started during the current check when all instances running are busy and the UseLoadBalancing option is enabled. The default is two (2).
IdleTimeCheckIntervalSeconds	(Optional) This option controls how often the Supervisor checks the idle time of the instances that are running to determine if they are busy so it can start additional ones when the UseLoadBalancing option is enabled. The default is 10 seconds.

Option	Description
IdleTimeCheckDelaySeconds	(Optional) This option controls the initial delay before the first idle time check is performed by the Supervisor when the UseLoadBalancing option is enabled. This time should be ample enough to allow all instances to start and reach an idle time equal to or greater than the value provided for the MinIdleTimeSeconds option. The default is 120 seconds.
IdleTimeChecks	(Optional) This option defines the number of consecutive idle time checks that must fail, meaning all instances were busy during each check, before more instances are started when the UseLoadBalancing option is enabled. Each check takes place at the IdleTimeCheckIntervalSeconds interval. The default is 12.
MinIdleTimeSeconds	(Optional) This option controls the minimum idle time for each instance. The idle time represents how long it has been since an instance processed the last request. If the Supervisor detects an instance has an idle time less than the value provided for this option, it considers it busy for the purpose of load balancing. The default is 5 seconds.
MaxIdleTimeSeconds	(Optional) This option controls the maximum idle time for an additional instance. The idle time represents how long it has been since an instance performed processing. If the Supervisor detects an instance, which was started for the purpose of load balancing, has reached an idle time greater than the value provided for this option, it sends the instance a shutdown request. The default is 120 seconds.
MaxTransactions	(Optional) This option controls the maximum number of transactions an instance can process before it is restarted by the Supervisor. The default is -1, which disables this option.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before it is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
OrderedRestartIntervalSeconds	(Optional) This option controls the interval used for restarting each process instance in a sequential/ordered manner when the MaxTransactions or MaxUpTime options are used. The Supervisor restarts one instance at a time and waits for an amount of time equal to the value specified for this option before it restarts the next one and so on until it has restarted all of them. If you set this option to less than 60 seconds, you can negatively affect performance. The default is 60 seconds.
WatchList	A comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted. Here is an example: c:/oracle/oracle_insurance_1/documaker/mstrres/dmres/fsiuser_3.ini,c:/oracle/oracle_insurance_1/documaker/mstrres/dmres/fsisys.ini
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is 5.

Here is an example:

Option	Value
StartCommand	/oracle_home/*InstallationLocation*/bin/docfactory_presenter
StartArguments	/ini=fsiuser_3.ini /debug=0 /phase=3
env.0.PATH	/oracle_home/*InstallationLocation*/oracle_instantclient_11_2,/oracle_home/*InstallationLocation*/jre/bin,/oracle_home/*InstallationLocation*/jre/bin/client,/oracle_home/*InstallationLocation*/bin
env.ORACLE_HOME	/oracle_home/*InstallationLocation*/bin
env.NLS_LANG	AMERICAN_AMERICA.AL32UTF8
env.TNS_ADMIN	/oracle_home/*InstallationLocation*/oracle_instantclient_11_2/NETWORK/ADMIN
env.JVM_OPTIONS	-Xmx256m,-Duser.name=oracle,-Dlog4j.configuration=/oracle_home/*InstallationLocation*/docfactory/temp/presenter/log4j.xml,-Dlog4j.file=/oracle_home/*InstallationLocation*/docfactory/temp/presenter/logs/log4j.log,-Djndi.context=/oracle_home/*InstallationLocation*/docfactory/config/context,-Dfactory.jndi.name=DMKRFactory,-Dconfig.jndi.name=DMKRConfig,-Dschema=DMKR_ASLINE
StartDirectory	/oracle_home/*InstallationLocation*/mstrres/correspondence
Instances	2
UseLoadBalancing	No
MaxInstances	8
IncrementCount	1
IdleTimeCheckIntervalSeconds	15
IdleTimeCheckDelaySeconds	240
IdleTimeChecks	5
MinIdleTimeSeconds	5
MaxIdleTimeSeconds	120
MaxTransactions	-1
MaxReportIntervalSeconds	180
MaxUpTimeSeconds	-1
WaitForShutdownSeconds	60
OrderedRestartIntervalSeconds	60
WatchList	/oracle_home/*InstallationLocation*/mstrres/dmres//fsiuser_3.ini,/oracle_home/*InstallationLocation*/mstrres//dmres//fsisys.ini
MaxRestarts	5

InstallationLocation = The installation location where you installed Document Factory.

Log4J configuration options

For specific information on the Log4J configuration options, see *Defining Log4J Configuration Options* on page 277.

ALCONFIGCONTEXT Table

These options are read from this table when the GROUP_NAME column value is *Status*:

Option	Description
Presenter-Ready	This is the status code that indicates a transaction is ready to be sent to the Presenter. The default is 411.
Presenter-ACK	This is the status code that indicates a transaction has been received and it is being processed by Presenter. The default is 431.
Presenter-Error	This is the status code that indicates the Presenter process failed to process a transaction. The default is 441.

Here is an example:

Option	Value
Presenter-Ready	411
Presenter-ACK	431
Presenter-Error	441

These options are read from the ALCONFIGCONTEXT table when the GROUP_NAME column value is *Bus*:

Option	Description
PresenterQueue	The name of the queue the Presenter uses to receive notifications from the Scheduler process.
*	Any other configuration options expected by the message bus.

Note Document Factory uses the same message bus java packages as Docupresentment, so it supports the same message bus configuration options as Docupresentment. See the Internet Document Server Guide for more information on message bus configuration options supported for MQ, MSMQ, and JMS.

Here is an example:

Option	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	weblogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001

Option	Value
jms.qcf.name	jms/qcf
PresenterQueue	jms/presenter_requestq
TimeoutSeconds	5

FSIUSER_3.INI File

This file can be found in the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section. It provides INI options required to run the Presenter process under the Document Factory.

BCHS table definition

These options are read from the DBTable:BCHS INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:BCHS >
    DBHandler = WIP_ODBC_ORA
    UniqueTag = BCH_ID
```

BCHS/RCPS table definition

This option is read from the DBTable:RCPS_RCPS INI control group:

Option	Description
DBHandler	The name of the database handler.

Here is an example:

```
< DBTable:BCHS_RCPS >
    DBHandler = WIP_ODBC_ORA
```

Pubs table definition

These options are read from the DBTable:Pubs INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:Pubs >
    DBHandler = WIP_ODBC_ORA
    UniqueTag = PUBUNIQUE_ID
```

PubsInfo table definition

These options are read from the DBTable:PubsInfo INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:PubsInfo >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = PUBUNIQUE_ID
```

RCPS table definition

These options are read from the DBTable:RCPS INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:RCPS >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = RCP_ID
```

WIP Index table definition

These options are read from the DBTable:WIP INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:WIP >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = FORMSETID
```

WIP Data table definition

These options are read from the DBTable:WIPData INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:WIPData >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = FORMSETID
```

Extract table definition

These options are read from the DBTable:EXTR INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:EXTR >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = TRN_ID
```

Recipients Print table definition

These options are read from the DBTable:RCBSPRT INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:RCBSPRT >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = RCP_ID
```

Jobs table definition

These options are read from the DBTable:JOBS INI control group:

Option	Description
DBHandler	The name of the database handler.
UniqueTag	The unique tag column name.

Here is an example:

```
< DBTable:JOBS >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = JOBUNIQUE_ID
```

DFD Definitions

These options are read from the WIPData INI control group:

Option	Description
DatabaseWIP	Set this option to Yes if you want to store WIP in a database. The default is No.

Option	Description
File	The internal name of the WIP table.
WIPDFDFile	The name of the WIP index DFD file.
WIPDataDFD	The name of the WIP data DFD file for XML NA/POL.
WIPDsDataDFD	The name of the WIP data DFD file for combined NA/POL.
Jobs	The name of the Jobs table.
JobsDFD	The name of the Jobs index DFD file.
TrnExtrDFD	The name of the extract DFD file.
TrnStatusDFD	The name of the transaction status DFD file.
BCHS	The name of the batch table.
BCHSDFD	The name of the batch DFD file.
BCHS_RCPSDFD	The name of the batch recipients DFD file.
BCHS_RCPS	The name of the batch recipients association table.
Pubs	The name of the publishing table.
PubsDFD	The name of the publishing DFD file.
PubsInfoDFD	The name of the publishing information DFD file.
PubsInfo	The name of the publishing information table.
RcbDfdFile	The name of the recipients DFD file.

Here is an example:

```
< WIPData >
    DatabaseWIP = Yes
    File = WIP
    WIPDFDFile = .\deflib\trnsdf.dfd
    WIPDataDFD = .\deflib\docdata.dfd
    WIPDsDataDFD = .\deflib\dsdata.dfd
    TRNEXTRDFD = .\deflib\trnsextr.dfd
    TRNSTATUSDFD = .\deflib\TRNSTATUS.dfd
    JOBS = JOBS
    JOBS = .\deflib\JOBS.dfd
    BCHS = BCHS
    BCHSDFD = .\deflib\BCHS.dfd
    BCHS_RCPSDFD = .\deflib\BCHS_RCPS.dfd
    BCHS_RCPS = BCHS_RCPS
    Pubs = Pubs
    PubsDFD = Pubs.dfd
    PubsInfoDFD = PubsInfo.dfd
    PubsInfo = PubsInfo
    RcbDfdFile = .\deflib\rcbdocf.dfd
```

Document Factory options

These options are read from the DocFactory INI control group:

Option	Description
Presenter_Start	The Presenter start status code. The default is 421.
Presenter_Processing	The Presenter ACK status code. The default is 431.
Presenter_Error	The Presenter error status code. The default is 441.
Presenter_End	The Presenter end status code. The default is 511.
Bindings	The path location for the Java Naming and Directory Interface (JNDI) .bindings file containing the data source information for JNI code. The default is /docfactory/config/context/.

Here is an example:

```
< DocFactory >
   Presenter_Start = 421
   Presenter_Processing = 431
   Presenter_Error = 441
   Presenter_End = 511
   Bindings = /oracle_home/InstallationLocation/docfactory/config/
context
```

Logging messages to the database

This option is read from the Environment INI control group:

Option	Description
JLOG_Enabled	Set this option to Yes to redirect warning and error messages to the LOGS and ERRS tables instead of being written to the trace file. The default is No.

Here is an example:

```
< Environment >
   JLOG_Enabled = Yes
```

Controlling log output

These options are read from the DocFactory_Presenter:JLog INI control group:

Option	Description
LogLogger	The name of the Log4J logger used to log warning messages to the LOGS table. This name should match the Log4J logger name in log4j.xml file.
ErrorLogger	The name of the Log4J logger used to log error messages to the ERRS table. This name should match the Log4J logger name in log4j.xml file.
ColumnNames	A comma-delimited list of table column names to GVM mappings. Is used by the loggers to capture the GVM values and set them as the column values. The format for each comma-delimited token can be ColumnName=GVMName or just ColumnName.
BufferSize	The maximum buffer size for messages. This value should match the length of the LOGMESSAGE and ERRMESSAGE columns.
Debug	Set this option to Yes if you want diagnostic output generated for the Logger. The default is No.

Option	Description
LogError	Set this option to No if you want the system to suppress all error messages. The default is Yes, which tells the system to issue error messages.
LogWarning	Set this option to Yes if you want the system to issue warning messages. The default is No, which suppresses all warning messages.

Here is an example:

```
< DocFactory_Presenter:JLog >
    LogLogger = LogLogger
    ErrorLogger = ErrorLogger
    BufferSize = 2000
    Debug = No
    LogError = Yes
    LogWarning = No
    ColumnNames = JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,
BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_ID
```

FSISYS.INI File

This file provides INI options required to run the Presenter process under the Document Factory. You can find this file in the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section.

Enabling Document Factory code

This option is read from the RunMode INI control group:

Option	Description
DocFactory	Must be set to Yes if you are using Document Factory. To facilitate legacy Documaker Server processing, this option defaults to No.

Here is an example:

```
< RunMode >
    DocFactory = Yes
```

AFGJOB_3.JDT File

This file provides the Documaker Server rules to run for the Presenter process under the Document Factory. You can find this file in the \deflib subdirectory under the path provided for the StartDirectory configuration option in the APPCONFIGCONTEXT configuration section. Here is an example:

```
/*  JDT Rules for Single-Step Processing Batching By Recipient. */
;RULStandardJobProc;1;Always the first job level rule;
/*;RULServerJobProc;1;Always the first job level rule; */
;SetErrHdr;1;*:;
...
;SetErrHdr;1;*:-------------------------------------------------;
;JobInit1;1;;
/* Every form set in this base uses these rules. */
;PrtDocFactory;2;DocFactory Phase 3;
/* Every image in this base uses these rules. */
;WIPImageProc;3;Always the first image level rule;
/* Every field in this base uses these rules. */
```

```
;WIPImageProc;4;Always the first field level rule;
```

CONFIGURING THE ARCHIVER

The Archiver retains copies of your forms and data in the destination you specify so you can later re-create or reuse those forms and data. You can choose from these archive destinations:

- AssureSign
- UCM
- FTP
- FileSystem

For each destination, you set the location and connection information in the Configuration Group properties. From there, set the values for the index information in the Defaults, Static Values, and Mappings groups. For more information about these groups and properties, see *Archiver Properties* on page 230.

Note There is only one configuration property for the FileSystem destination, which defines the location where the documents will be placed. The default location is:

`\oracle\oracle_insurance_1\documaker\filesystem-archive`

The structure within this directory is taken from the BCH_ID and the documents within are named from the PUBUNIQUE_ID and PUBPRTEXT.

Integrating with AssureSign

The Batching configuration options let you direct a set of documents to a given batch which can then be sent to a third-party vendor for an additional workflow. Typically, this workflow includes signature added by a recipient of the document. To enable the AssureSign destination, first enable the Archive process. The AssureSign system expects either PDF or Word documents so be sure the batch is configured with the correct print and MIME type.

Configuration

Within the Archiver configuration, use the AssureSign category to edit the required settings. The Configuration group contains the properties used to initialize the AssureSign destination and communicate with the AssureSign server.

AssureSign supports both a sandbox and a production server for the development and the use of signing activities. The initial, default configuration for the AssureSign server destination uses the sandbox server. You can modify the destination.assuresign.use.sandbox setting on the AssureSign destination configuration if you need to route some documents to the sandbox environment and others, in a different batch, to the production server. To do so, set up a uniquely-named AssureSign destination in the Archiver to point to each server and use the desired location for the AssureSign destination name for the batch.

The property settings in the Defaults group are added to the request before any Mappings settings are added. These are static text values. If you want these properties to be mapped to Document Factory data, delete the row — or just uncheck the Active box — and copy it to the Mappings group.

The Mappings group is where you define how Document Factory data is going to be mapped into destination specific properties.

Notice that the destination.assuresign.template.identifier is defined in the Configuration and in the Mappings groups. This lets you set up a default template the system will use if one is not specified in the Mappings group.

To set up a batch to use the AssureSign archive destination, update the Assembly Line Batching details to activate the AssureSign function and indicate the AssureSign destination configured in the Archiver settings.

ARCHIVER PROPERTIES

Use these groups and properties to configure the Archiver process:

Archiver

Define these properties for the Document Factory Archiver.

Option	Description
Section Name = Archiver	
StartCommand	Defines the command used to start the Archiver.
StartArguments	Defines the initialization arguments used to start the Archiver.
JavaClass	Defines the implementation class. Here is an example: oracle.documaker.process.ProcessShell
JVMOptions	Defines the JVM (Java Virtual Machine) arguments for the archiver. Here is an example: -Xmx256m -Djava.library.path=c:\oracle\documaker\bin
Instances	Defines the number of JVM instances to run.
UseLoadBalancing	Determines whether load balancing is on or off. The default is off (False).
WorkerClass	Defines the archiving class that does the work. Here is an example: oracle.documaker.archiver.Archiver
WorkerThreads	Defines the number of threads per JVM.
WorkerIntervalMillis	Defines, in milliseconds, the work time for each worker.
WorkerStartDelayMillis	Defines, in milliseconds, the ramp up delay for each worker thread.
ShutdownHookClass	Defines the implementation class to call upon shutdown. Here is an example: oracle.documaker.archiver.shutdown.ArchiverShutdownHook
HouseKeeperClass	Defines the implementation class to class for general housekeeping. Here is an example: oracle.documaker.archiver.housekeeping.ArchiverHouseKeeper
HouseKeeperIntervalMillis	Defines, in milliseconds, the housekeeper interval.

Option	Description
HouseKeeperStartDelayMillis	Defines the delay for each housekeeping thread.
Log4jIntervalMillis	Defines, milliseconds, the logging interval.
DocumentURL	Defines the location of the archived document. Here is an example: http://localhost:16200/cs/groups/secure/documents/document/
Log4jStartDelayMillis	Defines, in milliseconds the logging ramp up delay interval.

Archiver-Mapping

Use these properties to define archive mapping settings.

Option	Description
Section Name = Archiver-Mapping	
dDocTitle	Defines the title of the document. Can be a variable.
primaryFile	Defines the name of the primary file. Can be a variable.
primaryFileExt	Defines the extension for the primary file. Can be a variable.

Archiver-Source

Use these properties to define archive source settings.

Option	Description
Section Name = Archiver-Source	
source.administration.name	Defines the Archiver administration implementation class. Here is an example: oracle.documaker.archiver.ArchiverAdministration
source.count	Defines the source count.
source.name	Defines the source implementation name. Here is an example: oracle.documaker.archiver.ArchiverSource
table	Defines the archiver table.

CONFIGURING THE PUBLISHER

You can create your own custom archive destination using information provided in the Documaker Connector Developers Guide. When this class is available, perform these steps to integrate your new destination into Document Factory:

1. Add the jar file for the new destination into the documaker\docfactory\lib directory.
2. Add the destination name to the DMRK_TRANSLAT table in the admin schema for the Archiver application in the system and assembly line where it will be used.
3. Add the destination and its configuration to the Archiver worker using Documaker Administrator. Set up the configuration information, defaults, and mappings needed.
4. Reference the new destination in the batch definition for the Archiver or signing step.

Email Publisher

Use these properties to define the email publisher.

Option	Description
Section Name = EmailPublisher	
Host	Defines the publishing email host.
Sender	When publishing to an email list, this value is used to till the From: field. Here is an example: admin@docfactory.com

Email Servers

Use these properties to define email servers.

Option	Description
Section Name = EmailServers	
SMTPEmailServer	Defines the implementation class for the email system. Here is an example: oracle.documaker.publishing.email.SMTPEmailTransporter

Housekeeper

Define these properties to configure the housekeeping facilities.

Option	Description
Section Name = Housekeeper	
FetchSize	Defines the size to fetch.

Option	Description
TransactionTimeoutMillis	Defines, in milliseconds, the transaction timeout interval.

Notify Publisher Scheduler

Define these properties to configure the Document Factory Notify Publisher Scheduler engine.

Option	Description
Section Name = NotifyPresenterScheduler	
FetchSize	Defines the size to fetch.
IntervalMillis	Defines, in milliseconds, the notification interval.
StartDelayMillis	Defines, in milliseconds the ramp up delay interval.

Publisher Plug-ins

Define these properties to hook into the Document Factory Publisher.

Option	Description
Section Name = PublisherPlugins	
EmailPublisher	Defines the email plug-in for the publishing engine. Here is an example: oracle.documaker.publishing.EmailPublisher
PrinterPublisher	Defines the printer plug-in for the publishing engine. Here is an example: oracle.documaker.publishing.PrinterPublisher

CONFIGURING THE PUBNOTIFIER

Use these properties to configure the Document Factory PubNotifier engine.

Option	Description
Section Name = PubNotifier	
DefaultLanguage	Defines the default language. The default is en (English).
mail.smtp.user	Defines the SMTP (Simple Mail Transfer Protocol) user to whom to send mail.
mail.smtp.host	Defines the SMTP server host.
mail.smtp.port	Defines the SMTP server port.
mail.smtp.password	Defines the SMTP password for the given SMTP user.
mail.from	When sending emails, this is the default to use in the From: field.
DefaultMessageType	Defines the default message type.
UseEmailForSMS	Lets you use email as an SMS (Short Message Service) transport. The default is True.
EmailProvider	Defines the Java email implementation class. Here is an example: oracle.documaker.messaging.JavaMailTransport
EmailProvider-ums	Defines the UMS (Unified Messaging System) implementation class. Here is an example: oracle.documaker.messaging.UMSTransport
EmailProvider-javamail	Defines the Java email implementation class. Here is an example: oracle.documaker.messaging.JavaMailTransport
StartCommand	Defines the command to use to start the Publishing Notifier.
StartArguments	Defines the initialization arguments used to start the Publishing Notifier.
JavaClass	Defines the main implementation class. Here is an example: oracle.documaker.process.ProcessShell
JVMOptions	Defines the JVM (Java Virtual Machine) arguments for the Publishing Notifier. Here is an example: -Xmx256m -Djava.library.path=c:\oracle\documaker\bin
Instances	Defines the number of instances of JVM for this application.
UseLoadBalancing	Determines whether to use load balancing. The default is False.
WorkerClass	Defines the implementation class to class for notification. Here is an example: oracle.documaker.pubnotifier.PubNotifier
WorkerThreads	Defines the number of worker threads per JVM.
WorkerIntervalMillis	Defines, in milliseconds, the worker thread interval.
WorkerStartDelayMillis	Defines, in milliseconds, the worker thread ramp up delay.

Option	Description
UMSUsername	Defines the UMS (Unified Messaging System) user name.
UMSPassword	Defines the UMS password.
UMSEndpoint	Defines the UMS endpoint. Here is an example: http://localhost:8001/sdpmessaging/parlayx/SendMessageService
ShutdownHookClass	Defines the implementation class to call upon shutdown. Here is an example: oracle.documaker.pubnotifier.shutdown.PubNotifierShutdownHook
HouseKeeperClass	Defines the implementation class to use for general housekeeping. Here is an example: oracle.documaker.pubnotifier.housekeeping.PubNotifierHouseKeeper
HouseKeeperIntervalMillis	Defines, in milliseconds, the housekeeper interval.
HouseKeeperStartDelayMillis	Defines, in milliseconds, the housekeeping ramp up delay.
Log4jIntervalMillis	Defines, in milliseconds, the logging interval.
Log4jStartDelayMillis	Defines, in milliseconds, the logging ramp up delay.

SMTP Email Servers

Use these properties to define SMTP email servers.

Option	Description
Section Name = EmailServers	
Host	Defines the location of the SMTP (Simple Mail Transfer Protocol) server.

CONFIGURING THE HISTORIAN

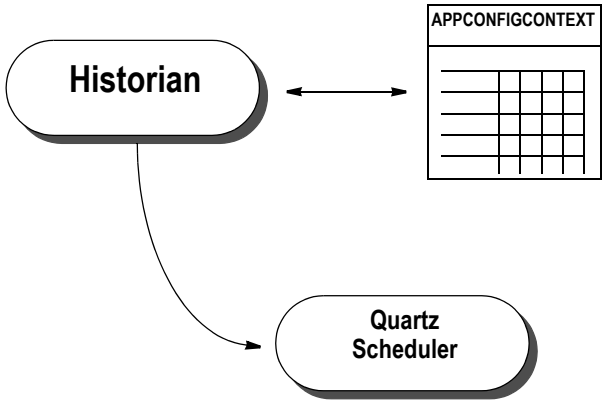
Use the Historian process to maintain the main processing tables of Document Factory as well as manage its historical data tables. The Historian is deployed and managed by the Supervisor process and executes configured tasks based on a schedule you create. For instance, you can configure the Historian to do its processing during off-hours to minimize the effect on the system resources.

This topic contains the following topics:

PROCESSING OVERVIEW

When you start Document Factory, the Historian reads the APPCONFIGCONTEXT table to get the settings it needs to start. This information includes Historian task configurations. A Historian task configuration defines how to process a specific data set, based on a schedule and filtering criteria.

The Historian creates and adds each configured task to an internal task scheduling mechanism (Quartz Scheduler), according to the task's configured schedule. Once the startup routines finish, Historian starts the Quartz Scheduler, which then executes Historian tasks based on the schedules.



Note Quartz Scheduler is an open source job scheduling service from Software AG that can be integrated with or used alongside Java EE or Java SE applications.

The Historian then enters the sleep state, waking periodically as defined by its configuration. During the wake state, the Historian performs no work and all processing is delegated to the Quartz Scheduler.

The Historian process architecture

The Historian process is single-threaded and is responsible for configuring jobs within and starting the Quartz Scheduler. A Historian task is a single-threaded.

The Quartz Scheduler is responsible for executing the Historian tasks and is multi-threaded. You can configure the Quartz Scheduler to control the number of threads available for running Historian tasks.

Note It is possible to configure more than one worker thread of the Historian this, however, is not recommended.

UNDERSTANDING HISTORIAN TASKS

A Historian task moves and deletes data based on its configuration. Historian tasks are created by the Historian process, which also adds Historian tasks to the Quartz Scheduler process. The Quartz Scheduler then instantiates and executes the Historian tasks, based on the task's configured schedules.

When executed by the Quartz Scheduler, the Historian task first determines if there is work to be performed according to the Source property of the task. This property indicates which tables should be processed by the Historian task. The Source property can contain one of these values:

- Historical
- Live
- Logs
- Errs

A Historian task makes the following determinations, based on how you configure it:

Determination	Description
Execute retention processing	The Historian task determines the retention processing mode by examining the UseRetention property. If you set this property to Yes, data retention processing occurs, based on how you configured the task.
Simulate a processing run	The Historian task next determines if it should perform a simulated processing run, by examine the Simulate property. If you set this property to Yes, all data manipulation statements are logged but not executed.
Select and manipulate the data	Finally, the Historian task selects and then manipulates the data as determined by its configuration settings. See the details of the default Historian tasks for additional information.

Default Historian Task Settings

The default Document Factory installation includes these Historian task configurations which are created during the installation:

Job	Description
Archive Jobs	Moves data for jobs completed five or more days ago from the Live Data processing tables to the Historical tables.
Purge Logs	Purges any non-job related rows from the LOGS table which are five or more days old.
Purge Errors	Purges any non-job related rows from the ERRS table which are five or more days old.
Purge History	Purges records from the Historical tables which are more than 30 days old.

Archive Jobs Processing

The Archive Jobs processing is executed if the Source property of the Historian task is set to *Live*. The Archive Jobs process is organized by reversing the hierarchy of the processing tables, which means processing them in this order:

- PUBS (see *PUBS table processing* on page 241 for more information)
- RCPS (see *RCPS table processing* on page 241 for more information)
- BCHS (see *BCHS table processing* on page 242 for more information)
- TRNS (see *TRNS table processing* on page 242 for more information)
- JOBS (see *JOBS table processing* on page 243 for more information)
- BCHS_RCPS

Note The LOGS and ERRS tables are also processed with each of these tables, with the exception of BCHS_RCPS which is processed with the JOBS table

Purge Logs Processing

The Purge Logs processing is executed if the Source property of the Historian task is set to *Logs*. The Historian task deletes all records from the LOGS table in which these criteria are met:

- JOB_ID is null
- LOGTIME value is five or more days old, based on the current system time

You can add more filters to the configuration to further limit the records available for deletion based on columns available in the LOGS table.

Purge Errors Processing

The Purge Errors processing is executed if the Source property of the Historian task is set to *Errs*. The Historian task deletes all records from the ERRS table in which these criteria are met:

- JOB_ID is null
- ERRTIME value is five or more days old, based on the current system time

You can add more filters to the configuration to further limit the records available for deletion based on columns available in the ERRS table.

Purge History Processing

Purge History processing is executed if the Historian's Source property is set to Historical. The process of purging history is organized by reversing the hierarchy of the processing tables. The tables are processed in this order:

- PUBSHIST (see *PUBSHIST table processing* on page 243 for more information)
- RCPSHIST (see *RCPSHIST table processing* on page 244 for more information)
- BCHSHIST (see *BCHSHIST table processing* on page 244 for more information)
- TRNSHIST (see *TRNSHIST table processing* on page 244 for more information)
- TRNSLOG
- JOBSHIST (see *JOBSHIST table processing* on page 245 for more information)
- BCHS_RCPSHIST

Note The TRNSLOG table is processed with the TRNSHIST table. The BCHS_RCPSHIST table is processed with the JOBSHIST table.

Table Processing

Here is an overview of how the Historian task processes the following tables:

- PUBS (see *PUBS table processing* on page 241 for more information)
- RCPS (see *RCPS table processing* on page 241 for more information)
- BCHS (see *BCHS table processing* on page 242 for more information)
- TRNS (see *TRNS table processing* on page 242 for more information)
- JOBS (see *JOBS table processing* on page 243 for more information)
- PUBSHIST (see *PUBSHIST table processing* on page 243 for more information)
- RCPSHIST (see *RCPSHIST table processing* on page 244 for more information)
- PUBHIST (see *BCHSHIST table processing* on page 244 for more information)
- TRNSHIST (see *TRNSHIST table processing* on page 244 for more information)

- JOBSHIST (see *JOBSHIST table processing* on page 245 for more information)

PUBS table processing

The Historian task gets a list of candidate items from the PUBS table. Candidate items are filtered based on these criteria:

- PUBSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes no filters for the PUBS table)

For each candidate PUB record, the Historian task gets the BCH_ID of the related BCH record. The Historian task determines if the related BCH record is complete by querying for BCHSTATUS = 999. If the related BCH record is complete, the Historian task gets the RCP_ID of the related RCP record.

The Historian task determines if the related RCP record is complete by querying for RCPSTATUS = 999. If the related RCP record is complete, the Historian task gets the TRN_ID of the related TRN record by querying the BCHS_RCPS relationship table.

The Historian task determines if the related TRN record is complete by querying for TRNSTATUS = 999. If the related TRN record is complete, the Historian task gets the JOB_ID of the related JOB record by querying the BCHS_RCPS relationship table.

This Historian task determines if the related JOB record is complete by querying for JOBSTATUS = 999. If any of the related items are incomplete, those candidate PUB items are removed from the list for processing by the Historian.

The candidate row items, identified by PUB_ID, are then deleted from the PUBS table. This causes the row data to be removed from the PUBS table and written to the PUBSHIST table. The candidate row item list is also used to delete records from the LOGS and ERRS tables which have matching PUB_ID values.

RCPS table processing

The Historian task gets a list of candidate items from the RCPS table. Candidate items are filtered based on these criteria:

- RCPSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes no filters for the RCPS table)

For each candidate RCP record, the Historian task gets the BCH_ID of the related BCH record. The Historian task determines if the related BCH record is complete by querying for BCHSTATUS = 999. If the related BCH record is complete, the Historian task gets the TRN_ID of the related TRN record by querying the BCHS_RCPS relationship table.

The Historian task determines if the related TRN record is complete by querying for TRNSTATUS = 999. If the related TRN record is complete, the Historian task gets the JOB_ID of the related JOB record by querying the BCHS_RCPS relationship table.

The Historian task determines if the related JOB record is complete by querying for JOBSTATUS = 999. If any of the related items are incomplete, those candidate RCP items are removed from the list for processing by the Historian.

The candidate row items, identified by RCP_ID, are then deleted from the RCPS table. This causes the row data to be removed from the RCPS table and written to the RCPSHIST table. The candidate row item list is also used to delete records from the LOGS and ERRS tables which have matching RCP_ID values.

BCHS table processing

The Historian task gets a list of candidate items from the BCHS table. Candidate items are filtered based on these criteria:

- BCHSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes no filters for the BCHS table)

Before it deletes candidate items from the BCHS table, the Historian task sets the retention date value for all candidate items. The retention date value is stored in the BCHRETENTION column of the BCHS table. The retention date value is calculated by adding number of days specified in the Retention property to the current system time. The default is 30 days.

Before it deletes candidate items from the BCHS table, the Historian task sets the retention date according to the retention processing configuration. For more information, see *Historian Retention Processing* on page 245.

The candidate row items, identified by BCH_ID, are then deleted from the BCHS table. The row data is removed from the BCHS table and written to the BCHSHIST table. The candidate row item list is also used to delete records from the LOGS and ERRS tables with matching BCH_ID values.

TRNS table processing

The Historian task gets a list of candidate items from the TRNS table. Candidate items are filtered based on these criteria:

- TRNSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes no filters for the TRNS table)

Before it deletes candidate items from the TRNS table, the Historian task sets the retention date according to the retention processing configuration. For more information, see *Historian Retention Processing* on page 245.

The candidate row items, identified by TRN_ID, are then deleted from the TRNS table. The row data is removed from the TRNS table and written to the TRNSHIST table. The candidate row item list is also used to delete records from the LOGS and ERRS tables which have matching TRN_ID values.

JOBS table processing

The Historian task gets a list of candidate items from the JOBS table. Candidate items are filtered based on these criteria:

- JOBSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes one filter for the JOBS table which selects the jobs where the JOBENDTIME column value is five or more days past the current system time)

Before it deletes candidate items from the JOBS table, the Historian task sets the retention date according to the retention processing configuration. For more information, see *Historian Retention Processing* on page 245.

The candidate row items, identified by JOB_ID, are then deleted from the JOBS table. The row data is removed from the JOBS table and written to the JOBSHIST table. The candidate row item list is used to delete records from the LOGS and ERRS tables which have matching JOB_ID values.

In addition, the Historian task also deletes records from the BCHS_RCPS table that match the JOB_ID values contained in the list of candidate row items. The row data is removed from the BCHS_RCPS table and written to the BCHS_RCPSHIST table.

PUBSHIST table processing

The Historian task gets a list of candidate items from the PUBSHIST table. Candidate items are filtered based on these criteria:

- PUBSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes no filters for the PUBSHIST table)

For each candidate PUBSHIST record, the Historian task gets the BCH_ID of the related BCHSHIST record. The Historian task determines if the related BCHSHIST record is complete by querying for BCHSTATUS = 999. If the related BCHSHIST record is complete, the Historian task gets the RCP_ID of the related RCPHIST record.

The Historian task determines if the related RCPHIST record is complete by querying for RCPSTATUS = 999. If the related RCPHIST record is complete, the Historian task gets the TRN_ID of the related TRNSHIST record by querying the BCHS_RCPSHIST relationship table.

The Historian task determines if the related TRNSHIST record is complete by querying for TRNSTATUS = 999. If the related TRNSHIST record is complete, the Historian task gets the JOB_ID of the related JOBSHIST record by querying the BCHS_RCPSHIST relationship table.

This Historian task determines if the related JOBSHIST record is complete by querying for JOBSTATUS = 999. If any of the related items are incomplete, the candidate PUBHIST item is removed from the list for processing by the Historian.

The candidate row items, identified by PUB_ID, are then deleted from the PUBSHIST table.

RCPSHIST table processing

The Historian task gets a list of candidate items from the RCPSHIST table. Candidate items are filtered based on these criteria:

- RCPSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes no filters for the RCPS table)

For each candidate RCPSHIST record, the Historian task gets the BCH_ID of the related BCHSHIST record. The Historian task determines if the related BCHSHIST record is completed by querying for BCHSTATUS = 999. If the related BCHSHIST record is complete, the Historian task gets the TRN_ID of the related TRNSHIST record by querying the BCHS_RCPSHIST relationship table.

The Historian task determines if the related TRNSHIST record is complete by querying for TRNSTATUS = 999. If the related TRNSHIST record is complete, the Historian task gets the JOB_ID of the related JOBSHIST record by querying the BCHS_RCPSHIST relationship table.

This Historian task determines if the related JOBSHIST record is complete by querying for JOBSTATUS = 999. If any of the related items are incomplete, the candidate RCPSHIST item is removed from the list for processing by the Historian.

The candidate row items, identified by RCP_ID, are then deleted from the RCPSHIST table.

BCHSHIST table processing

The Historian task gets a list of candidate items from the BCHSHIST table. Candidate items are filtered based on these criteria:

- BCHSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes no filters for the BCHSHIST table)

The list of candidate items is filtered by excluding candidate items where the RETHOLD column is not equal to zero (0) or the date value of the RETENTION column is greater than the current system date.

The definition of the RETENTION column for the BCHSHIST table is contained in the BchsRetentionColumn property. The default is BCHRETENTION.

The definition of the RETHOLD column for the TRNSHIST table is contained in the RetHoldColumn property. The default is RETHOLD. This value is appended to BCH to make the column name BCHRETHOLD.

The candidate row items, identified by BCH_ID, are then deleted from the BCHSHIST table.

TRNSHIST table processing

The Historian task gets a list of candidate items from the TRNSHIST table. Candidate items are filtered based on these criteria:

- TRNSTATUS = 999 (999 is the job completed status)

- Task filter configurations (the default installation includes no filters for the TRNSHIST table)

The list of candidate items is filtered by excluding candidate items where the RETHOLD column is not equal to zero (0) or the date value of the RETENTION column is greater than the current system date.

The definition of the RETENTION column for the TRNSHIST table is contained in the TrnsRetentionColumn property. The default is RETENTION.

The definition of the RETHOLD column for the TRNSHIST table is contained in the RetHoldColumn property. The default is RETHOLD. This value is appended to TRN to make the column name TRNRETHOLD.

The candidate row items, identified by TRN_ID, are then deleted from the TRNSHIST table. In addition, the Historian task also deletes records from the TRNSLOG table that match the TRN_ID values contained in the list of candidate row items.

JOBSHIST table processing

The Historian task gets a list of candidate items from the JOBSHIST table. Candidate items are filtered based on these criteria:

- JOBSTATUS = 999 (999 is the job completed status)
- Task filter configurations (the default installation includes one filter for the JOBS table which selects the jobs where the JOBENDTIME column value is five or more days past the current system time)

The list of candidate items is filtered by excluding candidate items where the RETHOLD column is not equal to zero (0) or the date value of the RETENTION column is greater than the current system date.

The definition of the RETENTION column for the BCHSHIST table is contained in the JobsRetentionColumn property. The default is JOBRETENTION.

The definition of the RETHOLD column for the JOBSHIST table is contained in the RetHoldColumn property. The default is RETHOLD. This value is appended to JOB to make the column name JOBRETHOLD.

The candidate row items, identified by JOB_ID, are then deleted from the JOBSHIST table. In addition, the Historian task also deletes records from the BCHS_RCPSHIST table that match the JOB_ID values contained in the list of candidate row items.

Historian Retention Processing

Retention processing *prevents* data from being removed from the historical database tables. The Historian task includes default tasks which move data from the live data tables into the historical tables. For some tables (BCHS, JOBS, and TRNS) there are columns which hold a retention value. This value defines the date and time until which the row of data in the table must be stored.

When the Historian task processes the live data tables, the retention date is set as the system moves data into the historical tables. When the Historian task processes the historical data tables, the retention date of each is compared to the current system date and time. If the retention date is in the future, the row is not removed from the historical data tables. If the retention date is in the past, the row is removed from the historical data processing tables.

There are two types of retention processing:

Type	Description
Simple	Simple retention processing applies a uniform retention date to all records processed by a particular Task. This is the default. For more information, see *Simple retention processing* on page 246.
Complex	Complex retention processing allows for rules-based application of retention dates. For more information, see *Complex retention processing* on page 247.

To enable Simple or Complex retention processing, select the appropriate value for the RetentionType property. You must also set the UseRetention property to Yes for retention processing to occur.

The following topics use this notation when referring to properties:

```
[Context]/[Category]/[Group]@[Property]=[Value]
```

If the Context, Category and Group are assumed to be known. Here are some examples:

To specify a property	Use this notation
With full context	RETENTION/Configuration/Historian@UseRetention
And value	RETENTION/Configuration/Historian@UseRetention=true
And value choices	RETENTION/Configuration/Historian@UseRetention=[true|false]
Without context	@UseRetention

Simple retention processing

If you chose Simple retention processing and set the UseRetention property to Yes, the system performs these steps:

1. If the current table being processed is BCHS, JOBS, or TRNS, the value of TASK/[Category]/Configuration@Retention is added to the current system date and time to calculate the retention date.

Note The Category of the setting can be anything, as long as TASK/[Category]/Configuration@SourceLive. The default category for this task is Archive Completed Jobs.

2. The calculated value of the retention date is then written to the column defined by the appropriate property setting:

If the table is	The calculated retention date is written to the column defined in
BCHS	DATABASE/Configuration/Retention@BchsRetentionColumn
JOBS	DATABASE/Configuration/Retention@JobsRetentionColumn
TRNS	DATABASE/Configuration/Retention@TrnsRetentionColumn

Complex retention processing

If you chose Complex as the RetentionType and set the UseRetention property to Yes, the system performs these steps when processing the BCHS, JOBS, or TRNS table:

1. The Historian task examines the list of retention filters for those with RETENTION/[Category]/Filter@Enabled=Checked.

Note Retention filters are defined in the RETENTION/[Category]/Filter property group, where *Category* is the name of the filter. You can use any category name, however, the general convention is to name the filter with the prefix *Filter* so all retention filters are displayed in close proximity in the Documaker Administrator. In the default installation, these filters are created:

- Filter:BCHS
- Filter:JOBS
- Filter:TRNS

2. The Historian task examines the RETENTION/[Category]/Filter@Field property to determine the applicable table for this filter. The Field property must be defined as shown here:

   ```
   [TABLE].[COLUMN]
   ```

   If the TABLE defined in the Field property matches the table being processed, the system gets the COLUMN value for each row being moved to the historical tables.

3. To determine if the filter is applicable to the row being processed, the system compares these properties to the column value obtained from RETENTION/[Category]/Filter@Field:

   - RETENTION/[Category]/Filter@Field, RETENTION/[Category]/Filter@Operator
   - RETENTION/[Category]/Filter@Value

   If the values match, the system applies the retention settings in the current filter (RETENTION/[Category]/Filter).

   If the values do not match, the system evaluates the next filter in the list.

   If all filters have been evaluated and none apply, the system applies the default retention settings in RETENTION/Configuration/Default.

4. The system determines the base retention date using the @BaseRetentionDate property.

If the value is	The system uses
Current Date	The current system date and time

Column	The system gets the value of the @BaseRetentionColumn property. This value must be specified as in this format:
	`[TABLE].[COLUMN]`
	Note that the TABLE value must match the table being processed. The value of this property is used to get a base date and time value from the live data tables. You can specify the format of the date in this column using the @BaseRetentionDateFormat property, which follows the Simple Date Format patterns.The default format is shown here:
	`EEE MMM dd H:mm:ss z yyyy`
	For more information on the Date and Time patterns you can use, see this web site:
	http://download.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html

Note If the system cannot find a date in the @BaseRetentionDate or @BaseRetentionColumn properties, it displays a warning and does not set a retention date for that row. If the date cannot be parsed using the format specified by @BaseRetentionDateFormat property, the system displays a warning and the does not set a retention date for that row.

5. Use these properties to specify the amount of time you want to add to the base retention date:

Property	Description
@RetentionCalc	The number of time units to be added to the base retention date. Enter a positive number.
@RetentionCalcType	Defines the type of time unit specified by @RetentionCalc. You can choose from these values: • Years • Months • Days Here are some examples: BaseRetentionDate = Current Date “01/01/2011” RetentionCalc = 12 RetentionCalcType = Months Calculated retention date = “01/01/2012”

6. The system calculates the retention date and writes it to the column defined by the appropriate property setting:

If the table is	The calculated retention date is written to the column defined in
BCHS	DATABASE/Configuration/Retention@TrnsRetentionColumn
JOBS	DATABASE/Configuration/Retention@TrnsRetentionColumn
TRNS	DATABASE/Configuration/Retention@TrnsRetentionColumn

USING HISTORIAN CONFIGURATION RESOURCES

The configuration for the Historian is stored in these resources:

Resource	Description
historian.jar file	Contains the minimal startup configuration information.
.bindings file	Contains the Java Naming and Directory Interface (JNDI) data sources.
APPCONFIGCONTEXT table	Contains the configuration options.
ALCONFIGCONTEXT table	Contains the minimal logging configuration options for the Historian.

historian.jar file

The historian.jar file is located in the deploy subdirectory of the Document Factory. It contains these configuration components:

Component	Description
deploy.properties	Contains the minimal startup configuration information.
log4j.xml	Used to control the different Log4J loggers to capture diagnostic and error output. Log4j is a Java logging or tracing API. For more information, see this web site: http://logging.apache.org/log4j
log4j.dtd	Used by the log4j.xml file.

deploy.properties file

The deploy.properties file is extracted and placed in the temp/historian working directory. This file contains the minimal startup configuration properties used to read the configuration for the Historian from the ALCONFIGCONTEXT and APPCONFIGCONTEXT tables:

Property	Description
system.id	Contains the value of SYS_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Historian configuration.
assemblyline.id	Contains the value of AL_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Historian configuration.
application.id	Contains the value of APP_ID column in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Historian configuration.
config	Contains the configuration name for the Historian. The default is Historian. This value overrides the value derived from the configuration jar file name. The value provided for this option is used as the GROUP_NAME column value in the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables for the Historian configuration.
config.jndi.name	Contains the Java Naming and Directory Interface (JNDI) name for the data source that contains the APPCONFIGCONTEXT and ALCONFIGCONTEXT tables.
config.schema	Contains the database schema used for the ALCONFIGCONTEXT and APPCONFIGCONTEXT configuration tables.

Property	Description
factory.jndi.name	Contains the JNDI name for the data source that contains the assembly line tables.
factory.schema	Contains the database schema used for the assembly line tables.

Here is an example:

```
system.id=1
assemblyline.id=1
application.id=12
config=Historian
config.jndi.name=DMKRConfig
config.schema=dmkr_admin
factory.jndi.name=DMKRFactory
factory.schema=dmkr_asline
```

Note The entries *dmkr_asline* and *dmkr_admin* may be different if they were changed during the installation.

log4j.xml file

The log4j.xml file is extracted and placed in the temp/historian working directory. This file contains loggers you can enable at run time to capture diagnostic information.

Logger	Description
oracle.documaker.historian	Logs Historian messages.
oracle.documaker.historian.housekeeping.HistorianHouseKeeper	Logs HouseKeeper messages.
oracle.documaker.historian.shutdown.HistorianShutdownHook	Logs shutdown hook messages.
org.quartz	Logs Quartz Scheduler messages as well as Historian task messages.
oracle.documaker.dao.AbstractDAO	Logs Data Access Object (DAO) messages.
oracle.documaker.db.Query	Logs Java Database Connectivity (JDBC) queries.
oracle.documaker.db.DataSourceUtil	Logs JDBC data source messages.
oracle.documaker.dao.DAOUtil	Logs JDBC data source and DAO messages.
oracle.documaker.process.ProcessShell	Logs process shell messages.
oracle.documaker.process.ipc.*	Logs messages related to inter-process communication.

Here is an example of a logger:

```
<category name="oracle.documaker.historian" additivity="false">
   <priority value="error"/>
   <appender-ref ref="stdout"/>
   <appender-ref ref="roll"/>
</category>
```

Note Switch the Priority value from *error* to *debug* to capture diagnostic information. See the Apache Log4J project for details about Log4J.

You can modify the file inside historian.jar file or the one that is written to the Document Factory temp\historian working directory when the Historian is started and deployed.

The file in the temp\historian directory is overwritten each time Document Factory is restarted or the Historian process is restarted. You can, however, modify the one in the temp\historian directory to apply changes at run time without restarting the Historian process.

See *Logging to the Database* on page 273 for more information.

.bindings file

The .bindings file is located in Document Factory's config\context subdirectory. It contains the Java Naming and Directory Interface (JNDI) data sources used by the Historian. Each JNDI data source contains these configuration properties:

Property	Description
ClassName	Contains the data source fully-qualified class name. Use this value: `javax.sql.DataSource`
FactoryName	Contains the data source factory fully-qualified class name. Use this value: `org.apache.commons.dbcp.BasicDataSourceFactory` The BasicDataSourceFactory class supports connection pooling.
driverClassName	Contains the Java Database Connectivity (JDBC) driver class name.
url	Contains the JDBC URL.
maxOpenPreparedStatements	Defines the maximum number of prepared statements to cache in the connection pool. Enter -1 to indicate there is no limit.
timeBetweenEvictionRunsMillis	Defines how often the idle object evictor thread should run and perform clean up of the stale connection handles. Enter -1 to disable the idle object evictor thread.
validationQuery	Contains the validation query you want to run when borrowing objects from the connection pool.
username	Contains the JDBC user name.
password	Contains the JDBC password.
testOnBorrow	Indicates if validationQuery should be used when borrowing and object from the connection pool.
initialSize	Defines the initial connection pool size.
maxActive	Defines the maximum number of active connections in the pool.
maxIdle	Defines the maximum number of idle connections in the pool.
minIdle	Defines the minimum number of idle connections in the pool.
maxWait	Defines the maximum amount of time, in milliseconds, to wait for a connection object to be retrieved from the pool before issuing an error.

Here is an example:

```
#Unix friendly Documaker Config JNDI DataSource
DMKRConfig/ClassName=javax.sql.DataSource
DMKRConfig/
FactoryName=org.apache.commons.dbcp.BasicDataSourceFactory
DMKRConfig/RefAddr/0/Content=oracle.jdbc.driver.OracleDriver
DMKRConfig/RefAddr/0/Encoding=String
DMKRConfig/RefAddr/0/Type=driverClassName
DMKRConfig/RefAddr/1/
Content=jdbc\:oracle\:thin\:@localhost\:1521\:IDMAKER
DMKRConfig/RefAddr/1/Encoding=String
DMKRConfig/RefAddr/1/Type=url
DMKRConfig/RefAddr/10/Content=-1
DMKRConfig/RefAddr/10/Encoding=String
DMKRConfig/RefAddr/10/Type=maxOpenPreparedStatements
DMKRConfig/RefAddr/11/Content=-1
DMKRConfig/RefAddr/11/Encoding=String
DMKRConfig/RefAddr/11/Type=timeBetweenEvictionRunsMillis
DMKRConfig/RefAddr/12/Content=select 1 from dual
DMKRConfig/RefAddr/12/Encoding=String
DMKRConfig/RefAddr/12/Type=validationQuery
DMKRConfig/RefAddr/2/Content=dmkr_admin
DMKRConfig/RefAddr/2/Encoding=String
DMKRConfig/RefAddr/2/Type=username
DMKRConfig/RefAddr/3/Content=oracle12
DMKRConfig/RefAddr/3/Encoding=String
DMKRConfig/RefAddr/3/Type=password
DMKRConfig/RefAddr/4/Content=true
DMKRConfig/RefAddr/4/Encoding=String
DMKRConfig/RefAddr/4/Type=testOnBorrow
DMKRConfig/RefAddr/5/Content=1
DMKRConfig/RefAddr/5/Encoding=String
DMKRConfig/RefAddr/5/Type=initialSize
DMKRConfig/RefAddr/6/Content=8
DMKRConfig/RefAddr/6/Encoding=String
DMKRConfig/RefAddr/6/Type=maxActive
DMKRConfig/RefAddr/7/Content=8
DMKRConfig/RefAddr/7/Encoding=String
DMKRConfig/RefAddr/7/Type=maxIdle
DMKRConfig/RefAddr/8/Content=0
DMKRConfig/RefAddr/8/Encoding=String
DMKRConfig/RefAddr/8/Type=minIdle
DMKRConfig/RefAddr/9/Content=60000
DMKRConfig/RefAddr/9/Encoding=String
DMKRConfig/RefAddr/9/Type=maxWait
```

Configuring the Historian Worker

The Historian reads configuration information from the deploy.properties file and APPCONFIGCONTEXT table.

Configuring the APPCONFIGCONTEXT table

These properties are read from this table when the GROUP_NAME column value is *Historian*:

Property	Description
StartCommand	Defines the start command. This value is used by the Supervisor to start the class specified in the JavaClass configuration option. The default is docfactory_historian.
StartArguments	Defines the start arguments for JavaClass. There is no default.
JavaClass	Contains the Java class used to start the worker class specified in the WorkerClass configuration option. Use this value: `oracle.documaker.process.ProcessShell` The ProcessShell class is a process shell that provides all of the functionality needed to communicate with the Supervisor process and to start and manage the worker class specified in WorkerClass configuration option.
JVMOptions	Lists any JVM options the Supervisor process should use to start JavaClass. There is no default, however, this value is initially configured upon installation.
Instances	Defines the number of instances the Supervisor should start for the Historian. This value should always be set to one (1).
UseLoadBalancing	Defines whether to use load balancing. This value should always be set to No.
MaxReportIntervalSeconds	(Optional) This option controls the maximum time interval that can elapse without an instance reporting back to the Supervisor before the instance is restarted. The default is 120 seconds.
MaxUpTimeSeconds	(Optional) This option controls the maximum time interval an instance can run before it is restarted by the Supervisor. The default is -1, which disables this option.
WaitForShutdownSeconds	(Optional) This option controls how long the Supervisor waits for an instance to shut down after it issues a shutdown command and before it terminates the instance. The default is 20 seconds.
WatchList	Contains a comma-delimited list of disk and file resources to watch for a change. If a change is detected, the instances of a process are restarted.
MaxRestarts	(Optional) This option controls the maximum number of restart attempts that can occur. The default is five (5).
WorkerClass	Defines the class that extends the oracle.documaker.process.worker.Worker Thread class and is started by the class specified in JavaClass configuration option. This value should always be set as shown here: `oracle.documaker.historian.Historian`

Property	Description
WorkerThreads	Defines how many threads of WorkerClass should be created by JavaClass. This value should always be set to one (1).
WorkerIntervalMillis	Defines how often each WorkerClass thread should perform its work. The default is 5000 milliseconds. Since the Historian worker does not perform any actual work, you should set this to a very high value, such as 360000.
WorkerStartDelayMillis	Defines how long each WorkerClass thread should wait after startup and before performing any work. The default is 10000 milliseconds. Since the Historian is not typically needed for immediate startup execution, you should set this to a value higher than the other workers.
ShutdownHookClass	This class extends the oracle.documaker.process.shutdown.ShutdownHook class. This value should always be as shown here: `oracle.documaker.historian.shutdown.HistorianShutdownHook`
HouseKeeperClass	Defines the class that extends the oracle.documaker.process.housekeeping.HouseKeeper class. This value should always be set as shown here: `oracle.documaker.historian.housekeeping.HistorianHouseKeeper`
HouseKeeperIntervalMillis	Defines how often the HouseKeeperClass thread should perform its work. The default is 15000 milliseconds.
HouseKeeperStartDelayMillis	Defines how long the HouseKeeperClass thread should wait after startup and before performing any work. The default is 30000 milliseconds.
IPCIntervalMillis	Defines how often the inter-process communication (IPC) thread should perform its work. This option is used by JavaClass to report back to the Supervisor process. The default is 1000 milliseconds.
IPCStartDelayMillis	Defines how long the inter-process communication (IPC) thread should wait after startup and before performing any work. This option is used by JavaClass to report back to the Supervisor process. The default is 10000 milliseconds.
Log4jIntervalMillis	Defines how often the Log4J resource monitor thread should perform its work. This option is used to monitor the log4j.xml file deployed under the temp\historian working directory and reload it when a change is detected. The default is 1000 milliseconds.
Log4jStartDelayMillis	Defines how long the Log4J resource monitor thread should wait after startup and before performing any work. This option is used to monitor log4j.xml file deployed under the temp\historian working directory and reload it when a change is detected. The default is 10000 milliseconds.
BchsRetentionColumn	The name of the column in the BCHS and BCHSHIST tables that contains the retention date value. The default is BCHRETENTION.
ErrsTableName	The name of the table which contains error logging. The default is ERRS.

Property	Description
JobsRetentionColumn	Contains the name of the column in the JOBS and JOBSHIST tables that contains the retention date value. The default is JOBRETENTION.
LogsTableName	Contains the name of the table which contains logging. The default is LOGS.
RetHoldColumn	Contains the base name of the column in the JOBS, BCHS, TRNS, JOBSHIST, TRNSHIST, and BCHSHIST tables that contains the retention hold value. The default is RETHOLD. The value of this setting is appended to one of these, based on the table name: For the JOBS and JOBSHIST tables, the column name is JOBRETHOLD For the TRNS and TRNSHIST tables, the column name is TRNRETHOLD For the BCHS and BCHSHIST tables, the column name is BCHRETHOLD
TrnsRetentionColumn	Defines name of the column in the TRNS and TRNSHIST tables which contains the retention date value. The default is RETENTION.
TrnslogTableName	Defines the name of the table which contains the transaction data. The default is TRNSLOG.
UseRetention	This indicates if retention processing should be used. If set to Yes, rows from the JOBS, BCHS, or TRNS tables are moved to historical data tables and the appropriate RETENTION columns are updated. Additionally, when creating criteria for selecting data during the Purge History job, the appropriate RETHOLD column is inspected. If set to No, none of this functionality is enabled.

Here is an example (only the Property and Value columns are shown):

Property	Value
StartCommand	docfactory_historian
JavaClass	oracle.documaker.process.ProcessShell
JVMOptions	-Xmx256m -Duser.name=oracle -Djava.library.path=C:/oracle/oracle_insurance_1/documaker/bin
Instances	1
UseLoadBalancing	No
WorkerClass	oracle.documaker.historian.Historian
WorkerThreads	1
WorkerIntervalMillis	360000
WorkerStartDelayMillis	20000
ShutdownHookClass	oracle.documaker.historian.shutdown.HistorianShutdownHook
HouseKeeperClass	oracle.documaker.historian.housekeeping.HistorianHouseKeeper
HouseKeeperIntervalMillis	300000
HouseKeeperStartDelayMillis	100000

Property	Value
IPCIntervalMillis	1000
IPCStartDelayMillis	10000
Log4jIntervalMillis	5000
Log4jStartDelayMillis	10000
BchsRetentionColumn	BCHRETENTION
ErrsTableName	ERRS
JobsRetentionColumn	JOBRETENTION
LogsTableName	LOGS
RetHoldColumn	RETHOLD
TrnsRetentionColumn	RETENTION
TrnslogTableName	TRNSLOG
UseRetention	Yes
RetentionType	1 or 2 (simple or complex)

CONFIGURING THE QUARTZ SCHEDULER

The Historian reads configuration information from the deploy.properties file and the APPCONFIGCONTEXT table. The following configuration options are passed to the Quartz Scheduler when it is created by the Historian.

Note Consult the Quartz Scheduler documentation at this web site for additional information:
http://www.quartz-scheduler.org/docs/1.x/configuration/index.html

Configuring the APPCONFIGCONTEXT table

These properties are read from this table when the GROUP_NAME column value is *Historian-Quartz*:

Property	Description
org.quartz.jobStore.class	Contains the name of the Quartz Scheduler class used to store Historian tasks.
org.quartz.jobStore.misfireThreshold	Defines the number of milliseconds the Scheduler will allow a trigger to go past its next-fire-time by, before being considered misfired. The default is 60000 (60 seconds).
org.quartz.scheduler.instanceName	This can be any string and the value has no meaning to the Scheduler itself, but rather serves as a mechanism for client code to distinguish Schedulers when multiple instances are used within the same program.
org.quartz.scheduler.rmi.export	This property is not currently used and should be set to No.
org.quartz.scheduler.rmi.proxy	This property is not currently used and should be set to No.
org.quartz.scheduler.skipUpdateCheck	Determines whether to skip running a web request to determine if there is an updated version of Quartz Scheduler available for download. If the check runs and an update is found, it is reported as available in Quartz Scheduler's logs.
org.quartz.scheduler.wrapJobExecution InUserTransaction	Set to Yes if you want Quartz Scheduler to start a UserTransaction before calling execute on your job. The Tx will commit after the job's execute method completes and after the JobDataMap is updated (if it is a StatefulJob). The default is No.
org.quartz.threadPool.class	Specifies the name of the threadPool implementation you want to use. The threadPool shipped with Quartz Scheduler is named *org.quartz.simpl.SimplethreadPool,* and should suffice for nearly every user. It provides a fixed-size pool of threads that last the lifetime of the Scheduler.

Property	Description
org.quartz.threadPool.threadCount	Defines the number of threads available for the concurrent execution of jobs. Typically, you will enter a number between 1 and 100. If you only have a few jobs that fire a few times a day, one (1) thread is plenty. If you have tens of thousands of jobs, with many firing every minute, you probably want a thread count of 50 or 100. The number you enter greatly depends on the nature of the work your jobs perform and your system's resources.
org.quartz.threadPool.threadPriority	Can be any number between Thread.MIN_PRIORITY (which is 1) and Thread.MAX_PRIORITY (which is 10). The default is Thread.NORM_PRIORITY (which is 5).
org.quartz.threadPool.threadsInheritContextClassLoaderOfInitializingThread	Specifies whether threads spawned by Quartz Scheduler inherit the context ClassLoader of the thread that initializes the Quartz Scheduler instance. This affects the following Quartz Scheduler threads: • The main scheduling thread • The JDBCjobStore's misfire handling thread (if JDBCjobStore is used) • The cluster recovery thread (if clustering is used) • Any threads in SimplethreadPool (if SimplethreadPool is used) Setting this to Yes may help with class loading, JNDI look-ups, and other issues related to using Quartz Scheduler within an application server.

Here is an example (only the Property and Value columns are shown):

Property	Value
org.quartz.jobStore.class	org.quartz.simpl.RAMjobStore
org.quartz.jobStore.misfireThreshold	60000
org.quartz.scheduler.instanceName	HistorianQuartzScheduler
org.quartz.scheduler.rmi.export	No
org.quartz.scheduler.rmi.proxy	No
org.quartz.scheduler.skipUpdateCheck	Yes
org.quartz.scheduler.wrapJobExecutionInUserTransaction	Yes
org.quartz.threadPool.class	org.quartz.simpl.SimplethreadPool
org.quartz.threadPool.threadCount	1
org.quartz.threadPool.threadPriority	5
org.quartz.threadPool.threadsInheritContextClassLoaderOfInitializingThread	Yes

CONFIGURING HISTORIAN TASKS

The Historian reads configuration information from the deploy.properties file and APPCONFIGCONTEXT table. Use the following configuration properties to create instances of Historian tasks, which are executed by Quartz Scheduler.

Configuring the APPCONFIGCONTEXT Table

These options are read from this table when the CONTEXT_NAME column is *Task*:

Property	Description
Enabled	If set to Yes, the Historian task is configured and added to Quartz Scheduler for execution. If set to No, the Historian task is ignored.
Filters	A comma-delimited list of filters which you want applied to this task. The values you enter here should match the filters defined as Group Names in the FILTERS/CFG context/category.
Priority	A number from 1 to 10 which indicates the priority of this Historian task over any other Historian tasks. This is used by the Quartz Scheduler. The default is five (5).
Retention	When the RetentionType is set to 1 (simple), this number indicates the number of days to add to the current system date when setting the RETENTION date value for applicable data in JOBS, BCHS, and TRNS tables.
Schedule	A Quartz Scheduler notation that indicates the schedule for executing this Historian task. Here is the default: `0 59 23 ? * SUN` This indicates every Sunday at 11:59 PM See *Using the CronTrigger Class* on page 268 for additional information:
Simulate	If you set this option to Yes, the activity is logged, but the deletion does not physically occur. If you set this option to No, the deletion activities defined for the Historian task take place.
Source	Indicates the data source for this Historian task. Choose one of these values: • Historical • Live • Logs • Errs

Here is an example (only the Property and Value columns are shown):

Property	Value
Enabled	Yes
Filters	1
Priority	5
Retention	30
Schedule	0 59 23 ? * SUN
Simulate	No

Property	Value
Source	Live

Setting Up Historian Task Filters

Each Historian task can have as many filters as needed to control which records are available to the task for processing. Each filter creates a *WHERE* clause that is appended to the selection of records.

These properties are read from this table when the CONTEXT_NAME column value is *FILTERS* and the CATEGORY column value is *CFG*:

Property	Description
Field	Defines the table and column name used in the filter, in the format `tablename.columnname` All columns in these tables are possible values: • JOBS • BCHS • RCPS • TRNS • PUBS • LOGS • ERRS Note: Special processing occurs for any column name containing the string *TIME*, such as JOBENDTIME, LOGTIME or ERRTIME. When this column name is used in a filter, the value specified by the Value property is added (or subtracted if the value is negative) to the current system time to make a date comparison.
Operator	Specifies a logical operator that is applied to the filtering condition.
Value	Contains the value used in the filtering condition.
ValueType	Specifies the Java class that defines the data type of the value supplied. This is used when configuring the filter, which requires certain values to be supplied with a data type. You can choose from these options: • java.lang.Integer • java.lang.Double • java.lang.Long All other values are assumed to be java.lang.String.

Here is an example (only the Property and Value columns are shown):

Property	Description
Field	JOBS.JOBENDTIME
Operator	
Value	-5

This filter yields this WHERE clause:

```
WHERE JOBS.JOBENDTIME < CURRENT_SYSTEM_TIME + -5
```

In this case, all records in JOBS where the JOBENDTIME are more than five days in the past would be subject to processing by this task.

Setting Up Historian Retention Filters

You can define as many Retention filters as needed to control how the retention date is set for data rows in the BCHS, JOBS, and TRNS tables. The system evaluates each filter against the table row data being processed. If the filter matches, the retention date settings for that filter are applied to row data.

These properties are read from this table when the CONTEXT_NAME column value is *RETENTION* and the GROUP column value is *Filter*. The CATEGORY column is the name of the filter.

Property	Description
Field	Defines the table and column name used in the filter, in the format `tablename.columnname` All columns in these tables are possible values: • JOBS • BCHS • TRNS
Operator	Specifies a logical operator that is applied to the filtering condition.
Value	Contains the value used in the filtering condition.
BaseRetentionColumn	If you set the BaseRetentionDate property to Column, the system uses the value you set for this property to get a date value. It then uses this date value as the basis for calculating the retention date. Specify this value as shown here: `[TABLE].[COLUMN]` Note that the TABLE value must match the table being processed and must be BCHS, JOBS, or TRANS. The system uses your entry to get a base date/time value from the live data tables.
BaseRetentionDate	Specifies the value to use for the base retention date. You can chose from these values: • Column • Current Date If you choose Current Date, the current system date/time is used as the basis for calculating the retention date.

Property	Description
Enabled	If checked, the system examines the filter for each data record being processed by the Historian task. **Note**: When you create a Retention filter, you must change the Value Type of this property to Boolean to show a check box. Otherwise you can accept the default Value Type (Alphanumeric) and enter one (1) for enabled or zero (0) for disabled.
RetentionCalc	Specifies the number of time units to be added to the base retention date. Enter a positive number.
RetentionCalcType	Defines the type of time unit specified by the RetentionCalc property. You can choose from the following: • Years • Months • Days
BaseRetentionDateFormat	Defines the format of the date. If you do not specify a format here, this default is used: `EEE MMM dd H:mm:ss z yyyy` For more information on the Date and Time patterns you can use, see this web site: http://download.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html

Here is an example (only the Property and Value columns are shown):

Property	Description
Field	TRNS.KEY1
Operator	=
Value	COMPANY
BaseRetentionDate	Column
BaseRetentionColumn	TRNS.TRNENDTIME
RetentionCalc	1
RetentionCalcType	Years

The system will apply this filter if the record being processed...

- Is in the TRNS table
- Has the TRNS.KEY1 value set to COMPANY

The system adds one year to the TRS.TRNENDTIME value to calculate the retention date. Note that since the BaseRetentionDateFormat property was not specified, the system uses the default date format:

```
EEE MMM dd H:mm:ss z yyyy
```

CREATING HISTORIAN TASKS

This topic explains how to create and configure a Historian task using Documaker Administrator. You can also create and configure a Historian task manually by making entries in the Assembly Line database. This, however, is not recommended.

To create a Historian task, first open a browser and go to the URL for Documaker Administrator. Enter the appropriate user credentials to log into the system. The URL and credentials for the Documaker Administrator can be provided by your system administrator. Here is an example:

```
http://localhost:10001/DocumakerAdministrator
```

Then follow these steps:

1. Click the Systems link. Expand the Systems and choose the assembly line in which you want to create a task.
2. Expand the assembly line and click the Historian application. Click the Configure button, which will open a new tab.
3. Click the Create Context button and enter the details for your new task:

Property	Description
Context Name	Enter **TASK**.
Category	Enter the unique name of your task.
Group Name	Enter **Configuration**.
Property	You must add one property when creating a Context. Add the property *Enabled*.
Value	You must add a value for your property. Add the value *False*. You can change this later if necessary.

   Click Ok.

4. Expand the newly-created context, and select the Configuration group. You can now create the additional properties for the task.
   - Click the + icon in the Properties pane to create a new property row.
   - Click the Property column and enter the name of the property. Refer to *Configuring Historian Tasks* on page 259 for a list of the properties you can enter here.

Note When entering the Schedule property, you can use the wizard to build the Quartz Scheduler string for you, rather than deciphering the syntax yourself.

   - Click the Value column and enter the value of the property. Then click Save to save the new property.

Note You must include the Enabled property, set to a value of Yes, if you want the task to be included in the Historian's runtime execution.

Creating a Filter for a Historian Task

To create a filter for a Historian task, first open a browser and navigate to the URL for Documaker Administrator. Enter the appropriate user credentials to log in to the system. The URL and credentials for the Documaker Administrator can be provided by your system administrator. Here is an example:

```
http://localhost:10001/DocumakerAdministrator
```

Then, follow these steps:

1. Click the Systems link. Expand the Systems group and choose the assembly line in which you want to create a job.
2. Expand the assembly line and click the Historian row, then click the Configure button.
3. Expand the assembly line and click the FILTERS CFG context/category combination.
4. Click the Create Group button, then use these properties to define the group:

Property	Description
Group Name	Name your filter in a unique manner. The name you use here is referenced in the tasks that use this filter.
Property	You must add one property when the property group is created. Create a Property field and set the value to the name of the table and field that contains the field you for which you are creating this filter. Here is an example: `JOBS.JOBSTATUS` You can change this later if necessary.

   Click Ok when finished.

5. Click to select the group for the filter you just created. You can now define additional properties for the filter. Perform these steps to create a property:
   - Click the + icon in the Properties panel to create a new property row.
   - Click the Property column and enter the name of the property. See *Setting Up Historian Task Filters* on page 260 for a list of properties you can enter here.
   - Click the Value column and enter the value of the property.

   Click Save when finished.

Creating a Retention Filter

To create a filter for a Retention task, first open a browser and navigate to the URL for Documaker Administrator. Enter the appropriate user credentials to log in to the system. The URL and credentials for the Documaker Administrator can be provided by your system administrator. Here is an example:

```
http://localhost:10001/DocumakerAdministrator
```

Then, follow these steps:

1. Click the Systems link. Expand the Systems group and choose the assembly line in which you want to create a job.
2. Expand the assembly line and click the Historian row, then click the Configure button.
3. Click the FILTERS CFG context/category combination.
4. Click the Create Context button, then use these properties to define the filter:

Property	Description
Context Name	Enter **RETENTION**.
Category Name	Enter a unique name for your filter.
Group Name	Enter **Filter**.
Property	You must add one property when you create the property group. Select one of the properties from those discussed in *Setting Up Historian Retention Filters* on page 261 to use for your first property.

   Click Ok when finished.
5. Click to select the group for the filter you just created. You can now define additional properties for the filter. Perform these steps to create a property:
   - Click the + icon in the Properties panel to create a new property row.
   - Click the Property column and enter the name of the property. For a list of properties you can choose from, see *Setting Up Historian Retention Filters* on page 261.
   - Click the Value column and enter the value of the property.

   Click Save when finished.

LOGGING HISTORIAN INFORMATION

By default, the system logs only error information and this information is logged to the database tables. You may want to enable more verbose output to validate processing, diagnose issues, or get detailed information on how the Historian works. You can also redirect logging to a file.

Controlling What is Logged

To control what information is logged, first open a browser, go to the URL for Documaker Administrator, and then log into the system.

Then follow these steps:

1. Click the Systems link. Expand Systems and choose the appropriate assembly line in which you want to create a task.
2. Expand the assembly line and click the Historian application. Click the Configure button, which will open a new tab.
3. Locate the Context- Category combination *LOG4J - logger*. Expand this selection.
4. Select the oracle.documaker.historian group. In the Properties panel, locate the Priority property and click the Value column. Select one of these options:

Option	Description
Error	Only error messages are output from the Historian and its tasks. This is the default setting.
Info	Informational and error messages are output from the Historian and its tasks. You can use this setting to output messages that indicate the ID numbers and counts of records processed by Historian tasks to use for validation.
Debug	Debugging information and error messages are output from the Historian and its tasks.

Click the Save icon to save your changes.

Selecting the Output Location

By default all logging output goes to the LOGS database table. To have the system output logging information to a file, first open a browser, go to the URL for Documaker Administrator, and then log into the system.

Then follow these steps:

1. Click the Systems link. Expand the Systems and choose the appropriate assembly line in which you want to create a task.
2. Expand the assembly line and click the Historian application. Click the Configure button, which will open a new tab.
3. Locate and expand the Log4J - Appender Context-Category combination. Then locate the process-roll group. Set these properties:

Property	Description
Property	Set this to **File**
Value	Set the value to be the location and name of the log file into which logging output is written. This should be a relative path under the docfactory/temp/historian path. Here is an example: `logs/Historian.log`

Click the Save icon to save your changes.

4. Locate and expand the Log4J - Appender Context-Category combination. Then locate the roll group. Set these properties:

Property	Description
Property	Set this to **File**
Value	Set the value to be the location and name of the log file into which logging output is written. This should be a relative path under the docfactory/temp/historian path. Here is an example: `logs/Historian.log`

Click the Save icon to save your changes.

Note If your configuration does not have the Log4J - Appender context-category combination, you can create it using the Create Context button and then enter the appropriate values.

USING THE CRONTRIGGER CLASS

The CronTrigger class is based on the scheduling capabilities of cron, which is a UNIX tool with powerful scheduling capabilities.

CronTrigger uses cron expressions, which create firing schedules such as the ones shown here:

- At 8:00am every Monday through Friday
- At 1:30am on the last Friday of every month

Note For more information about cron, see this web site:

http://www.quartz-scheduler.org/docs/tutorials/CronTrigger.html

Creating a cron Expression

A cron expression is a string comprised of six or seven fields separated by spaces. The fields can contain any of the allowed values, along with various combinations of the special characters allowed for that field.

Some cron expressions are as simple as this example:

```
* * * * ? *
```

While others are more complex, like this example:

```
0/5 14,18,3-39,52 * ? JAN,MAR,SEP MON-FRI 2002-2010
```

This table explains the various fields:

Field	Values allowed	Special characters allowed
Seconds	0-59	, - * /
Minutes	0-59	, - * /
Hours	0-23	, - * /
Day of Month	1-31	, - * ? / L W
Month	1-12 or JAN-DEC	, - * /
Day of Week	1-7 or SUN-SAT	, - * ? / L #
Year *	empty, 1970-2099	, - * /

* Year is the only optional field.

This table explains the special characters you can use:

Character	Description
* (asterisk)	Used to select all values within a field. For example, an asterisk (*) in the Minutes field means *every minute*.

Character	Description
? (question mark)	Used to specify something in one of the two fields in which the character is allowed, but not the other. For example, if you want a trigger to fire on a particular day of the month (say, the 10th), but do not care what day of the week that happens to be, you would enter *10* in the Day-of-Month field and *?* in the Day-of-Week field.
- (dash)	Used to specify ranges. For example, *10-12* in the Hour field means the hours 10, 11 and 12.
, (comma)	Used to specify additional values. For example, *MON,WED,FRI* in the Day-of-Week field means the days Monday, Wednesday, and Friday.
/ (slash)	Used to specify increments. For example, *0/15* in the Seconds field means the seconds 0, 15, 30, and 45. If you enter *5/15* in the Seconds field, it means the seconds 5, 20, 35, and 50. You can also enter a slash (/) after the quotation mark (“) — in this case the quotation mark is equivalent to having a zero (0) before the slash (/). If you enter *1/3* in the Day-of-Month field, it means fire every three days starting on the first day of the month.
L	Has different meaning, depending on the field in which it is entered. An *L* in the Day-of-Month field means the last day of the month which, for example, is day 31 for January and day 28 for February on non-leap years. You can also specify an offset from the last day of the month, such as *L-3* which would mean the third-to-last day of the calendar month. If used in the Day-of-Week field, it indicates the last day of the week (Saturday).If used in the Day-of-Week field after another value, it means the last (week) day of the month. For example, *6L* means the last Friday of the month. Note: When using the L option, do not specify lists or ranges of values or you will get unexpected results.
W	Used to specify the weekday (Monday-Friday) nearest to the given day. For example, if you specify *15W* in the Day-of-Month field, it means the nearest weekday to the 15th of the month. So if the 15th is a Saturday, the trigger fires on Friday the 14th. If the 15th is a Sunday, the trigger fires on Monday the 16th. If the 15th is a Tuesday, then it fires on Tuesday the 15th. If, however, you specify *1W* in the Day-of-Month field and the 1st is a Saturday, the trigger will fire on Monday the 3rd, as it is the first weekday in the new month. You can only include the W character when the Day-of-Month field indicates a single day and not a range or list of days. You can combine the L and W characters in the Day-of-Month field (LW) to indicate the last weekday of the month.
# (octothorpe)	Used to specify the *nth* day of the month. For example, *6#3* in the Day-of-Week field means the third Friday of the month (day *6* is Friday and *#3* means the 3rd one in the month). Here are some other examples: • *2#1* indicates the first Monday of the month • *4#5* indicates the fifth Wednesday of the month Note that if you specify 4#*5* and there are not five Wednesdays in the month, no firing occurs that month.

Note The characters and the names of months and days of the week are not case sensitive. For instance, *MON* is the same as *mon*, and both equal Monday.

Here are some examples. Note how the question mark (?) and asterisk (*) affect the Day-of-Week and Day-of-Month fields.

This example	Tells the system to fire at...
`0 0 12 * * ?`	12pm (noon) every day
`0 15 10 ? * *`	10:15am every day
`0 15 10 * * ?`	10:15am every day
`0 15 10 * * ? *`	10:15am every day
`0 15 10 * * ? 2005`	10:15am every day during the year 2005
`0 * 14 * * ?`	Every minute starting at 2pm and ending at 2:59pm, every day
`0 0/5 14 * * ?`	Every five minutes starting at 2pm and ending at 2:55pm, every day
`0 0/5 14,18 * * ?`	Every five minutes starting at 2pm and ending at 2:55pm *and* fire every five minutes starting at 6pm and ending at 6:55pm, every day
`0 0-5 14 * * ?`	Every minute starting at 2pm and ending at 2:05pm, every day
`0 10,44 14 ? 3 WED`	2:10pm and at 2:44pm every Wednesday in the month of March
`0 15 10 ? * MON-FRI`	10:15am every Monday, Tuesday, Wednesday, Thursday, and Friday
`0 15 10 15 * ?`	10:15am on the 15th day of every month
`0 15 10 L * ?`	10:15am on the last day of every month
`0 15 10 L-2 * ?`	10:15am on the 2nd-to-last day of every month
`0 15 10 ? * 6L`	10:15am on the last Friday of every month
`0 15 10 ? * 6L 2002-2005`	10:15am on every last friday of every month during the years 2002, 2003, 2004, and 2005
`0 15 10 ? * 6#3`	10:15am on the third Friday of every month
`0 0 12 1/5 * ?`	12pm (noon) every five days every month, starting on the first day of the month
`0 11 11 11 11 ?`	Every November 11th at 11:11am

Keep in mind...

- You must use the ? character in at least one of these fields:
  - Day of Week
  - Day of Month
- Be careful when setting firing times between the hours of the morning when daylight savings changes occur in your locale. For US locales, this would typically be the hour before and after 2:00am. The time shift can cause a skip or a repeat, depending on whether the time moves back or jumps forward.

STARTING AND STOPPING THE HISTORIAN

This table shows you how to start, stop, and make sure the Historian is running.

To	Then
Start the Historian	Place the historian.jar file in the deploy directory of Document Factory.
Stop the Historian	Remove the historian.jar file from the deploy directory of Document Factory.
Verify the Historian is running	Verify that the docfactory_historian process is running.

Note The historian.jar configuration file is uncompressed and deployed to the temp\historian directory. This directory becomes the working directory for the Historian. Any output, including Log4J output, uses this directory as the starting directory.

Chapter 4

Logging to the Database

Document Factory uses the Log4J API to log diagnostic and error information for each process. Log4j is a Java logging or tracing API. This chapter discusses the following topics:

For more information about Log4J, visit this web site:

http://logging.apache.org/log4j/

OVERVIEW

Logging information for Documaker Interactive is stored in the log4j.xml file in the oracle.idocumaker.ids.bc.jar, if found within the idm.war. This WAR file is contained in the idm.ear file that is installed by Oracle Documaker Enterprise Edition.

You define what information is sent to the LOGS table in the Assembly Line schema as well as the WebLogic (or other web application) console, and the ccmdebug.log file. using the Category's Priority property.

Option	Description
Error	Sends messages that note events that cannot be processed *and* stop Documaker Interactive from running. This is the default setting.
Debug	Sends debugging information to the LOGS table. To enable debug logging for Documaker Interactive to be written to the LOGS table in the Assembly Line schema, you must enable the ErrDBAppender reference within the Category. The ErrDBAppender includes a filter to provide finer control over the messages written to the LOGS table. This filter applies to all Categories that reference the ErrDBAppender.
Warn	Sends messages that note events that cannot be processed *but do not* stop Documaker Interactive from running.

Note For most issues, approach debugging Documaker Interactive by setting the oracle.documaker Category to Debug and reviewing the ccmdebug.log file.

LOGGING FILTERS

Document Factory uses log filters to determine which Log4J log statements are written to the database and which ones are written to the file system. Each process in an assembly line reads the *LogFilter* entries from the ALCONFIGCONTEXT table to determine which log statements are written to the database.

The filter entries in the ALCONFIGCONTEXT table provide the package names that should be logged to the database. Package names that are not included in the filter entries are logged to the file system in the docfactory/temp/*ProcessName* directory, where *ProcessName* is the name of the process running under the Document Factory Supervisor process (see the docfactory/temp directory).

Package names that match the ones in the filters are logged to the Logs or Errs tables. Warning, Debug, and Information log statements are written to the Logs table while Error and Fatal log statements are written to the Errs table.

Here is a list of log filters in the ALCONFIGCONTEXT table where the...

- Context_Name column value is *LOG4J*
- Category column value is *LogFilter*
- Group_Name column value is *LogFilter*

The Property and Notes columns are shown here:

Property	Notes
oracle.documaker.archiver	The Java package name for the Archiver process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.batch	The Java package name for the Batcher process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.historian	The Java package name for the Historian process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.identifier	The Java package name for the Identifier process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.receiver	The Java package name for the Receiver process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.scheduler	The Java package name for the Scheduler process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.process	The Java package name for the ProcessShell class.Used by all Java processes running under the Supervisor process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.processmonitor	The Java package name for the Supervisor process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.publishing	The Java package name for the Publisher process. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.PubNotifier	The Java package name for the PubNotifier process. Diagnostic and error messages for this package go to the Logs and Errs tables.

Property	Notes
oracle.documaker.rp	The Java package name for the code used through the Java Native Interface (JNI) by the Assembler, Distributor, and Presenter processes. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.na	The Java package name for the NA/POL loader code used through the JNI by the Assembler, Distributor, and Presenter processes. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.pol	The Java package name for the NA/POL loader code used through the JNI by the Assembler, Distributor, and Presenter processes. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.napol	The Java package name for the NA/POL loader code used through the JNI by the Assembler, Distributor, and Presenter processes. Diagnostic and error messages for this package go to the Logs and Errs tables.
oracle.documaker.NaPol	The Java package name for the NA/POL loader code used through the JNI by the Assembler, Distributor, and Presenter processes. Diagnostic and error messages for this package go to the Logs and Errs tables.
all	A global option for enabling the logging of all Java packages to the database. This option should only be enabled for debugging. Use caution when enabling this option as the amount of log statements can be overwhelming depending on which Log4J loggers are enabled at each process level. Also, keep in mind that enabling this option may also cause sensitive information, such as queue names and IP addresses, to be logged to the database. You can enable this option by setting the ACTIVE column value to one (1) in the ALCONFIGCONTEXT table. The default for the ACTIVE column is zero (0).

DEFINING LOG4J CONFIGURATION OPTIONS

The following Log4J configuration options are read from the APPCONFIGCONTEXT table when the...

- Context_Name column value is *LOG4J*
- Category column value is *Logger*

The Group_Name, Property, and Value columns are shown:

Group_Name	Property	Value
oracle.documaker	priority	error
oracle.documaker.util.Manifest	priority	error
oracle.documaker.util.PlatformSafe	priority	error
oracle.documaker.log4j	priority	error
oracle.documaker.db.DataSourceUtil	priority	error
oracle.documaker.config.jpa.JPAConfigurationFactory	priority	error
oracle.documaker.config.jpa.JPAConfiguration	priority	error
oracle.documaker.config.db.DataSourceConfigurationFactory	priority	error
oracle.documaker.config.db.DataSourceConfiguration	priority	error
oracle.documaker.config.xml.XMLConfigurationFactory	priority	error
oracle.documaker.config.xml.XMLConfiguration	priority	error
oracle.documaker.config.AbstractConfiguration	priority	error
oracle.documaker.config.ConfigurationUtil	priority	error
oracle.documaker.db.documaker.DocumakerDataSourceFactory	priority	error
oracle.documaker.db.jndi.JNDIDataSourceFactory	priority	error
oracle.documaker.db.Query	priority	error
oracle.documaker.db.SQLData	priority	error
oracle.documaker.dao.DAOUtil	priority	error
oracle.documaker.dao.AbstractDAO	priority	error
oracle.documaker.dao.AbstractDAO.Timer	priority	error
oracle.documaker.dao.tables.jobs.LockDAO	priority	error
oracle.documaker.dao.tables.jobs.JobDAO	priority	error
oracle.documaker.dao.tables.jobs.JOBS	priority	error
oracle.documaker.dao.tables.jobs.JOBS.Exception	priority	error

Group_Name	Property	Value
oracle.documaker.dao.tables.jobs.JOBS.Extract	priority	error
oracle.documaker.dao.tables.trns.TrnDAO	priority	error
oracle.documaker.dao.tables.trns.TRNS	priority	error
oracle.documaker.dao.tables.rcps.RcpDAO	priority	error
oracle.documaker.dao.tables.trns.RCPS	priority	error
oracle.documaker.dao.tables.bchs.BchDAO	priority	error
oracle.documaker.dao.tables.bchs.BCHS	priority	error
oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	priority	error
oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	priority	error
oracle.documaker.dao.tables.pubs.PubDAO	priority	error
oracle.documaker.dao.tables.pubs.PUBS	priority	error
http.debug	priority	error
mqseries.debug	priority	error
msmq.debug	priority	error
jms.debug	priority	error
oracle.documaker.bus	priority	error
ProcessMonitor.output	priority	error
oracle.documaker.processmonitor.ProcessMonitor	priority	error
oracle.documaker.processmonitor.monitors.SelfMonitor	priority	error
oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	priority	error
oracle.documaker.processmonitor.monitors.FileMonitor	priority	error
oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	priority	error
oracle.documaker.processmonitor.loadbalancing.LoadBalancer	priority	error
oracle.documaker.processmonitor.deployment.HotDeployer	priority	error
oracle.documaker.processmonitor.deployment.DeployWorker	priority	error
oracle.documaker.processmonitor.process.Process	priority	error
oracle.documaker.processmonitor.process.data.ProcessData	priority	error
oracle.documaker.processmonitor.process.monitors.InstanceMonitor	priority	error
oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	priority	error

Group_Name	Property	Value
oracle.documaker.processmonitor.process.instance.Instance	priority	error
oracle.documaker.processmonitor.ipc.PipeReader	priority	error
oracle.documaker.processmonitor.ipc.PipeWriter	priority	error
com.docucorp.jnative	priority	error
EMAIL	priority	error
LogLogger	priority	error
ErrorLogger	priority	error
root	priority	error

Note Change the Value column value from *error* to *debug* for any of the previous Log4J Loggers to enable logging. Reference the loggers in *Overview* on page 274 for descriptions of what each logger does and its additional Log4J configuration options.

CONFIGURING THE LOG4J APPENDERS

Document Factory uses Log4J appenders to write log statements to different destinations. The appenders are defined in the ALCONFIGCONTEXT table so they can be shared across all processes in the same assembly line. The appenders can also be defined at the application level in APPCONFIGCONTEXT table, in which case they override the values provided in the ALCONFIGCONTEXT table.

Here is a list of the appenders defined in the ALCONFIGCONTEXT table, where the...

- Context_Name column value is *LOG4J*
- Category column value is *Appenders*
- Group_Name column value is *Appender*

The Property, Value, and Notes columns are shown here:

Property	Value	Notes
name	stdout	The name of the appender that logs the Log4J statement to STDOUT (standard output).
name	roll	The appender used to log statements to the file system.
name	process-roll	The appender used to log statements to the file system. This appender is sometimes used instead of roll appender to log process specific messages to a different file system destination.
name	LogAppender	The appender used to log Info, Warn and Debug level Log4J statements to the Logs table.
name	ErrorAppender	The appender used to log Error and Fatal level Log4J statements to the Errs table.
name	EMAIL	The email appender used to send email notifications for process critical error messages.

Here is a list of the appenders and their configuration options defined in the ALCONFIGCONTEXT table, where the...

- Context_Name column value is *LOG4J*
- Category column value is *Appender*
- Group_Name column value is the value for each appender in the previous table

The Group_Name, Property, Value, and Notes columns are shown here:

Group_Name	Property	Value	Notes
stdout	class	org.apache.log4j.ConsoleAppender	The STDOUT (standard output) appender fully-qualified class name.
stdout	LayoutClass	org.apache.log4j.PatternLayout	The pattern layout fully-qualified class name.

Group_Name	Property	Value	Notes
stdout	ConversionPattern	%d{ISO8601}-%5p-[%t]-%C.%M: %m%n	The conversion pattern used to write each Log4J statement to STDOUT.
roll	class	oracle.documaker.log4j.appender.DocumakerRollingFileAppender	The rolling file appender fully-qualified class name.
roll	File	logs/~THREADID.log	The file to write the Log4J statements to, where ~*THREADID* is replaced at run time by the current thread ID writing the Log4J statement.
roll	Encoding	ISO-8859-1	The character encoding to use when writing the Log4J statements.
roll	MaxFileSize	100MB	The maximum file size for each file that belongs to this rolling file appender. When a file exceeds the size it is rolled as the previous to last file and a new file is created. The system uses the MaxFileSize and MaxBackupIndex options to avoid running out of disk space and uncontrolled logging.
roll	MaxBackupIndex	5	The maximum number of files to keep for this rolling file appender. The system maintains up to *MaxBackupIndex* files for this appender rolling the files to keep the latest set of files on disk. The system uses the MaxFileSize and MaxBackupIndex options to avoid running out of disk space and uncontrolled logging.
roll	LayoutClass	org.apache.log4j.PatternLayout	The pattern layout fully-qualified class name.
roll	ConversionPattern	%d{ISO8601}-%5p-[%t]-[%F:%L]-%C.%M: %m%n	The conversion pattern used to write each Log4J statement to the file system.
process-roll	class	oracle.documaker.log4j.appender.DocumakerRollingFileAppender	The rolling file appender fully-qualified class name.
process-roll	File	logs/~PROGRAM.log	The file to write the Log4J statements to, where ~*PROGRAM* is replaced at run time by the program name of the process writing the Log4J statement.
process-roll	Encoding	ISO-8859-1	The character encoding to use when writing the Log4J statement

Group_Name	Property	Value	Notes
process-roll	MaxFileSize	100MB	The maximum file size for each file that belongs to this rolling file appender. When a file exceeds the size it is rolled as the previous to last file and a new file is created. The system uses the MaxFileSize and MaxBackupIndex options to avoid running out of disk space and uncontrolled logging.
process-roll	MaxBackupIndex	5	The maximum number of files to keep for this rolling file appender. The system maintains up to *MaxBackupIndex* files for this appender rolling the files to keep the latest set of files on disk. By using MaxFileSize with MaxBackupIndex options, the system avoids uncontrolled logging and running out disk space.
process-roll	LayoutClass	org.apache.log4j.PatternLayout	The pattern layout fully-qualified class name.
process-roll	ConversionPattern	%d{ISO8601}-%5p-[%t]-[%F:%L]-%C.%M:%m%n	The conversion pattern used to write each Log4J statement to the file system.
LogAppender	class	oracle.documaker.log4j.appender.jdbc.DFAppender	The appender used to write Log4J statements to the Logs database table.
LogAppender	LayoutClass	oracle.documaker.log4j.layout.DocumakerPatternLayout	The pattern layout fully-qualified class name.
LogAppender	ConversionPattern	insert into LOGS (LOGTIME, LOGHOSTNAME, LOGPROGRAM, LOGVERSION, LOGMODULE, LOGTHREAD_ID, LOGPROCESS_ID, LOGCATEGORY, LOGMESSAGE) values (%d, %H, %P, %V, %Y, %T, %I, %L, %m)	The conversion pattern/JDBC statement used to write Log4J statements to the Logs database table.
LogAppender	FilterClass	org.apache.log4j.varia.LevelRangeFilter	The level range filter fully-qualified class name. This class is used to filter Log4J statements.
LogAppender	LevelMin	debug	The minimum Log4J level accepted by the filter for this appender. Log4J statements that do not meet this filter criteria are not logged.

Group_Name	Property	Value	Notes
LogAppender	LevelMax	warn	The maximum Log4J level accepted by the filter for this appender. Log4J statements that do not meet this filter criteria are not logged.
ErrorAppender	class	oracle.documaker.log4j.appender.jdbc.DFAppender	The appender used to write Log4J statements to the Errs database table.
ErrorAppender	LayoutClass	oracle.documaker.log4j.layout.DocumakerPatternLayout	The pattern layout fully-qualified class name.
ErrorAppender	ConversionPattern	insert into ERRS (ERRTIME, ERRHOSTNAME, ERRPROGRAM, ERRVERSION, ERRMODULE, ERRTHREAD_ID, ERRPROCESS_ID, ERRCATEGORY, ERRMESSAGE) values (%d, %H, %P, %V, %Y, %T, %I, %L, %m)	The conversion pattern/JDBC statement used to write Log4J statements to the Errs database table.
ErrorAppender	FilterClass	org.apache.log4j.varia.LevelRangeFilter	The level range filter fully-qualified class name. This class is used to filter Log4J statements.
ErrorAppender	LevelMin	error	The minimum Log4J level accepted by the filter for this appender. Log4J statements that do not meet this filter criteria are not logged.
ErrorAppender	LevelMax	fatal	The maximum Log4J level accepted by the filter for this appender. Log4J statements that do not meet this filter criteria are not logged.
EMAIL	class	org.apache.log4j.net.SMTPAppender	The fully-qualified class name of the email appender
EMAIL	BufferSize	1	The buffer size that dictates how many messages can be in the internal queue before being flushed/written to the destination. Set this value to one (1) to flush messages immediately.
EMAIL	SMTPHost	127.0.0.1	The IP address or host name of the email server.
EMAIL	SMTPUserName	null	The user name for authentication against the email server. Can be left blank if the server supports anonymous authentication

Group_Name	Property	Value	Notes
EMAIL	SMTPPassword	null	The password for authentication against the email server. You can leave this blank if the server supports anonymous authentication.
EMAIL	From	docfactory@oracle.com	The email address used to send the emails.
EMAIL	To	doc.factory@oracle.com	A comma-delimited list of email addresses that will receive the email notifications.
EMAIL	cc	null	A comma-delimited list of email addresses that will receive a carbon copy of email notifications.
EMAIL	bcc	null	A comma-delimited list of email addresses that will receive a blind carbon copy of email notifications.
EMAIL	Subject	Document Factory Error Message	The subject to use for all email notifications.
EMAIL	ThreshHold	error	The ThreshHold level that filters email notifications. Set the value to error to log only error or fatal error messages.
EMAIL	LayoutClass	org.apache.log4j.PatternLayout	The pattern layout fully-qualified class name for this appender.
EMAIL	ConversionPattern	%d{ISO8601} %-5p [%t] - %m\r\n	The conversion pattern this appender uses to write Log4J statements.

CONFIGURING THE LOG4J LOGGERS

Document Factory uses Log4J loggers to write log statements for different packages and class names. The loggers are defined in the ALCONFIGCONTEXT table so they can be shared across all processes in the same assembly line. They can, however, also be defined at the application level in APPCONFIGCONTEXT table.

Note Values in the APPCONFIGCONTEXT table override values in the ALCONFIGCONTEXT table.

Here is a list of the loggers defined in the ALCONFIGCONTEXT table, where the...

- Context_Name column value is *LOG4J*
- Category column value is *Loggers*
- Group_Name column value is *Logger*

The Property, Value, and Notes columns are shown here:

Property	Value	Notes
name	oracle.documaker	The default logger used when no other logger can be found for a class. Can be used to log diagnostic or error information.
name	oracle.documaker.util.Manifest	Logs diagnostic and error information for the Manifest class in the Documaker-Util package when parsing the MANIFEST.MF file inside a deployment JAR file during start up of a program.
name	oracle.documaker.util.PlatformSafe	Logs diagnostic and error information for the PlatformSafe class in the Documaker-Util package when converting paths from Windows to UNIX and back.
name	oracle.documaker.log4j	Logs diagnostic and error information for the custom JDBCApppender class in the Documaker-Log4J package when logging messages to the Logs and Errs tables in the database.
name	oracle.documaker.db.DataSourceUtil	Logs diagnostic and error information for the DataSourceUtil class in the Documaker-DB package when retrieving a .bindings file or a JNDI data source.
name	oracle.documaker.config.jpa.JPAConfigurationFactory	Logs diagnostic and error information for the JPAConfigurationFactory class in the Documaker-Config package when creating a JPAConfiguration object to read configuration information from a database.
name	oracle.documaker.config.jpa.JPAConfiguration	Logs diagnostic and error information for the JPAConfiguration class in the Documaker-Config package when retrieving configuration properties from one of the *CONFIGCONTEXT database tables.

Property	Value	Notes
name	oracle.documaker.config.db.DataSourceConfigurationFactory	Logs diagnostic and error information for the DataSourceConfigurationFactory class in the Documaker-Config package when creating a DataSourceConfiguration object to read configuration information from a database.
name	oracle.documaker.config.db.DataSourceConfiguration	Logs diagnostic and error information for the DataSourceConfiguration class in the Documaker-Config package when retrieving configuration properties from one of the *CONFIGCONTEXT database tables.
name	oracle.documaker.config.xml.XMLConfigurationFactory	Logs diagnostic and error information for the XMLConfigurationFactory class in the Documaker-Config package when creating an XMLConfiguration object to store and retrieve configuration properties to/from XML.
name	oracle.documaker.config.xml.XMLConfiguration	Logs diagnostic and error information for the XMLConfiguration class in the Documaker-Config package when retrieving configuration properties from an XML configuration file.
name	oracle.documaker.config.AbstractConfiguration	Logs diagnostic and error information for the AbstractConfiguration class in the Documaker-Config package used by the JPAConfiguration, DataSourceConfiguration and XMLConfiguration classes.
name	oracle.documaker.config.ConfigurationUtil	Logs diagnostic and error information for the ConfigurationUtil class in the Documaker-Config package when retrieving a Configuration object.
name	oracle.documaker.db.documaker.DocumakerDataSourceFactory	Logs diagnostic and error information for the DocumakerDataSourceFactory class in the Documaker-DB package when it creates instances of the DocumakerDataSource class which extends the BasicDataSource class from Apache.
name	oracle.documaker.db.jndi.JNDIDataSourceFactory	Logs diagnostic and error information for the JNDIDataSourceFactory class in the Documaker-DB package when it returns a JNDI - Java Naming and Directory Interface data source.
name	oracle.documaker.db.Query	Logs diagnostic and error information for the Query class in the Documaker-DB package when it returns connection information, table column metadata, and SQL queries.
name	oracle.documaker.db.SQLData	Logs diagnostic and error information for the SQLData class in the Documaker-DB package which shows the data being passed in or returned for an SQL query.
name	oracle.documaker.dao.DAOUtil	Logs diagnostic and error information for the DAOUtil class in the Documaker-DAO package when it creates a DAO instance through reflection.

Property	Value	Notes
name	oracle.documaker.dao.AbstractDAO	Logs diagnostic and error information for the AbstractDAO class in the Documaker-DAO package when it performs SQL queries to provide the base functionality for all other DAO classes.
name	oracle.documaker.dao.AbstractDAO.Timer	Logs start and completion times for the different SQL queries in AbstractDAO class in the Documaker-DAO package. Useful in capturing times for JDBC operations.
name	oracle.documaker.dao.tables.jobs.LockDAO	Logs diagnostic and error information for the LockDAO class in the Documaker-DAO package as it performs different table lock operations for program synchronization.
name	oracle.documaker.dao.tables.jobs.JobDAO	Logs diagnostic and error information for the JobDAO class in the Documaker-DAO package. This class is used when interfacing with the JOBS table.
name	oracle.documaker.dao.tables.jobs.JOBS	Logs diagnostic and error information for the JOBS class in the Documaker-DAO package. Useful when troubleshooting marshalling/unmarshalling operations for job related objects.
name	oracle.documaker.dao.tables.jobs.JOBS.Exception	Logs XML parsing exceptions for the JOBS class in the Documaker-DAO package. Useful when troubleshooting XML parsing operations for job related objects.
name	oracle.documaker.dao.tables.jobs.JOBS.Extract	Logs input parsing information for the JOBS class in the Documaker-DAO package. Useful when troubleshooting XML/Text parsing operations for job related objects. Enabling this logger displays the content of each XML/Text transaction being parsed.
name	oracle.documaker.dao.tables.trns.TrnDAO	Logs diagnostic and error information for the TrnDAO class in the Documaker-DAO package. Useful when troubleshooting SQL operations against the TRNS table.
name	oracle.documaker.dao.tables.trns.TRNS	Logs diagnostic and error information for the TRNS class in the Documaker-DAO package. Useful when troubleshooting marshalling/unmarshalling operations for transaction related objects.
name	oracle.documaker.dao.tables.rcps.RcpDAO	Logs diagnostic and error information for the RcpDAO class in the Documaker-DAO package. Useful when troubleshooting SQL operations against the RCPS table.
name	oracle.documaker.dao.tables.trns.RCPS	Logs diagnostic and error information for the RCPS class in the Documaker-DAO package. Useful when troubleshooting marshalling/unmarshalling operations for recipient related objects.
name	oracle.documaker.dao.tables.bchs.BchDAO	Logs diagnostic and error information for the BchDAO class in the Documaker-DAO package. Useful when troubleshooting SQL operations against the BCHS table.

Property	Value	Notes
name	oracle.documaker.dao.tables.bchs.BCHS	Logs diagnostic and error information for the BCHS class in the Documaker-DAO package. Useful when troubleshooting marshalling/unmarshalling operations for batch related objects.
name	oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	Logs diagnostic and error information for the RcpBchDAO class in the Documaker-DAO package. Useful when troubleshooting SQL operations against the BCHS_RCPS table.
name	oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	Logs diagnostic and error information for the RCPS_BCHS class in the Documaker-DAO package. Useful when troubleshooting marshalling/ unmarshalling operations for batch/recipient objects.
name	oracle.documaker.dao.tables.pubs.PubDAO	Logs diagnostic and error information for the PubDAO class in the Documaker-DAO package. Useful when troubleshooting SQL operations against the PUBS table.
name	oracle.documaker.dao.tables.pubs.PUBS	Logs diagnostic and error information for the PUBS class in the Documaker-DAO package. Useful when troubleshooting marshalling/unmarshalling operations for publishing objects.
name	oracle.documaker.db.documaker.DBUtil	Logs diagnostic and error information for the DBUtil class in the Documaker-DB package when returning table column metadata.
name	http.debug	Logs diagnostic and error information for the HTTP message bus.
name	mqseries.debug	Logs diagnostic and error information for the IBM WebSphere MQ message bus.
name	msmq.debug	Logs diagnostic and error information for the Microsoft MSMQ message bus.
name	jms.debug	Logs diagnostic and error information for the JMS message bus.
name	oracle.documaker.bus	Logs diagnostic and error information for the Documaker-BUS package when performing message bus operations.
name	oracle.documaker.process.ProcessShell	Logs diagnostic and error information for the Documaker-Process package; used by all Java processes running under the Document Factory.
name	oracle.documaker.process.exception.ExceptionHandler	Logs diagnostic and error information for the ExceptionHandler class in the Documaker-Process package when handling an unhandled exception. This class catches any unexpected Throwables a Java process may throw.
name	oracle.documaker.process.util.ProcessUtil	Logs diagnostic and error information for the ProcessUtil class in the Documaker-Process package when it retrieves JVM options and process information such as a process ID.

Property	Value	Notes
name	oracle.documaker.process.ipc.IPCConnector	Logs diagnostic and error information for the IPCConnector class in the Documaker-Process package as it communicates with the Supervisor program via named pipes. This class is responsible for the inter-process communication between a Java process and the Supervisor process.
name	oracle.documaker.process.ipc.PipeReader	Logs diagnostic and error information for the PipeReader class in the Documaker-Process package as it reads messages from the Supervisor program via an input named pipe.
name	oracle.documaker.process.ipc.PipeWriter	Logs diagnostic and error information for the PipeWriter class in the Documaker-Process package as it writes messages for the Supervisor program via an output named pipe.
name	com.oracle.npc	Logs diagnostic and error information for the JNI npc class in the Documaker-Process package used to instantiate a native named pipe object.
name	oracle.documaker.process.ipc.IPCThread	Logs diagnostic and error information for the IPCThread thread class in the Documaker-Process package that runs periodically reading and writing messages from and to the input and output named pipes.
name	oracle.documaker.process.monitors.DataSourceMonitor	Logs diagnostic and error information for the DataSourceMonitor thread class in the Documaker-Process package as it periodically monitors the health of a data source for a Java process.
name	oracle.documaker.process.monitors.WorkerMonitor	Logs diagnostic and error information for the WorkerMonitor thread class in the Documaker-Process package as it periodically monitors the health of all the workers in a process.
name	oracle.documaker.process.monitors.Log4jMonitor	Logs diagnostic and error information for the Log4jMonitor thread class in the Documaker-Process package as it periodically monitors the log4j configuration in case that it may need to reloaded.
name	oracle.documaker.process.monitors.FileMonitor	Logs diagnostic and error information for the FileMonitor thread class in the Documaker-Process package as it periodically monitors file resources for changes indicating a process needs to be restarted.
name	oracle.documaker.process.worker.Worker	Logs diagnostic and error information for the Worker thread class in the Documaker-Process package.
name	root	The root fall-back logger for a Log4J configuration.
name	ProcessMonitor.output	Logs start up and shut down messages for the Supervisor.
name	oracle.documaker.processmonitor.ProcessMonitor	Logs diagnostic and error information for the ProcessMonitor thread class in the Documaker-ProcessMonitor package as it starts up and monitors the different processes in a Document Factory.

Property	Value	Notes
name	oracle.documaker.processmonitor.monitors.SelfMonitor	Logs diagnostic and error information for the SelfMonitor thread class in the Documaker-ProcessMonitor package as it periodically monitors the health of the Supervisor program.
name	oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	Logs diagnostic and error information for the SelfLog4jMonitor thread class in the Documaker-ProcessMonitor package as it periodically monitors the log4j configuration in case that it needs to be reloaded.
name	oracle.documaker.processmonitor.monitors.FileMonitor	Logs diagnostic and error information for the FileMonitor thread class in the Documaker-ProcessMonitor package as it periodically monitors file resources for changes indicating the Supervisor needs to be restarted.
name	oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	Logs diagnostic and error information for the DBConfigurationMonitor thread class in the Documaker-ProcessMonitor package as it periodically monitors the database configuration tables for changes in a process configuration indicating a process needs to be restarted.
name	oracle.documaker.processmonitor.loadbalancing.LoadBalancer	Logs diagnostic and error information for the LoadBalancer thread class in the Documaker-ProcessMonitor package as it periodically monitors the instances for a process for the purpose of load balancing.
name	oracle.documaker.processmonitor.deployment.HotDeployer	Logs diagnostic and error information for the HotDeployer thread class in the Documaker-ProcessMonitor package as it periodically monitors the deploy subdirectory to deploy, undeploy, and redeploy a process.
name	oracle.documaker.processmonitor.deployment.DeployWorker	Logs diagnostic and error information for the DeployWorker thread class in the Documaker-ProcessMonitor package as it deploys, undeploys, and redploys a process.
name	oracle.documaker.processmonitor.process.Process	Logs diagnostic and error information for the Process class in the Documaker-ProcessMonitor package as it starts and shuts down a process.
name	oracle.documaker.processmonitor.process.data.ProcessData	Logs diagnostic and error information for the ProcessData class in the Documaker-ProcessMonitor package as it retrieves configuration information for a process.
name	oracle.documaker.processmonitor.process.monitors.InstanceMonitor	Logs diagnostic and error information for the InstanceMonitor thread class in the Documaker-ProcessMonitor package as it periodically monitors the health of a process instance.
name	oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	Logs diagnostic and error information for the InstanceMonitor thread class in the Documaker-ProcessMonitor package as it restarts a process instance.

Property	Value	Notes
name	oracle.documaker.processmonitor.process.instance.Instance	Logs diagnostic and error information for the Instance class in the Documaker-ProcessMonitor package. This class represents a process instance for a process - a process can have more than one instance. This class is used to start, restart and stop a process instance.
name	oracle.documaker.processmonitor.ipc.PipeReader	Logs diagnostic and error information for the PipeReader class in the Documaker-ProcessMonitor package as it reads messages from a process instance via an input named pipe.
name	oracle.documaker.processmonitor.ipc.PipeWriter	Logs diagnostic and error information for the PipeWriter class in the Documaker-ProcessMonitor package as it writes messages to a process instance via an output named pipe.
name	com.docucorp.jnative	Logs diagnostic and error information for the JNDI jnative class in Documaker-ProcessMonitor package as it installs UNIX signal handlers during start up of a process when running on a UNIX environment.
name	EMAIL	The email logger used by the Supervisor to send diagnostic and error messages when a process instance encounters a fatal error.
name	oracle.documaker.scheduler.Scheduler	Logs diagnostic and error information for the Scheduler thread class in the Documaker-Scheduler package. This is the main entry point class for the Scheduler program.
name	oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper	Logs diagnostic and error information for the SchedulerHouseKeeper thread class in the Documaker-Scheduler package. This class does all house keeping and clean up for the Scheduler program such as checking for unchanged acknowledgement status codes and setting them to error codes after a timeout interval elapses.
name	oracle.documaker.scheduler.shutdown.SchedulerShutdownHook	Logs diagnostic and error information for the SchedulerShutdownHook thread class in the Documaker-Scheduler package as it performs any shut down clean up for the Scheduler program such as closing queue and database connections.
name	oracle.documaker.scheduler.monitors.NotifyIdentifier	Logs diagnostic and error information for the NotifyIdentifier thread class in the Documaker-Scheduler package. This class is responsible for notifying the Identifier program that there is work to be done.
name	oracle.documaker.scheduler.monitors.NotifyAssembler	Logs diagnostic and error information for the NotifyAssembler thread class in the Documaker-Scheduler package. This class is responsible for notifying the Assembler program that there is work to be done.

Property	Value	Notes
name	oracle.documaker.scheduler.monitors.NotifyDistributor	Logs diagnostic and error information for the NotifyDistributor thread class in the Documaker-Scheduler package. This class is responsible for notifying the Distributor program that there is work to be done.
name	oracle.documaker.scheduler.monitors.NotifyPresenterImmediate	Logs diagnostic and error information for the NotifyPresenterImmediate thread class in the Documaker-Scheduler package. This class is responsible for notifying the Presenter program about immediate print transactions awaiting processing.
name	oracle.documaker.scheduler.monitors.NotifyPresenterScheduled	Logs diagnostic and error information for the NotifyPresenterScheduled thread class in the Documaker-Scheduler package. This class is responsible for notifying the Presenter program about scheduled print transactions awaiting processing.
name	oracle.documaker.scheduler.monitors.NotifyArchiver	Logs diagnostic and error information for the NotifyArchiver thread class in the Documaker-Scheduler package. This class is responsible for notifying the Archiver program that there is work to be done.
name	oracle.documaker.scheduler.monitors.NotifyPublisher	Logs diagnostic and error information for the NotifyPublisher thread class in the Documaker-Scheduler package. This class is responsible for notifying the Publisher program that there is work to be done.
name	oracle.documaker.scheduler.monitors.NotifyPubNotifier	Logs diagnostic and error information for the NotifyPubNotifier thread class in the Documaker-Scheduler package. This class is responsible for notifying the PubNotifier program that there is work to be done.
name	oracle.documaker.receiver.Receiver	Logs diagnostic and error information for the Receiver thread class in the Documaker-Receiver package. This is the main entry point class for the Receiver program.
name	oracle.documaker.receiver.shutdown.ReceiverShutdownHook	Logs diagnostic and error information for the ReceiverShutdownHook thread class in the Documaker-Receiver package as it performs any shut down clean up for the Receiver program such as closing queue and database connections.
name	oracle.documaker.receiver.monitors.FileReceiver	Logs diagnostic and error information for the FileReceiver thread class in the Documaker-Receiver package. This is the thread that monitors the hot directories for input files that should parsed to insert records in the Jobs table.
name	oracle.documaker.receiver.monitors.QueueReceiver	Logs diagnostic and error information for the QueueReceiver thread class in the Documaker-Receiver package. This is the thread that monitors the receiver request queue for jobs that need to be parsed and inserted in the Jobs table.

Property	Value	Notes
name	oracle.documaker.receiver.monitors.QueueReceiverWorker	Logs diagnostic and error information for the QueueReceiverWorker thread class in the Documaker-Receiver package. This is the thread that inserts a job in the Jobs table and returns the print streams for it back to DWS doPublishFromImport web service operation.
name	oracle.documaker.identifier.Identifier	Logs diagnostic and error information for the Identifier thread class in the Documaker-Identifier package. This is the main entry point class for the Identifier program.
name	oracle.documaker.identifier.shutdown.IdentifierShutdownHook	Logs diagnostic and error information for the IdentifierShutdownHook thread class in the Documaker-Identifier package as it performs any shut down clean up for the Identifier program such as closing queue and database connections.
name	LogLogger	Logs Info, Warn and Debug level Log4J messages to the Logs database table for the loggers specified in the LogFilter entries in ALCONFIGCONTEXT table.
name	ErrorLogger	Logs Error and Fatal level Log4J messages to the Errs database table for the loggers specified in the LogFilter entries in ALCONFIGCONTEXT table.
name	oracle.documaker.rp.jdbc.GenericDAO	Logs diagnostic and error information for the GenericDAO class in the Documaker-RP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to interface with the database tables.
name	oracle.documaker.rp.jdbc.DAO	Logs diagnostic and error information for the DAO class in the Documaker-RP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to interface with the database tables.
name	oracle.documaker.rp.bus.Bus	Logs diagnostic and error information for the Bus class in the Documaker-RP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to retrieve/put messages from/to a message bus.
name	oracle.documaker.rp.config.Configuration	Logs diagnostic and error information for the Configuration class in the Documaker-RP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to retrieve configuration information from the *CONFIGCONTEXT tables.
name	oracle.documaker.na.Loader	Logs diagnostic and error information for the na.Loader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to load NA information for a transaction.

Property	Value	Notes
name	oracle.documaker.na.Unloader	Logs diagnostic and error information for the na.Unloader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to retrieve NA information for a transaction.
name	oracle.documaker.pol.Loader	Logs diagnostic and error information for the pol.Loader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to load POL information for a transaction.
name	oracle.documaker.pol.Unloader	Logs diagnostic and error information for the pol.Unloader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to retrieve POL information for a transaction.
name	oracle.documaker.napol.Loader	Logs diagnostic and error information for the napol.Loader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to load NA/POL information for a transaction.
name	oracle.documaker.napol.Unloader	Logs diagnostic and error information for the napol.Unloader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to unload NA/POL information for a transaction.
name	oracle.documaker.section.Loader	Logs diagnostic and error information for the section.Loader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to load section information for a transaction.
name	oracle.documaker.section.Unloader	Logs diagnostic and error information for the section.Unloader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor, and Presenter programs to retrieve section information for a transaction.
name	oracle.documaker.fap.loader.FapLoader	Logs diagnostic and error information for the FapLoader class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor and Presenter programs to load/unload FAP information for a transaction.
name	oracle.documaker.NaPolManager	Logs diagnostic and error information for the NaPolManager class in the Documaker-FAP package. This class is used through JNI by the C Assembler, Distributor and Presenter programs to retrieve NA/POL information for a transaction.
name	oracle.documaker.rp.MonitorMemory	Logs diagnostic and error information for the MonitorMemory class in the Documaker-RP package. This class is used through JNI by the C Assembler, Distributor and Presenter programs to monitor memory usage.

Property	Value	Notes
name	oracle.documaker.rp.MonitorThreads	Logs diagnostic and error information for the MonitorThreads class in the Documaker-RP package. This class is used through JNI by the C Assembler, Distributor and Presenter programs to monitor thread usage.
name	oracle.documaker.rp.MonitorClassLoading	Logs diagnostic and error information for the MonitorClassLoading class in the Documaker-RP package. This class is used through JNI by the C Assembler, Distributor and Presenter programs to monitor class loading.
name	oracle.documaker.batch.Batcher	Logs diagnostic and error information for the Batcher thread class in the Documaker-Batcher package. This class is the main entry point for the Batcher program.
name	oracle.documaker.batch.monitors.BatchTransactions	Logs diagnostic and error information for the BatchTransactions thread class in the Documaker-Batcher package. This is the class that performs all the batching work for the Batcher program.
name	oracle.documaker.batch.housekeeping.BatcherHouseKeeper	Logs diagnostic and error information for the BatcherHouseKeeper thread class in the Documaker-Batcher package. This is the class that performs all the house keeping and clean up for the Batcher program.
name	oracle.documaker.batch.shutdown.BatcherShutdownHook	Logs diagnostic and error information for the BatcherShutdownHook thread class in the Documaker-Batcher package as it performs shut down clean up for the Batcher program such as closing database connections.
name	oracle.documaker.publishing.PublishingManager	Logs diagnostic and error information for the PublishingManager thread class in the Documaker-Publisher package. This class is the main entry point for the Publisher program.
name	oracle.documaker.publishing.PrinterPublisher	Logs diagnostic and error information for the PrinterPublisher class in the Documaker-Publisher package. This class is used by the Publisher program to publish document streams.
name	oracle.documaker.publishing.EmailPublisher	Logs diagnostic and error information for the EmailPublisher class in the Documaker-Publisher package. This class is used by the Publisher program to email document streams.
name	oracle.documaker.archiver.Archiver	Logs diagnostic and error information for the Archiver thread class in the Documaker-Archiver package. This class is the main entry point for the Archiver program.
name	oracle.documaker.archiver.ArchiverHouseKeeper	Logs diagnostic and error information for the ArchiverHouseKeeper thread class in the Documaker-Archiver package. This class is used to perform clean up and maintenance for Archiver program.

Property	Value	Notes
name	oracle.documaker.archiver.shutdown.ArchiverShutdownHook	Logs diagnostic and error information for the ArchiverShutdownHook class in the Documaker-Archiver package as it performs clean up during shut down of the Archiver program.
name	oracle.documaker.archiver.db.PubInterface	Logs diagnostic and error information for the PubInterface class in the Documaker-Archiver package.
name	oracle.documaker.archiver.db.BatchInterface	Logs diagnostic and error information for the BatchInterface class in the Documaker-Archiver package.
name	oracle.documaker.archiver.PropertyUtils	Logs diagnostic and error information for the PropertyUtils class in the Documaker-Archiver package.
name	oracle.documaker.archiver.ArchiveEngine	Logs diagnostic and error information for the ArchiverEngine class in the Documaker-Archiver package.
name	oracle.documaker.archiver.ArchiverSource	Logs diagnostic and error information for the ArchiverSource class in the Documaker-Archiver package.
name	oracle.documaker.archiver.ArchiverBatchManager	Logs diagnostic and error information for the ArchiverBatchManager class in the Documaker-Archiver package.
name	oracle.documaker.connector.destination.UCMDestination	Logs diagnostic and error information for the UCMDestination class in the Documaker-Archiver package.
name	oracle.documaker.connector.destination.requests.Request	Logs diagnostic and error information for the requests.Request class in the Documaker-Archiver package.
name	oracle.documaker.connector.destination.requests.PingRequest	Logs diagnostic and error information for the requests.PingRequest class in the Documaker-Archiver package.
name	oracle.documaker.connector.destination.requests.ImportRequest	Logs diagnostic and error information for the requests.ImportRequest class in the Documaker-Archiver package.
name	oracle.documaker.connector.destination.requests.GetCustomFieldsRequest	Logs diagnostic and error information for the requests.GetCustomFieldsRequest class in the Documaker-Archiver package.
name	oracle.documaker.ezridc.ImportRequest	Logs diagnostic and error information for the ezridc.ImportRequest class in the Documaker-Archiver package.
name	oracle.documaker.ezridc.PingRequest	Logs diagnostic and error information for the ezridc.PingRequest class in the Documaker-Archiver package.
name	oracle.documaker.ezridc.Request	Logs diagnostic and error information for the ezridc.Request class in the Documaker-Archiver package.

Property	Value	Notes
name	oracle.documaker.ezridc.GetCustomFieldsRequest	Logs diagnostic and error information for the ezridc.GetCustomFieldsRequest class in the Documaker-Archiver package.
name	oracle.documaker.PubNotifier.PubNotifier	Logs diagnostic and error information for the PubNotifier thread class in the Documaker-PubNotifier package. This is the main entry point class for the PubNotifier program.
name	oracle.documaker.PubNotifier.housekeeping.PubNotifierHouseKeeper	Logs diagnostic and error information for the PubNotifierHouseKeeper thread class in the Documaker-PubNotifier package. This is the class that performs the housekeeping and cleanup for the PubNotifier program.
name	oracle.documaker.PubNotifier.shutdown.PubNotifierShutdownHook	Logs diagnostic and error information for the PubNotifierShutdownHook thread class as it performs clean up during shutdown of the PubNotifier program.
name	oracle.documaker.PubNotifier.db.PubntfsInterface	Logs diagnostic and error information for the PubntfsInterface class in the Documaker-PubNotifier package.
name	oracle.documaker.PubNotifier.db.RcpInterface	Logs diagnostic and error information for the RcpInterface class in the Documaker-PubNotifier package.
name	oracle.documaker.PubNotifier.db.BchRcpInterface	Logs diagnostic and error information for the BchRcpInterface class in the Documaker-PubNotifier package.
name	oracle.documaker.historian	Logs diagnostic and error information for the Historian program in the Documaker-Historian package.
name	oracle.quartz	Logs diagnostic and error information for the Historian program in the Documaker-Historian package.
name	oracle.documaker.config	Logs diagnostic and error information for the Documaker-Config package.
name	oracle.documaker.dao	Logs diagnostic and error information for the Documaker-DAO package.
name	oracle.documaker.db	Logs diagnostic and error information for the Documaker-DB package.
name	oracle.documaker.process	Logs diagnostic and error information for the Documaker-Process package.
name	oracle.documaker.util	Logs diagnostic and error information for the Documaker-Util package.

Here is a list of the loggers and their configuration options defined in the ALCONFIGCONTEXT table, where the...

- Context_Name column value is *LOG4J*

- Category column value is *Logger*
- Group_Name column value is the value for each logger in the previous table

The Group_Name, Property, Value, and Notes columns are shown here:

Group_Name	Property	Value	Notes
oracle.documaker	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT (standard output).
oracle.documaker	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.util.Manifest	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.util.Manifest	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.util.Manifest	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.util.Manifest	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.util.Manifest	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.util.Manifest	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.util.Manifest	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.util.PlatformSafe	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.util.PlatformSafe	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.util.PlatformSafe	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.util.PlatformSafe	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.util.PlatformSafe	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.util.PlatformSafe	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.util.PlatformSafe	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.log4j	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.log4j	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.log4j	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.log4j	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.log4j	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.log4j	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.log4j	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.db.DataSourceUtil	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.db.DataSourceUtil	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.db.DataSourceUtil	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.db.DataSourceUtil	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.db.DataSourceUtil	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.db.DataSourceUtil	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.db.DataSourceUtil	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.config.jpa.JPAConfigurationFactory	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.config.jpa.JPAConfigurationFactory	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config.jpa.JPAConfigurationFactory	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config.jpa.JPAConfigurationFactory	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.config.jpa.JPAConfigurationFactory	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.config.jpa.JPAConfigurationFactory	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.config.jpa.JPAConfigurationFactory	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.config.jpa.JPAConfiguration	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.config.jpa.JPAConfiguration	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config.jpa.JPAConfiguration	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config.jpa.JPAConfiguration	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.config.jpa.JPAConfiguration	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.config.jpa.JPAConfiguration	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.config.jpa.JPAConfiguration	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.config.db.DataSourceConfigurationFactory	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.config.db.DataSourceConfigurationFactory	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config.db.DataSourceConfigurationFactory	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config.db.DataSourceConfigurationFactory	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.config.db.DataSourceConfigurationFactory	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.config.db.DataSourceConfigurationFactory	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.config.db.DataSourceConfigurationFactory	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.config.db.DataSourceConfiguration	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.config.db.DataSourceConfiguration	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config.db.DataSourceConfiguration	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config.db.DataSourceConfiguration	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.config.db.DataSourceConfiguration	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.config.db.DataSourceConfiguration	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.config.db.DataSourceConfiguration	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.config.xml.XMLConfigurationFactory	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.config.xml.XMLConfigurationFactory	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config.xml.XMLConfigurationFactory	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config.xml.XMLConfigurationFactory	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.config.xml.XMLConfigurationFactory	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.config.xml.XMLConfigurationFactory	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.config.xml.XMLConfigurationFactory	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.config.xml.XMLConfiguration	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.config.xml.XMLConfiguration	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config.xml.XMLConfiguration	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config.xml.XMLConfiguration	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.config.xml.XMLConfiguration	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.config.xml.XMLConfiguration	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.config.xml.XMLConfiguration	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.config.AbstractConfiguration	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.config.AbstractConfiguration	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config.AbstractConfiguration	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config.AbstractConfiguration	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.config.AbstractConfiguration	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.config.AbstractConfiguration	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.config.AbstractConfiguration	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.config.ConfigurationUtil	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.config.ConfigurationUtil	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config.ConfigurationUtil	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config.ConfigurationUtil	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.config.ConfigurationUtil	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.config.ConfigurationUtil	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.config.ConfigurationUtil	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.db.documaker.DocumakerDataSourceFactory	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.db.documaker.DocumakerDataSourceFactory	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.db.documaker.DocumakerDataSourceFactory	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.db.documaker.DocumakerDataSourceFactory	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.db.documaker.DocumakerDataSourceFactory	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.db.documaker.DocumakerDataSourceFactory	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.db.documaker.DocumakerDataSourceFactory	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.db.jndi.JNDIDataSourceFactory	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.db.jndi.JNDIDataSourceFactory	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.db.jndi.JNDIDataSourceFactory	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.db.jndi.JNDIDataSourceFactory	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.db.jndi.JNDIDataSourceFactory	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.db.jndi.JNDIDataSourceFactory	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.db.jndi.JNDIDataSourceFactory	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.db.Query	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.db.Query	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.db.Query	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.db.Query	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.db.Query	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.db.Query	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.db.Query	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.db.SQLData	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.db.SQLData	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.db.SQLData	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.db.SQLData	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.db.SQLData	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.db.SQLData	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.db.SQLData	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.DAOUtil	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.DAOUtil	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.DAOUtil	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.DAOUtil	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.dao.DAOUtil	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.DAOUtil	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.DAOUtil	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.AbstractDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.AbstractDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.AbstractDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.AbstractDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.AbstractDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.AbstractDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.AbstractDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.AbstractDAO.Timer	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.dao.AbstractDAO.Timer	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.AbstractDAO.Timer	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.AbstractDAO.Timer	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.AbstractDAO.Timer	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.AbstractDAO.Timer	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.AbstractDAO.Timer	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.jobs.LockDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.jobs.LockDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.jobs.LockDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.jobs.LockDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.jobs.LockDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.jobs.LockDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.jobs.LockDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.jobs.JobDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.jobs.JobDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.jobs.JobDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.jobs.JobDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.jobs.JobDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.jobs.JobDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.jobs.JobDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.jobs.JOBS	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.jobs.JOBS	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.jobs.JOBS	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.jobs.JOBS	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.jobs.JOBS	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.jobs.JOBS	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.jobs.JOBS	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.jobs.JOBS.Exception	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.jobs.JOBS.Exception	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.jobs.JOBS.Exception	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.jobs.JOBS.Exception	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.jobs.JOBS.Exception	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.jobs.JOBS.Exception	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.jobs.JOBS.Exception	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.jobs.JOBS.Extract	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.jobs.JOBS.Extract	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.jobs.JOBS.Extract	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.jobs.JOBS.Extract	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.jobs.JOBS.Extract	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.jobs.JOBS.Extract	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.jobs.JOBS.Extract	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.trns.TrnDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.trns.TrnDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.trns.TrnDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.trns.TrnDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.trns.TrnDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.trns.TrnDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.trns.TrnDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.trns.TRNS	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.trns.TRNS	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.trns.TRNS	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.trns.TRNS	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.trns.TRNS	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.trns.TRNS	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.trns.TRNS	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.rcps.RcpDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.rcps.RcpDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.rcps.RcpDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.rcps.RcpDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.rcps.RcpDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.rcps.RcpDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.rcps.RcpDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.trns.RCPS	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.trns.RCPS	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.trns.RCPS	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.trns.RCPS	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.trns.RCPS	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.trns.RCPS	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.trns.RCPS	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.bchs.BchDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.bchs.BchDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.bchs.BchDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.bchs.BchDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.bchs.BchDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.bchs.BchDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.bchs.BchDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.bchs.BCHS	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.bchs.BCHS	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.bchs.BCHS	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.bchs.BCHS	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.bchs.BCHS	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.bchs.BCHS	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.bchs.BCHS	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.rcps_bchs.RcpBchDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.rcps_bchs.RCPS_BCHS	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.pubs.PubDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.pubs.PubDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.pubs.PubDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.pubs.PubDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.dao.tables.pubs.PubDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.pubs.PubDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.pubs.PubDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao.tables.pubs.PUBS	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao.tables.pubs.PUBS	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao.tables.pubs.PUBS	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao.tables.pubs.PUBS	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao.tables.pubs.PUBS	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao.tables.pubs.PUBS	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao.tables.pubs.PUBS	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.db.documaker.DBUtil	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.db.documaker.DBUtil	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.db.documaker.DBUtil	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.db.documaker.DBUtil	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.db.documaker.DBUtil	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.db.documaker.DBUtil	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.db.documaker.DBUtil	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
http.debug	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
http.debug	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
http.debug	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
http.debug	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
http.debug	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
http.debug	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
http.debug	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
mqseries.debug	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
mqseries.debug	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
mqseries.debug	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
mqseries.debug	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
mqseries.debug	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
mqseries.debug	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
mqseries.debug	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
msmq.debug	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
msmq.debug	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
msmq.debug	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
msmq.debug	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
msmq.debug	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
msmq.debug	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
msmq.debug	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
jms.debug	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
jms.debug	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
jms.debug	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
jms.debug	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
jms.debug	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
jms.debug	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
jms.debug	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.bus	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.bus	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.bus	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.bus	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.bus	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.bus	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.bus	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.ProcessShell	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.ProcessShell	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.ProcessShell	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.ProcessShell	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.ProcessShell	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.ProcessShell	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.ProcessShell	priority	info	null

Group_Name	Property	Value	Notes
oracle.documaker.process.exception.ExceptionHandler	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.exception.ExceptionHandler	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.exception.ExceptionHandler	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.exception.ExceptionHandler	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.exception.ExceptionHandler	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.exception.ExceptionHandler	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.exception.ExceptionHandler	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.util.ProcessUtil	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.util.ProcessUtil	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.util.ProcessUtil	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.util.ProcessUtil	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.process.util.ProcessUtil	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.util.ProcessUtil	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.util.ProcessUtil	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.ipc.IPCConnector	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.ipc.IPCConnector	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.ipc.IPCConnector	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.ipc.IPCConnector	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.ipc.IPCConnector	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.ipc.IPCConnector	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.ipc.IPCConnector	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.ipc.PipeReader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.process.ipc.PipeReader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.ipc.PipeReader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.ipc.PipeReader	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.ipc.PipeReader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.ipc.PipeReader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.ipc.PipeReader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.ipc.PipeWriter	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.ipc.PipeWriter	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.ipc.PipeWriter	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.ipc.PipeWriter	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.ipc.PipeWriter	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.ipc.PipeWriter	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.process.ipc.PipeWriter	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
com.oracle.npc	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
com.oracle.npc	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
com.oracle.npc	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
com.oracle.npc	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
com.oracle.npc	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
com.oracle.npc	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
com.oracle.npc	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.ipc.IPCThread	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.ipc.IPCThread	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.ipc.IPCThread	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.process.ipc.IPCThread	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.ipc.IPCThread	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.ipc.IPCThread	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.ipc.IPCThread	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.monitors.DataSourceMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.monitors.DataSourceMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.monitors.DataSourceMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.monitors.DataSourceMonitor	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.monitors.DataSourceMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.monitors.DataSourceMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.monitors.DataSourceMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.process.monitors.WorkerMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.monitors.WorkerMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.monitors.WorkerMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.monitors.WorkerMonitor	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.monitors.WorkerMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.monitors.WorkerMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.monitors.WorkerMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.monitors.Log4jMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.monitors.Log4jMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.monitors.Log4jMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.monitors.Log4jMonitor	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.process.monitors.Log4jMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.monitors.Log4jMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.monitors.Log4jMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.monitors.FileMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process.monitors.FileMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.monitors.FileMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.monitors.FileMonitor	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.monitors.FileMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.monitors.FileMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.monitors.FileMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process.worker.Worker	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.process.worker.Worker	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process.worker.Worker	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process.worker.Worker	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.process.worker.Worker	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.process.worker.Worker	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process.worker.Worker	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
root	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
root	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
root	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
root	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
root	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
ProcessMonitor.output	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
ProcessMonitor.output	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
ProcessMonitor.output	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
ProcessMonitor.output	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
ProcessMonitor.output	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
ProcessMonitor.output	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
ProcessMonitor.output	priority	info	null
oracle.documaker.processmonitor.ProcessMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.ProcessMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.ProcessMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.ProcessMonitor	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.processmonitor.ProcessMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.ProcessMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.ProcessMonitor	priority	info	null
oracle.documaker.processmonitor.monitors.SelfMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.monitors.SelfMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.monitors.SelfMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.monitors.SelfMonitor	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.processmonitor.monitors.SelfMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.monitors.SelfMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.monitors.SelfMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	appender-ref	process-roll	The name of the appender that logs Log4J statements to the file system.
oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.monitors.SelfLog4jMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.monitors.FileMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.monitors.FileMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.monitors.FileMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.monitors.FileMonitor	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.monitors.FileMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.monitors.FileMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.monitors.FileMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.process.monitors.DBConfigurationMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.loadbalancing.LoadBalancer	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.loadbalancing.LoadBalancer	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.loadbalancing.LoadBalancer	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.loadbalancing.LoadBalancer	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.loadbalancing.LoadBalancer	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.loadbalancing.LoadBalancer	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.loadbalancing.LoadBalancer	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.deployment.HotDeployer	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.deployment.HotDeployer	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.deployment.HotDeployer	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.deployment.HotDeployer	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.deployment.HotDeployer	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.deployment.HotDeployer	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.deployment.HotDeployer	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.deployment.DeployWorker	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.deployment.DeployWorker	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.deployment.DeployWorker	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.deployment.DeployWorker	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.deployment.DeployWorker	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.deployment.DeployWorker	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.deployment.DeployWorker	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.process.Process	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.process.Process	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.process.Process	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.process.Process	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.process.Process	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.process.Process	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.process.Process	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.process.data.ProcessData	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.process.data.ProcessData	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.process.data.ProcessData	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.process.data.ProcessData	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.process.data.ProcessData	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.process.data.ProcessData	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.process.data.ProcessData	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.process.monitors.InstanceMonitor.Restart	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.process.instance.Instance	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.process.instance.Instance	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.process.instance.Instance	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.process.instance.Instance	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.process.instance.Instance	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.process.instance.Instance	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.process.instance.Instance	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.ipc.PipeReader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.processmonitor.ipc.PipeReader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.ipc.PipeReader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.ipc.PipeReader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.ipc.PipeReader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.ipc.PipeReader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.ipc.PipeReader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.processmonitor.ipc.PipeWriter	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.processmonitor.ipc.PipeWriter	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.processmonitor.ipc.PipeWriter	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.processmonitor.ipc.PipeWriter	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.processmonitor.ipc.PipeWriter	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.processmonitor.ipc.PipeWriter	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.processmonitor.ipc.PipeWriter	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
com.docucorp.jnative	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
com.docucorp.jnative	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
com.docucorp.jnative	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
com.docucorp.jnative	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
com.docucorp.jnative	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
com.docucorp.jnative	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
com.docucorp.jnative	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
EMAIL	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
EMAIL	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
EMAIL	appender-ref	EMAIL	The email appender to use when sending error or fatal notifications.
EMAIL	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.Scheduler	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.Scheduler	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.Scheduler	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.Scheduler	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.Scheduler	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.Scheduler	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.scheduler.Scheduler	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.housekeeping.SchedulerHouseKeeper	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.shutdown.SchedulerShutdownHook	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.scheduler.shutdown.SchedulerShutdownHook	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.shutdown.SchedulerShutdownHook	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.shutdown.SchedulerShutdownHook	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.shutdown.SchedulerShutdownHook	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.shutdown.SchedulerShutdownHook	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.shutdown.SchedulerShutdownHook	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.monitors.NotifyIdentifier	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.monitors.NotifyIdentifier	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.monitors.NotifyIdentifier	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.monitors.NotifyIdentifier	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.monitors.NotifyIdentifier	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.monitors.NotifyIdentifier	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.scheduler.monitors.NotifyIdentifier	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.monitors.NotifyAssembler	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.monitors.NotifyAssembler	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.monitors.NotifyAssembler	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.monitors.NotifyAssembler	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.monitors.NotifyAssembler	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.monitors.NotifyAssembler	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.monitors.NotifyAssembler	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.monitors.NotifyDistributor	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.monitors.NotifyDistributor	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.monitors.NotifyDistributor	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.scheduler.monitors.NotifyDistributor	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.monitors.NotifyDistributor	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.monitors.NotifyDistributor	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.monitors.NotifyDistributor	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.monitors.NotifyPresenterImmediate	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.monitors.NotifyPresenterImmediate	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.monitors.NotifyPresenterImmediate	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.monitors.NotifyPresenterImmediate	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.monitors.NotifyPresenterImmediate	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.monitors.NotifyPresenterImmediate	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.monitors.NotifyPresenterImmediate	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.scheduler.monitors.NotifyPresenterScheduled	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.monitors.NotifyPresenterScheduled	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.monitors.NotifyPresenterScheduled	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.monitors.NotifyPresenterScheduled	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.monitors.NotifyPresenterScheduled	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.monitors.NotifyPresenterScheduled	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.monitors.NotifyPresenterScheduled	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.monitors.NotifyArchiver	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.monitors.NotifyArchiver	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.monitors.NotifyArchiver	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.scheduler.monitors.NotifyArchiver	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.monitors.NotifyArchiver	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.monitors.NotifyArchiver	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.monitors.NotifyArchiver	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.scheduler.monitors.NotifyPublisher	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.monitors.NotifyPublisher	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.monitors.NotifyPublisher	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.monitors.NotifyPublisher	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.monitors.NotifyPublisher	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.monitors.NotifyPublisher	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.monitors.NotifyPublisher	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.scheduler.monitors.NotifyPubNotifier	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.scheduler.monitors.NotifyPubNotifier	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.scheduler.monitors.NotifyPubNotifier	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.scheduler.monitors.NotifyPubNotifier	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.scheduler.monitors.NotifyPubNotifier	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.scheduler.monitors.NotifyPubNotifier	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.scheduler.monitors.NotifyPubNotifier	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.receiver.Receiver	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.receiver.Receiver	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.receiver.Receiver	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.receiver.Receiver	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.receiver.Receiver	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.receiver.Receiver	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.receiver.Receiver	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.receiver.housekeeping.ReceiverHouseKeeper	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.receiver.housekeeping.ReceiverHouseKeeper	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.receiver.housekeeping.ReceiverHouseKeeper	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.receiver.housekeeping.ReceiverHouseKeeper	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.receiver.housekeeping.ReceiverHouseKeeper	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.receiver.housekeeping.ReceiverHouseKeeper	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.receiver.housekeeping.ReceiverHouseKeeper	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.receiver.shutdown.ReceiverShutdownHook	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.receiver.shutdown.ReceiverShutdownHook	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.receiver.shutdown.ReceiverShutdownHook	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.receiver.shutdown.ReceiverShutdownHook	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.receiver.shutdown.ReceiverShutdownHook	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.receiver.shutdown.ReceiverShutdownHook	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.receiver.shutdown.ReceiverShutdownHook	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.receiver.monitors.FileReceiver	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.receiver.monitors.FileReceiver	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.receiver.monitors.FileReceiver	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.receiver.monitors.FileReceiver	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.receiver.monitors.FileReceiver	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.receiver.monitors.FileReceiver	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.receiver.monitors.FileReceiver	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.receiver.monitors.QueueReceiver	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.receiver.monitors.QueueReceiver	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.receiver.monitors.QueueReceiver	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.receiver.monitors.QueueReceiver	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.receiver.monitors.QueueReceiver	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.receiver.monitors.QueueReceiver	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.receiver.monitors.QueueReceiver	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.receiver.monitors.QueueReceiverWorker	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.receiver.monitors.QueueReceiverWorker	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.receiver.monitors.QueueReceiverWorker	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.receiver.monitors.QueueReceiverWorker	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.receiver.monitors.QueueReceiverWorker	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.receiver.monitors.QueueReceiverWorker	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.receiver.monitors.QueueReceiverWorker	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.identifier.Identifier	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.identifier.Identifier	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.identifier.Identifier	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.identifier.Identifier	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.identifier.Identifier	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.identifier.Identifier	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.identifier.Identifier	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.identifier.housekeeping.IdentifierHouseKeeper	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.identifier.housekeeping.IdentifierHouseKeeper	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.identifier.housekeeping.IdentifierHouseKeeper	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.identifier.housekeeping.IdentifierHouseKeeper	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.identifier.housekeeping.IdentifierHouseKeeper	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.identifier.housekeeping.IdentifierHouseKeeper	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.identifier.housekeeping.IdentifierHouseKeeper	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.identifier.shutdown.IdentifierShutdownHook	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.identifier.shutdown.IdentifierShutdownHook	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.identifier.shutdown.IdentifierShutdownHook	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.identifier.shutdown.IdentifierShutdownHook	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.identifier.shutdown.IdentifierShutdownHook	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.identifier.shutdown.IdentifierShutdownHook	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.identifier.shutdown.IdentifierShutdownHook	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
LogLogger	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
LogLogger	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
LogLogger	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
LogLogger	priority	debug	null

Group_Name	Property	Value	Notes
ErrorLogger	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
ErrorLogger	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
ErrorLogger	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
ErrorLogger	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.rp.jdbc.GenericDAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.rp.jdbc.GenericDAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.rp.jdbc.GenericDAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.rp.jdbc.GenericDAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.rp.jdbc.GenericDAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.rp.jdbc.GenericDAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.rp.jdbc.GenericDAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.rp.jdbc.DAO	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.rp.jdbc.DAO	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.rp.jdbc.DAO	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.rp.jdbc.DAO	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.rp.jdbc.DAO	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.rp.jdbc.DAO	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.rp.jdbc.DAO	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.rp.bus.Bus	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.rp.bus.Bus	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.rp.bus.Bus	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.rp.bus.Bus	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.rp.bus.Bus	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.rp.bus.Bus	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.rp.bus.Bus	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.rp.config.Configuration	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.rp.config.Configuration	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.rp.config.Configuration	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.rp.config.Configuration	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.rp.config.Configuration	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.rp.config.Configuration	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.rp.config.Configuration	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.na.Loader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.na.Loader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.

Group_Name	Property	Value	Notes
oracle.documaker.na.Loader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.na.Loader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.na.Loader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.na.Loader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.na.Loader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.na.Unloader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.na.Unloader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.na.Unloader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.na.Unloader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.na.Unloader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.na.Unloader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.na.Unloader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.pol.Loader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.pol.Loader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.pol.Loader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.pol.Loader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.pol.Loader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.pol.Loader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.pol.Loader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.pol.Unloader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.pol.Unloader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.pol.Unloader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.pol.Unloader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.pol.Unloader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.pol.Unloader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.pol.Unloader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.napol.Loader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.napol.Loader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.napol.Loader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.napol.Loader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.napol.Loader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.napol.Loader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.napol.Loader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.napol.Unloader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.napol.Unloader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.napol.Unloader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.napol.Unloader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.napol.Unloader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.napol.Unloader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.napol.Unloader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.section.Loader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.section.Loader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.section.Loader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.section.Loader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.section.Loader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.section.Loader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.section.Loader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.section.Unloader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.section.Unloader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.section.Unloader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.section.Unloader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.section.Unloader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.section.Unloader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.section.Unloader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.fap.loader.FapLoader	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.fap.loader.FapLoader	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.fap.loader.FapLoader	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.fap.loader.FapLoader	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.fap.loader.FapLoader	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.fap.loader.FapLoader	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.fap.loader.FapLoader	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.NaPolManager	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.NaPolManager	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.NaPolManager	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.NaPolManager	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.NaPolManager	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.NaPolManager	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.NaPolManager	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.rp.MonitorThreads	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.rp.MonitorThreads	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.

Group_Name	Property	Value	Notes
oracle.documaker.rp.MonitorThreads	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.rp.MonitorThreads	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.rp.MonitorThreads	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.rp.MonitorThreads	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.rp.MonitorThreads	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.rp.MonitorClassLoading	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.rp.MonitorClassLoading	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.rp.MonitorClassLoading	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.rp.MonitorClassLoading	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.rp.MonitorClassLoading	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.rp.MonitorClassLoading	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.rp.MonitorClassLoading	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.batch.Batcher	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.batch.Batcher	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.batch.Batcher	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.batch.Batcher	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.batch.Batcher	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.batch.Batcher	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.batch.Batcher	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.batch.monitors.BatchTransactions	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.batch.monitors.BatchTransactions	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.batch.monitors.BatchTransactions	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.batch.monitors.BatchTransactions	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.batch.monitors.BatchTransactions	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.batch.monitors.BatchTransactions	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.batch.monitors.BatchTransactions	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.batch.housekeeping.BatcherHouseKeeper	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.batch.housekeeping.BatcherHouseKeeper	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.batch.housekeeping.BatcherHouseKeeper	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.batch.housekeeping.BatcherHouseKeeper	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.batch.housekeeping.BatcherHouseKeeper	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.batch.housekeeping.BatcherHouseKeeper	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.batch.housekeeping.BatcherHouseKeeper	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.batch.shutdown.BatcherShutdownHook	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.batch.shutdown.BatcherShutdownHook	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.batch.shutdown.BatcherShutdownHook	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.batch.shutdown.BatcherShutdownHook	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.batch.shutdown.BatcherShutdownHook	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.batch.shutdown.BatcherShutdownHook	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.batch.shutdown.BatcherShutdownHook	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.publishing.PublishingManager	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.publishing.PublishingManager	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.publishing.PublishingManager	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.publishing.PublishingManager	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.publishing.PublishingManager	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.publishing.PublishingManager	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.publishing.PublishingManager	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.publishing.PrinterPublisher	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.publishing.PrinterPublisher	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.publishing.PrinterPublisher	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.publishing.PrinterPublisher	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.publishing.PrinterPublisher	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.publishing.PrinterPublisher	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.publishing.PrinterPublisher	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.publishing.EmailPublisher	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.publishing.EmailPublisher	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.publishing.EmailPublisher	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.publishing.EmailPublisher	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.publishing.EmailPublisher	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.publishing.EmailPublisher	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.publishing.EmailPublisher	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.archiver.Archiver	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.archiver.Archiver	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.Archiver	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.archiver.Archiver	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.Archiver	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.archiver.Archiver	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.archiver.Archiver	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.archiver.housekeeping.ArchiverHouseKeeper	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.archiver.housekeeping.ArchiverHouseKeeper	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.housekeeping.ArchiverHouseKeeper	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.archiver.housekeeping.ArchiverHouseKeeper	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.housekeeping.ArchiverHouseKeeper	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.archiver.housekeeping.ArchiverHouseKeeper	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.archiver.housekeeping.ArchiverHouseKeeper	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.archiver.shutdown.ArchiverShutdownHook	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.archiver.shutdown.ArchiverShutdownHook	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.shutdown.ArchiverShutdownHook	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.archiver.shutdown.ArchiverShutdownHook	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.shutdown.ArchiverShutdownHook	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.archiver.shutdown.ArchiverShutdownHook	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.archiver.shutdown.ArchiverShutdownHook	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.archiver.db.PubInterface	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.archiver.db.PubInterface	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.db.PubInterface	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.archiver.db.PubInterface	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.db.PubInterface	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.archiver.db.PubInterface	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.archiver.db.PubInterface	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.archiver.db.BatchInterface	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.archiver.db.BatchInterface	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.db.BatchInterface	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.archiver.db.BatchInterface	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.db.BatchInterface	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.archiver.db.BatchInterface	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.archiver.db.BatchInterface	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.archiver.PropertyUtils	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.archiver.PropertyUtils	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.PropertyUtils	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.

Group_Name	Property	Value	Notes
oracle.documaker.archiver.PropertyUtils	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.PropertyUtils	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.archiver.PropertyUtils	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.archiver.PropertyUtils	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.archiver.ArchiveEngine	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.archiver.ArchiveEngine	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.ArchiveEngine	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.archiver.ArchiveEngine	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.ArchiveEngine	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.archiver.ArchiveEngine	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.archiver.ArchiveEngine	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.archiver.ArchiverSource	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.archiver.ArchiverSource	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.ArchiverSource	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.archiver.ArchiverSource	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.ArchiverSource	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.archiver.ArchiverSource	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.archiver.ArchiverSource	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.archiver.ArchiverBatchManager	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.archiver.ArchiverBatchManager	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.archiver.ArchiverBatchManager	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.archiver.ArchiverBatchManager	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.archiver.ArchiverBatchManager	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.archiver.ArchiverBatchManager	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.archiver.ArchiverBatchManager	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.connector.destination.UCMDestination	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.connector.destination.UCMDestination	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.connector.destination.UCMDestination	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.connector.destination.UCMDestination	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.connector.destination.UCMDestination	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.connector.destination.UCMDestination	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.connector.destination.UCMDestination	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.connector.destination.requests.Request	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.connector.destination.requests.Request	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.

Group_Name	Property	Value	Notes
oracle.documaker.connector.destination.requests.Request	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.connector.destination.requests.Request	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.connector.destination.requests.Request	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.connector.destination.requests.Request	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.connector.destination.requests.Request	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.connector.destination.requests.PingRequest	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.connector.destination.requests.PingRequest	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.connector.destination.requests.PingRequest	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.connector.destination.requests.PingRequest	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.connector.destination.requests.PingRequest	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.connector.destination.requests.PingRequest	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.connector.destination.requests.PingRequest	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.connector.destination.requests.ImportRequest	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.connector.destination.requests.ImportRequest	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.connector.destination.requests.ImportRequest	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.connector.destination.requests.ImportRequest	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.connector.destination.requests.ImportRequest	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.connector.destination.requests.ImportRequest	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.connector.destination.requests.ImportRequest	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.connector.destination.requests.GetCustomFieldsRequest	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.connector.destination.requests.GetCustomFieldsRequest	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.connector.destination.requests.GetCustomFieldsRequest	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.connector.destination.requests.GetCustomFieldsRequest	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.connector.destination.requests.GetCustomFieldsRequest	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.connector.destination.requests.GetCustomFieldsRequest	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.connector.destination.requests.GetCustomFieldsRequest	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.ezridc.ImportRequest	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.ezridc.ImportRequest	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.ezridc.ImportRequest	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.ezridc.ImportRequest	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.ezridc.ImportRequest	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.ezridc.ImportRequest	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.ezridc.ImportRequest	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.ezridc.PingRequest	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.ezridc.PingRequest	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.ezridc.PingRequest	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.ezridc.PingRequest	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.ezridc.PingRequest	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.ezridc.PingRequest	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.ezridc.PingRequest	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.ezridc.Request	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.ezridc.Request	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.ezridc.Request	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.ezridc.Request	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.

Group_Name	Property	Value	Notes
oracle.documaker.ezridc.Request	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.ezridc.Request	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.ezridc.Request	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.ezridc.GetCustomFieldsRequest	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.ezridc.GetCustomFieldsRequest	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.ezridc.GetCustomFieldsRequest	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.ezridc.GetCustomFieldsRequest	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.ezridc.GetCustomFieldsRequest	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.ezridc.GetCustomFieldsRequest	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.ezridc.GetCustomFieldsRequest	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.PubNotifier.PubNotifier	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.

Group_Name	Property	Value	Notes
oracle.documaker.PubNotifier.PubNotifier	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.PubNotifier.PubNotifier	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.PubNotifier.PubNotifier	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.PubNotifier.PubNotifier	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.PubNotifier.PubNotifier	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.PubNotifier.PubNotifier	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.PubNotifier.housekeeping.PubNotifierHouseKeeper	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.PubNotifier.housekeeping.PubNotifierHouseKeeper	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.PubNotifier.housekeeping.PubNotifierHouseKeeper	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.PubNotifier.housekeeping.PubNotifierHouseKeeper	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.PubNotifier.housekeeping.PubNotifierHouseKeeper	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.PubNotifier.housekeeping.PubNotifierHouseKeeper	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.

Group_Name	Property	Value	Notes
oracle.documaker.PubNotifier.housekeeping.PubNotifierHouseKeeper	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.PubNotifier.shutdown.PubNotifierShutdownHook	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.PubNotifier.shutdown.PubNotifierShutdownHook	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.PubNotifier.shutdown.PubNotifierShutdownHook	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.PubNotifier.shutdown.PubNotifierShutdownHook	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.PubNotifier.shutdown.PubNotifierShutdownHook	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.PubNotifier.shutdown.PubNotifierShutdownHook	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.PubNotifier.shutdown.PubNotifierShutdownHook	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.PubNotifier.db.PubntfsInterface	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.PubNotifier.db.PubntfsInterface	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.

Group_Name	Property	Value	Notes
oracle.documaker.PubNotifier.db.PubntfsInterface	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.PubNotifier.db.PubntfsInterface	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.PubNotifier.db.PubntfsInterface	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.PubNotifier.db.PubntfsInterface	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.PubNotifier.db.PubntfsInterface	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.PubNotifier.db.RcpInterface	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.PubNotifier.db.RcpInterface	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.PubNotifier.db.RcpInterface	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.PubNotifier.db.RcpInterface	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.PubNotifier.db.RcpInterface	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.PubNotifier.db.RcpInterface	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.PubNotifier.db.RcpInterface	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.PubNotifier.db.BchRcpInterface	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.PubNotifier.db.BchRcpInterface	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.PubNotifier.db.BchRcpInterface	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.PubNotifier.db.BchRcpInterface	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.PubNotifier.db.BchRcpInterface	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.PubNotifier.db.BchRcpInterface	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.PubNotifier.db.BchRcpInterface	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.config	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.config	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.config	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.config	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.config	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.config	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.config	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.dao	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.dao	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.dao	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.dao	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.dao	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.dao	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.dao	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.db	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.db	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.

Group_Name	Property	Value	Notes
oracle.documaker.db	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.db	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.db	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.db	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.db	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.historian	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.historian	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.historian	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.historian	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.historian	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.historian	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.historian	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Group_Name	Property	Value	Notes
oracle.documaker.identifier	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.identifier	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.identifier	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.identifier	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.identifier	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.identifier	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.identifier	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.process	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.process	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.process	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.process	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.process	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.

Group_Name	Property	Value	Notes
oracle.documaker.process	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.process	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.documaker.util	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.documaker.util	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.
oracle.documaker.util	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.documaker.util	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.documaker.util	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.documaker.util	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.documaker.util	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.
oracle.quartz	additivity	No	In Log4J, all loggers have a hierarchy. A logger is an ancestor of another logger if its name space is included in the descendant's name space. Additivity means that Log4J statements that are logged by ancestors are also inherited and logged by this logger. Set the value to No to avoid duplicate logging.
oracle.quartz	class	oracle.documaker.log4j.logger.DFLogger	The fully-qualified class name of the logger class to log the Log4J statements; should be the DFLogger class.

Group_Name	Property	Value	Notes
oracle.quartz	appender-ref	stdout	The name of the appender that logs the Log4J statement to STDOUT.
oracle.quartz	appender-ref	roll	The name of the appender that logs the Log4J statement to the file system.
oracle.quartz	appender-ref	LogAppender	The name of the appender that logs the Log4J statement to the Logs database table.
oracle.quartz	appender-ref	ErrorAppender	The name of the appender that logs the Log4J statement to the Errs database table.
oracle.quartz	priority	error	The Log4J priority/level. You can set this to: info, warn, debug, error, fatal, all, or off. See the Log4J documentation for additional details. Set this value to error to tell the system to log only error or fatal messages.

Note See the Log4J sections for each process configuration topic for more information the Log4J loggers you can enable and disable at the application level to capture diagnostic information.

Chapter 5

Configuring Documaker Interactive: Correspondence

This chapter describes how to configure Documaker Interactive: Correspondence. It includes examples and descriptions of additional configuration options that extend the functionality of the default product.

This chapter includes the following topics:

CONFIGURING THE IDS CONNECTION

The section name for the default connection is *idsConnection*. The values you enter for that section are used as defaults for every request in the system.

Using the default Docupresentment (IDS) configuration, you can create variations based on the type of configuration (JMS, HTTP, and so on), the master resource library (MRL), or individual IDS requests. You can define different types of connections and within those connections configure a request differently for a particular IDS request.

SETTING UP MRL-BASED CONNECTIONS

You can configure a separate connection for a specific MRL. If an MRL connection exists, it overrides the default connection.

To create an MRL-based configuration, just prefix the section name of the connection with the name of the MRL, followed by an underscore. For example, to create a configuration used only for an MRL named *acme*, you could create this connection:

```
acme_idsConnection
```

You would then configure this connection in the same manner as the default connection. Choose the implementation class and then fill in the properties needed for that implementation along with your environment's values. An MRL-based connection requires an IDS request to be defined as part of that MRL.

SETTING UP REQUEST-BASED CONNECTIONS

You can also based a configuration on a particular request. Configurations can be shared among the whole system, MRLs, or requests.

For example, suppose you want to use the JMS connection defined previously as the default. You would just leave the section named *idsConnection*, along with a similar JMS configuration. For a couple of requests, however, you want to use an HTTP connection. You could define a new HTTP connection the same way as in the first example, except with a new section name.

For this example, call it *specialConnection*. All you have to do is change the first example (idsConnection) to use the section name *specialConnection* everywhere it says *idsConnection*.

Note Keep in mind, this is in addition to the default connection. This is not a replacement for it.

Each configuration element in the system has the same pattern of section name, property, and value. This includes the IDS requests. Let's look at a basic IDS request and tell it to use the *specialConnection*. If a connection is not explicitly defined, the default is used.

Here is a sample IDS request using a non-default IDS connection:

Option	Description
Section Name = FORMS_INIT_DATA	
(class)	Defines the implementation class. Here is an example: oracle.documaker.ids.bll.IDSFormsInitData
request	Defines the IDS method name. Here is an example: iDM_GetMRLResource
.....	Defines other properties for this request. These are ignored in this example.
idsConnection	Defines the connection used for this request. The connection (specialConnection) has to exist as a section name somewhere in the configuration. This section name/connection has to be a working, configured IDS connection.

In this example, each IDS request of type *FORMS_INIT_DATA* connects to the IDS instance *specialConnection* is configured to connect to.

Configuring IDS Requests

Each IDS request has its own configuration in the system with its own properties. Here is an example of the common properties for all IDS requests.

Option	Description
Section Name = *someIDSRequest*	
(class)	The (class) defines the implementation class for this request. It is code that implements this IDS request. Typically, you should accept the default.
request	This is the name of the actual IDS request configured on IDS, such as iDM_GetMRLResource.
config	(Optional) This defines the MRL for this implementation, such as Correspondence. You use this property if you want an MRL-specific configuration.
idsConnection	(Optional) The connection used for this request. The value (*aDefinedConnection*) must exist as a section name somewhere in the configuration. This section name/ connection has to be a working, configured IDS connection. The idsConnection parameter overrides any configuration or system connection. If you define this parameter, no other connection is used.
username	(Optional) The user name for this IDS request, if applicable. This is needed if connection credentials are required by IDS.
password	(Optional) The password for this IDS request, if applicable. This is needed if connection credentials are required by IDS.
locale	(Optional) The locale for this IDS request, such as US. An entry here overrides the default, system-level locale.
extra	(Optional) This variable lets you configure this request using an extra variable that may not be accounted for, such as a newly added variable, that you want to explicitly configure for IDS. If there is no property for a particular IDS property, you can enter it explicitly as an extra property.

Option	Description
mapping	(Optional) You can use this property to map the IDS configuration property to a method in Documaker Interactive. Do not change these values unless you are changing the code to handle it.
attachmentMapping	(Optional) You can use this property to map the attachment in an IDS configuration property to a method in Documaker Interactive. Do not change these values unless you are changing the code to handle it.
response	(Optional) You can use this property to map the configuration property in an IDS response to a method in Documaker Interactive. Do not change these values unless you are changing the code to handle it.
responseAttachment	(Optional) You can use this property to map the configuration property in an IDS response attachment to a method in Documaker Interactive. Do not change these values unless you are changing the code to handle it.

Note You can use the mapping, attachmentMapping, response, and responseAttachments properties to map IDS properties to specific Documaker Interactive implementations. These properties are not usually changed unless you are writing IDS implementation components.

DEFINING SYSTEM-WIDE DEFAULTS

Some settings, such as the idsConnection settings, define system-wide default behavior. Non-connection, system-level configurations are located in the SYSTEM_IDS configuration section.

Option	Description
Section Name = SYSTEM_IDS	
(class)	Defines the implementation class for the base system configuration. Do not change this value.
(init-method)	Defines a method to load upon initialization. This lets the Documaker Interactive system load all the configuration data when you boot the application. Do not change this value.
defaultConfig	Defines the master resource library (MRL) is used throughout the system unless an MRL is defined in the request.
configList	Defines a comma-separated list of supported MRLs.
timeout	Defines the timeout interval, in milliseconds, for each IDS request. The default is 180 seconds (180000).
tries	Defines the number of times to try each IDS request after a failure. The default is 1 (one).
locale	Defines the default locale to use for each IDS request. The default is en (English).
helpLink	Defines the URL that points to the Documaker Interactive Help system. The Help system can be hosted remotely or locally.
inboxRefresh	Defines how often, in milliseconds, to refresh the inbox data.
responseAttachment	You can use this property to map the configuration property in an IDS response attachment to a method in Documaker Interactive. Do not change these values unless you are also changing the code to handle it.

CONFIGURING IDS REQUESTS

Each IDS request Documaker Interactive uses has a configuration section. This topic describes the default configuration. In this section you edit the default configuration for a particular request or define a new one for a configuration.

Keep in mind the default structure and configuration for each IDS request are the same and can be configured accordingly regardless of the overlying implementation. This means that for every IDS request, you can define the base IDS properties such as connection, config, and so on, and also request-specific properties. Each request is different and has properties specific to it.

Common IDS Request Properties

Anytime you configure an IDS request, because a request property has to be defined (iDM_GetMRLResource in this example), you can use all of the properties in this section.

Here are the properties common to all IDS requests:

Option	Description
Section Name = *someIDSRequest*	
(class)	This value identifies a piece of code that implements this IDS request. This value changes for each IDS request. You can have different implementations for the same IDS request if the processing is handled differently.
request	Defines the IDS method name, such as iDM_GetMRLResource.
config	(Optional) Defines the MRL for this particular IDS request. Correspondence is an example.
idsConnection	(Optional) Defines the connection used for this request. The value (aDefinedConnection) has to exist as a section name somewhere in the configuration. This section name/connection has to be a working, configured IDS connection.
username	(Optional) Defines, if applicable, the user name for this particular IDS request.
password	(Optional) Defines, if applicable, the password for this particular IDS request.
locale	(Optional) Defines the default locale to use for this IDS request.
extra	(Optional) This variable lets you configure this request with an extra variable that may not be accounted for, such as newly-added variables, you want to explicitly configure for IDS.
mapping	(Optional) This lets you map an IDS configuration property to a method in Documaker Interactive. Do not change these values unless you are also changing the code to handle it.
attachmentMapping	(Optional) This lets you map the attachment in an IDS configuration property to a method in Documaker Interactive. Do not change these values unless you are also changing the code to handle it.

Option	Description
response	(Optional) This lets you map a configuration property in an IDS response to a method in Documaker Interactive. Do not change these values unless you are also changing the code to handle it.
responseAttachment	(Optional) This lets you map a configuration property in an IDS response attachment to a method in Documaker Interactive. Do not change these values unless you are also changing the code to handle it.

Configuring Documaker Interactive

The Interactive piece of Documaker Enterprise uses IDS and the Correspondence web application and the Documaker Interactive configuration is contained within those components of Document Factory.

The configuration options for IDS are defined in the DBPOOL group. The configuration information for Correspondence is stored in the sections listed below. Typically, you only need to change the following items in these configuration options:

- The config value
- The IP address and port values for the WIP Edit plug-in and web applications
- The urlText property for the Approval Rules process
- The UCM connection information

Use these sections to configure Documaker Interactive: Correspondence:

BPEL_CLIENT_DATA

Option	Description
Section Name = BPEL_CLIENT_DATA	
(class)	Defines the implementation class. Do not change.
decisionServiceAddress	Defines the decision service URL. Here is an example: http://localhost:8001/soa-infra/services/default/iDMkrApprovalRulesProj/iDMkrApprovalRules_DecisionService_ep
documakerServiceAddress	Defines the Documaker service URL. Here is an example: http://localhost:8001/DWS/CompositionService
urlText	Defines the URL text. Here is an example: http://localhost:8001/soa-infra/services/default/iDMkr_Correspondence/correspondenceprocesses_client_ep?WSDL

DBPOOL:correspondence

Option	Description
Section Name = DBPOOL:correspondence	
platform.credentials	Defines the database password.
platform.driver	Defines the database driver. Here is an example: oracle.jdbc.OracleDriver
platform.principal	Defines the database user name. The default is dmkr_asline, but you should change this ot match your configuration.
platform.url	Defines the database URL. Here is an example: jdbc:oracle:thin:@localhost:1521:IDMAKER

ENTRY_ACTION_FORMS

Option	Description
Section Name = ENTRY_ACTION_FORMS	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
DPRINCLUDECATEGORY	Defines the DPR include category.
DPRSTANDARDINDEX	Defines the DPR standard index.
attachmentMapping	This determines which implementation to use to generate the attachment to send to IDS. Here is an example: XMLIMPORT=getKeysXML

Option	Description
request	Determines which IDS request to call. Here is an example: iDM_GetMRLResource
responseAttachment	Determines which implementation to use for the IDS attachment. Here is an example: DOCUMENTSTREAM=setForms

ENTRY_ACTION_PLUGIN_GETRESOURCE

Option	Description
Section Name = ENTRY_ACTION_PLUGIN_GETRESOURCE	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
request	Determines which IDS request to call. Here is an example: iDM_PluginGetResource
config	Determines which MRL configuration to use.
password	Defines the credentials for this request.
responseAttachment	Determines which implementation to use for the IDS response attachment.Here is an example: DOCUMENTSTREAM=setResource
DPRSTANDARDINDEX	Defines the DPR standard index.

ENTRY_ACTION_PLUGIN_INIT

Defines the parameters and options needed to initialize and define the WIP Edit plug-in.

Option	Description
Section Name = ENTRY_ACTION_PLUGIN_INIT	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
config	Determines which MRL configuration to use.
DPRSTANDARDINDEX	Defines, in milliseconds, the housekeeping ramp up delay.
getScript	On the server side, you can call scripts to do many different jobs. For initializing plug-ins, this script (getpluginresource) is required. Do not change this setting.
httpCookieName	Defines the session or cookie name.
HTTPQUERYSTRING	Defines the HTTP query string code.

Option	Description
HTTPQUERYSTRING1.NAME	Defines the name of the HTTP query string.
installer	Defines the URL that points to the installer. Here is an example: http://localhost/WipEditW32Rel120p00.exe
mapping	Use to map specific IDS properties to an implementation. Here is an example: PUTURL=getPutURL/nGETSCRIPT=getGetScript/ nREFRESHSCRIPT=getRefreshScript/nSCRIPT=getScript/ nUNIQUE_ID=getRecnum/nPRTTYPE=getPrtType/ nSAVE_REQTYPE=getSaveReqType
password	Defines the credentials for this request.
prtType	Defines the print type, such as DPW.
putURL	Defines the posting URL for the plug-in.
refreshScript	The refresh script that makes sure the session does not expire. Here is an example: refresh/debug
request	Defines which IDS request to call. Here is an example: iDM_PluginInit
response	Use to map specific IDS response properties to an implementation. Here is an example: RF_POSTFILE=setRfPostFile/nREMOTEPRINTFILE=setRemotePrintFile/ nRECNUM=setRecnum/nSAVE_REQTYPE=setSaveReqType/ nGETSCRIPT=setGetScript/nPUTURL=setPutURL/ nREFRESHSCRIPT=setRefreshScript/nRECNUM=setRecnum/ nCONFIG=setConfig/nGETSCRIPT=setGetscript/ nPASSWORD=setPassword/nPRTTYPE=setPrttype/ nREQTYPE=setReqtype/nSCRIPT=setScript/nCONFIG=setConfig/ nREQTYPE=setReqtype/nSAVE_REQTYPE=setSaveReqType/ nUSERID=setUserid
responseAttachment	Determines which implementation to use for the IDS attachment. Here is an example: DOCUMENTSTREAM=setDpw
saveReqType	Defines the IDS request name for the save action. Here is an example: iDM_PluginSave
script	Defines the script to use when starting the WIP Edit plug-in. This option tells the system to run this script and save it on the server. Here is an example: pluginsave
username	Defines the IDS user name credentials for this request.

ENTRY_ACTION_PLUGIN_SAVE

Option	Description
Section Name = ENTRY_ACTION_PLUGIN_SAVE	

Option	Description
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
ACTION	Defines the action code.
attachmentMapping	Defines the implementation mapping to build the attachment. Here is an example: XMLIMPORT=getXmlImport
config	Determines which MRL configuration to use.
DPRSTANDARDINDEX	Defines the DPR standard index.
password	Defines the credentials for this request.
prtType	Defines the print type, such as DPW.
request	Determines which IDS request to call. Here is an example: iDM_PluginSave
SAVE_REQTYPE	Defines which IDS request to call for the save. Here is an example: iDM_PluginSave

FORMS_INIT_DATA

Option	Description
Section Name = FORMS_INIT_DATA	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
attachmentMapping	Defines the implementation mapping to build the attachment.
DPRINCLUDECATEGORY	Defines the DPR include category.
DPRSTANDARDINDEX	Defines the DPR standard index.
IDMKR_LOCAL_LANG	Defines the locale for this IDS request. The default is en (English).
request	Determines which IDS request to call. Here is an example: iDM_GetMRLResource
responseAttachment	Determines which implementation to use for the IDS response attachment. Here is an example: DOCUMENTSTREAM=setData
username	Defines the IDS user name credentials for this request.

FORMS_INIT_KEYS

Option	Description
Section Name = FORMS_INIT_KEYS	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
DPRINCLUDECATEGORY	Defines the DPR include category.
DPRSTANDARDINDEX	Defines the DPR standard index.
IDMKR_LOCAL_LANG	Defines the locale for this IDS request. The default is en (English).
request	Determines which MRL configuration to use. Here is an example: iDM_GetMRLResource
responseAttachment	Determines which implementation to use for the IDS response attachment. Here is an example: DOCUMENTSTREAM=setKeys
username	Defines the IDS user name credentials for this request.

FORMS_PREVIEW

Option	Description
Section Name = FORMS_PREVIEW	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
DPRSTANDARDINDEX	Defines the DPR standard index.
allRecipients	Use to print output in PDF format for all recipients. This lets you omit specifying individual recipients.
attachmentMapping	Defines the implementation mapping to build the attachment. Here is an example: XMLIMPORT=getDocuments
config	Determines which MRL configuration to use.
prtType	Defines the print output type, such as PDF or Word. The default is PDF.
request	Determines which IDS request to call. Here is an example: iDM_PreviewForm
response	Use to map specific IDS response properties to an implementation. Here is an example: REMOTEPRINTFILE=setPrintFile
responseAttachment	Determines which implementation to use for the IDS attachment. Here is an example: DOCUMENTSTREAM=setPrintData

Option	Description
username	Defines the IDS user name credentials for this request.

getMRLResourceKeys

Used to load key mappings for the system.

Option	Description
Section Name = getMRLResourceKeys	
(class)	Defines the implementation class (oracle.documaker.ids.bll.IDSGetGroups). Do not change.
(scope)	Defines the application scope. Do not change.
DPRINCLUDECATEGORY	Defines the DPR include category.
DPRSTANDARDINDEX	Defines the DPR standard index.
password	Defines the credentials for this request.
request	Defines which IDS request to call, such as iDM_GetMRLResource
responseAttachment	Determines which implementation to use for the IDS response attachment.

getSSS

This section tells you the version of the application that is running.

Option	Description
Section Name = getSSS	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
DPRSTANDARDINDEX	Defines the DPR standard index.
request	Defines the IDS request name, such as SSS.

PUBLISH_ACTION_PRINT

Option	Description
Section Name = PUBLISH_ACTION_PRINT	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
DPRSTANDARDINDEX	Defines the DPR standard index.

Option	Description
IDMKR_LOCAL_LANG	Defines the locale. The default is en (English).
request	Determines which IDS request to call. Here is an example: i_Print

PUBLISH_ACTION_RUN_RP

This is not used in the default configuration.

Option	Description
Section Name = PUBLISH_ACTION_RUN_RP	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
attachmentMapping	Defines the implementation mapping to build the attachment. Here is an example: XMLIMPORT=getDocuments
CONFIG	Determines which MRL configuration to use.
DPRSTANDARDINDEX	Defines the DPR standard index.
EXTRFILE	Defines the external file location. Here is an example: c:\oracle\documaker\mstrres\dmres\input\extrfile.xml
IDMKR_LOCAL_LANG	Defines the locale for this IDS request. The default is en (English).
KEY1	Defines the Key1 mapping.
KEY2	Defines the Key2 mapping.
KEYID	Defines the KeyID.
PASSWORD	Defines the credentials for this request.
PRINTBATCHES	Defines the print in batches code.
RECTYPE	Defines the record type code.
request	Determines which IDS request to call. Here is an example: iDM_Correspondence_RunRP
USERID	Defines the IDS user name credentials for this request.

SYSTEM_ATTACHMENT_MAPPING

Option	Description
Section Name = SYSTEM_ATTACHMENT_MAPPING	
IDMKR_LOCAL_LANG	Defines the locale for this IDS request. The default is en (English).

Option	Description
XMLIMPORT	Defines the default implementation mapping for IDS XML attachments. Here is an example: getFormsXML
responseAttachment	Defines the default implementation mapping for IDS response XML attachments. Here is an example: DOCUMENTSTREAM=setFormData

UCM_CONNECT

Option	Description
Section Name = UCM_CONNECT	
(class)	Defines the implementation class. Do not change.
connectionString	Defines the UCM (Universal Content Management) connection URL. Here is an example: idc://documakerucm.us.oracle.com:4444
passWord	Defines the credentials for this request.
userName	Defines the IDS user name credentials for this request.

WIP_ACTION_ADD

Option	Description
Section Name = WIP_ACTION_ADD	
(class)	Defines the implementation class. Do not change.
ACTION	Defines the action code to add.
DPRSTANDARDINDEX	Defines the DPR standard index.
UCM_IdcConnection	Defines the UCM (Universal Content Management) IDC connection URL.
assignUserId	Defines the user name to assign.
attachmentMapping	Defines the implementation to build the attachment. Here is an example: XMLIMPORT=getImportFile
currgroup	Defines the current group.
mapping	Use to map specific IDS properties to an implementation. Here is an example: ASSIGNUSERID=getAssignUserId/n
request	Determines which IDS request to call. Here is an example: iDM_AddWIP
response	Use to map specific IDS response properties to an implementation. Here is an example: UNIQUE_ID=setRecnum

WIP_ACTION_EDIT_GETENTRY

Option	Description
Section Name = WIP_ACTION_EDIT_GETENTRY	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
DPRSTANDARDINDEX	Defines the DPR standard index.
fileType	Defines the file type (XML).
prtType	Defines the print type (XML).
request	Determines which IDS request to call. Here is an example: iDM_GetWIPEntry

WIP_ACTION_EDIT_PRINTPROOF

Option	Description
Section Name = WIP_ACTION_EDIT_PRINTPROOF	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
allRecipients	
config	Determines which MRL configuration to use.
dprProofLogo	Use the DPR proof logo.
DPRSTANDARDINDEX	Defines the DPR standard index.
prtType	Defines the print type. The choices are PDF or Word.
request	Determines which IDS request to call. Here is an example: iDM_PrintWIPFormset
response	Use to map specific IDS response properties to an implementation. Here is an example: PRINTFILE=setPrintFile
responseAttachment	Determines which implementation to use for the IDS response attachment. Here is an example: DOCUMENTSTREAM=setPrintData

WIP_ACTION_MODIFY

Option	Description
Section Name = WIP_ACTION_MODIFY	

Option	Description
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
ACTION	Defines the action code to modify.
DPRSTANDARDINDEX	Defines the DPR standard index.
GOCHANGE	Tells IDS to update the WIP indexes. This is a legacy setting.
NEWWIP	Send new WIP.
NEWWIP1.ACTION	Defines whether to send this WIP document as a new WIP. Enter 1 for Yes, or zero (0) for No.
UCM_IdcConnection	Defines the UCM (Universal Content Management) IDC connection URL.
WIPS	Defines which attachment VAR will define the columns to update.
attachmentMapping	Defines the implementation to build the attachment. Here is an example: XMLIMPORT=getImportFile
mapping	Use to map specific IDS properties to an implementation. Here is an example: UNIQUE_ID=getRecNumnWIPS1.UNIQUE_ID=getRecNum/ nCURRUSER=getEntityId
request	Determines which IDS request to call. Here is an example: iDM_ModifyWIPData

WIP_ACTION_PREVIEW

Option	Description
Section Name = WIP_ACTION_PREVIEW	
(class)	Defines the implementation class. Do not change.
(scope)	Defines the application scope. Do not change.
DPRSTANDARDINDEX	Defines the DPR standard index.
request	Determines which IDS request to call.

Chapter 6

Using Documaker Web Services

Documaker Web Services (DWS) are web services that provide support for the latest web service standards.

This chapter discusses these web services in these topics:

CHOOSING THE RIGHT WEB SERVICES

Oracle Documaker offers two different web services applications:

- Enterprise Web Publishing Services (EWPS)
- Documaker Web Services (DWS)

Use this table to determine which web service to use:

Use	To interact with Oracle Documaker...
EWPS	Library resources or transactions in a state of publishing by Documaker Server. These web service methods offer a number of ways to gather information about the MRL, locate documents or field information, and retrieve a form during transaction processing. EWPS also lets you update a document in WIP, publish a document from an extract file or publish a document stored in WIP. See Using Enterprise Web Publishing Services for information about the methods offered with EWPS.
DWS	Document Factory. These web services, introduced in Documaker version 12.0, let you submit a job that tells the system to publish a document from an input or extract file. DWS also provides a generic web service method, doCallIDS, that lets you work with Docupresentment (DS) using specific request types. Because of Documaker Web Services' concrete schema, you should use the doCallIDS method with the Business Process Execution Language (BPEL) to facilitate workflow within the Documaker Interactive: Correspondence application. This method can also be used by BPEL outside of Documaker Interactive: Correspondence or by other web service clients to make specific requests to IDS or Documaker and should be used if your request needs to be asynchronous.

INTRODUCTION TO DWS

Documaker Web Services (DWS) provide ease of integration, interoperability and ease of deployment. Ease of integration is provided by exposing Docupresentment and Document Factory functionality through web service operations that can be used by web service enabled applications, including BPEL.

Interoperability is provided as DWS relies on the JAX-WS framework which supports Web Services Interoperability Technology (WSIT), an open source project started by Sun and co-sponsored by Microsoft to make web services interoperable between Java and .NET Windows Communication Foundation (WCF).

Documaker Web Services are comprised of these types of service operations:

- Composition services
- Publishing services

Composition Services

Composition service operations expose a Docupresentment request type to compose documents. These request types can create different type of documents, including work-in-progress documents for review by policy systems, archive documents, and PDF and other output types from library templates. In addition, the request types provide other base functions you can use to accomplish tasks such as distributing documents through FTP, email, or to a printer. These functions and rules are also extensible through the Docupresentment APIs, so there is no limit as to what Docupresentment can do.

Note Please refer to the *Composition Services* section of this guide for a list of service operations. Please refer to the Internet Document Server Guide and the SDK Reference for more information regarding what Docupresentment does and the bridges and base functions it uses for each request type.

Publishing Services

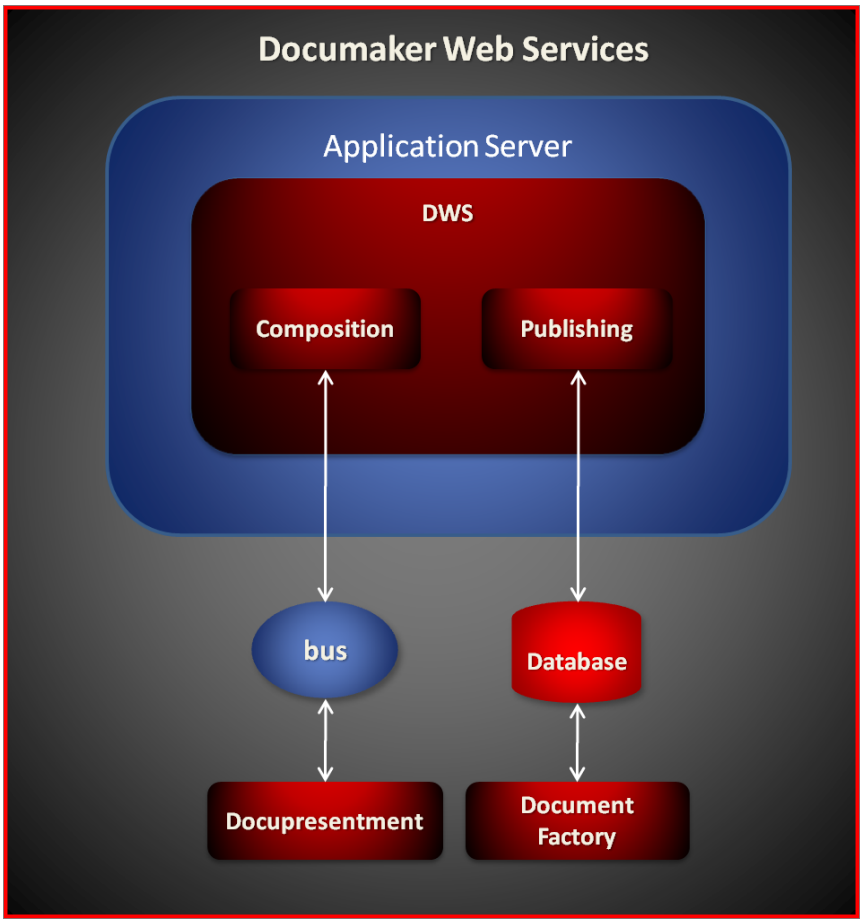
Publishing service operations expose Document Factory functionality and the Documaker core run time to assemble, publish, and distribute documents. The Document Factory is an assembly line of different processes that assemble, publish, or distribute documents at different stages.

Some of these processes also use the Documaker core run time, which provides base rules for this purpose. You can use these rules for a variety of tasks, such as...

- Adding banner pages
- Splitting transactions apart
- Determining which library forms and sections to use
- Determining the order in which library forms and sections are used
- Merging in the data provided in an input extract file into form templates
- Publishing documents as different type print streams

- Distributing print streams to printers, or via SMS, email, and so on

Note See *Using Publishing Services* on page 443 for a list of service operations. Refer to the Documaker Administration Guide for information regarding Documaker.



Web Services Standards

By relying on the JAX-WS framework, Documaker Web Services also take advantage of the WS-* standards provided by the JAX-WS RI, including:

- WS-Addressing
- WS-RM
- WS-Security
- WS-Trust
- WS-SecureConversation
- MTOM

Note Please see the documentation for your application server/container regarding what WS-* standards it supports and how to configure them. For example, if you are interested in WS-Security, then you should see the documentation for your container regarding support for WS-Security and how it is configured.

Components

Here is a list of the components used by Documaker Web Services.

Component	Description
Documaker-WS.jar	Web services package. It contains all classes for the Documaker Web Services Composition and Publishing operations.
Documaker-Schema.jar	Documaker Web Services Schema package. It contains all the XSD schemas and JAXB schema generated classes used by DWS.
Documaker-BUS.jar	Message Bus package. This package contains the message bus functionality to communicate with Docupresentment via JMS, WebSphere MQ, MSMQ or HTTP.
Documaker-Config.jar	Configuration package. Used to retrieve configuration information from the Document Factory administration tables, such as the default message bus for Docupresentment.
Documaker-DAO.jar	DAO (Data Access Objects) package. Used to interface with the Document Factory assembly tables. Can be used to insert or retrieve records from the assembly tables.
Documaker-DB.jar	Database package. It contains JDBC utility functions and routines for generating database agnostic SQL queries.
Documaker-Util.jar	Utility package. It provides IO, Zip, LOGJ and other utility functions.
DWS.war	Documaker Web Services web application archive file. This is the WAR file that is deployed to non-J2EE application servers such as Tomcat.
DWS.ear	Documaker Web Services enterprise application archive file. This is the EAR file that is deployed to J2EE application servers.
DWS-Loader-Catalina.jar	Tomcat-only custom class loader. It provides the ability to override the JAX-WS RI provided by the bootstrap classpath with the JAX-WS RI included in the Tomcat DWS.war file.

Component	Description
tomcat-juli.jar	Tomcat-only custom logger. Used by the custom class loader.
persistence-config.jar	JPA (Java Persistence API) provider configuration - used when JPA is the Configuration implementation used by Documaker Web Services. It provides the JNDI data source to use for retrieving configuration information from the Document Factory administration tables.

USING COMPOSITION SERVICES

Composition service operations expose the Docupresentment request types to compose documents.

Docupresentment

Docupresentment allows high flexibility, customization, and extensibility via its configuration in the docserv.xml file. This configuration file basically contains a list of request types. Each request type contains a list of rules the Docupresentment should run. Composition service operations invoke these request types to compose documents.

Note See the Internet Document Server Guide and the SDK Reference for more information about Docupresentment.

WSDL URLs

Composition service operations are exposed through these URLs:

SOAP version	URL
1.1	http://IpAddress:Port/DWS/CompositionService?WSDL
1.2	http://IpAddress:Port/DWS/CompositionServiceSoap12?WSDL

Where *IpAddress* and *Port* reflect the IP address and port of the application server hosting DWS.

Here is a list of the service operations provided.

Operation	Description
doCallIDS	A web service operation that serves as a Docupresentment client and can submit any request type Docupresentment is configured to support. See *doCallIDS* on page 419 for more information.

Error Handling

Composition services return a CompositionFault SOAP element with a detailed description of the error encountered. For more information, see *CompositionFault Schema* on page 438 and *CompositionFault* on page 442.

CONFIGURING ASSEMBLY LINE FOR DWS

One DWS application instance can only interface with one Document Factory assembly line. To invoke composition service operations, you must first set up the assembly line the DWS application instance should interface with.

This is necessary so composition service operations can retrieve the default Docupresentment message bus configuration from the ALCONFIGCONTEXT Document Factory administration table. This configuration is achieved through web.xml file configuration parameters in WEB-INF directory of the DWS.war file.

Note See *web.xml File* on page 480 for more information on JNDI and the assembly line configuration options.

CONFIGURING THE DOCUPRESENTMENT MESSAGE BUS

Docupresentment uses a message bus to retrieve/return request/response messages from/to client applications. Composition service operations read the default message bus configuration properties for Docupresentment from the bus GROUP_NAME column in the ALCONFIGCONEXT Document Factory administration table. This table is created when a Document Factory assembly line is installed and configured.

Here is an example of the bus properties from ALCONFIGCONTEXT table (only the PROPERTY and VALUE columns are shown):

Property	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	WebLogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://127.0.0.1:7001
jms.qcf.name	jms/qcf
IDSRequestQueue	jms/requestq
IDSResultQueue	jms/resultq
TimeoutSeconds	5

doCallIDS

The doCallIDS web service exposes Docupresentment to compose documents. You can use doCallIDS to invoke any request type or rule that a Docupresentment instance is configured to support.

The doCallIDS service operation provides name/value VAR schema element pairs in the request payload. These are used to provide the Docupresentment request type value as well as any other name/value pairs the individual functions might expect.

Here is an example of a request type in the docserv.xml configuration file for the Docupresentment. The request type name or what it does is not important in this example; what is important is that this is what a Docupresentment request type looks like and how it can be invoked from doCallIDS service operation.

```
<section name="ReqType:SSS">
    <entry name="function">atcw32->ATCLoadAttachment</entry>
    <entry name="function">atcw32->ATCUnloadAttachment</entry>
    <entry name="function">irlw32->IRLStatistics</entry>
    <entry name="function">irlw32->IRLSendVersion</entry>
    <entry name="function">dprw32->DPRSendVersion</entry>
</section>
```

Note See the Internet Document Server Guide and the SDK Reference for more information regarding specific request types and rules.

Here is an example of a doCallIDS request payload that invokes the Docupresentment SSS request type. In this example, the rules listed for the SSS request type do not expect any input name/value pairs so only the ReqType variable is submitted along with the value SSS. Other request types and rules may expect different input name/value pairs. See the SDK Reference for a description of the input name/value pairs each base rule expects.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cmn="oracle/documaker/schema/common"
xmlns:compcmn="oracle/documaker/schema/ws/composition/common"
xmlns:req="oracle/documaker/schema/ws/composition/doCallIDS/v1/
request"
xmlns:tns="oracle/documaker/schema/ws/composition"
xmlns:v1="oracle/documaker/schema/ws/composition/doCallIDS/v1">
  <soap:Body>
    <tns:doCallIDSRequest>
      <tns:doCallIDSRequestV1>
        <compcmn:timeoutMillis>30000</compcmn:timeoutMillis>
        <v1:IDSRequest>
          <req:DSIMSG>
            <compcmn:MSGVARS>
              <compcmn:VAR NAME="ReqType">sss</compcmn:VAR>
            </compcmn:MSGVARS>
          </req:DSIMSG>
        </v1:IDSRequest>
        <v1:ResponseProperties/>
      </tns:doCallIDSRequestV1>
    </tns:doCallIDSRequest>
  </soap:Body>
</soap:Envelope>
```

Overriding the Default Message Bus

The default message bus properties that are read from the bus GROUP_NAME column in ALCONFIGCONTEXT Document Factory administration table can be overridden at the request payload level so the doCallIDS web service operation can invoke different Docupresentment instances. This is done via the Properties schema element, which can contain one of these elements:

- HTTP
- MQ
- MSMQ
- JMS

The Properties schema element and all other schema elements are described in detail in the following topics. Here is an example of a request payload that uses the Properties and JMS elements to communicate with Docupresentment and override the default properties in the Bus section in the ALCONFIGCONTEXT Document Factory administration table:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cmn="oracle/documaker/schema/common"
xmlns:compcmn="oracle/documaker/schema/ws/composition/common"
xmlns:req="oracle/documaker/schema/ws/composition/doCallIDS/v1/
request"
xmlns:tns="oracle/documaker/schema/ws/composition"
xmlns:v1="oracle/documaker/schema/ws/composition/doCallIDS/v1">
  <soap:Body>
    <tns:doCallIDSRequest>
      <tns:doCallIDSRequestV1>
        <compcmn:timeoutMillis>30000</compcmn:timeoutMillis>
        <v1:Properties>
          <v1:JMS>
            <cmn:queuefactory.class>com.docucorp.messaging.jms.DSIJM
SJNDIMessageQueueFactory</cmn:queuefactory.class>
            <cmn:jms.initial.context.factory>weblogic.jndi.WLInitial
ContextFactory</cmn:jms.initial.context.factory>
            <cmn:jms.provider.URL>t3://df121x64:7001</
cmn:jms.provider.URL>
            <cmn:jms.qcf.name>jms/al1/qcf</cmn:jms.qcf.name>
            <cmn:jms.inputqueue.connectstring>jms/al1/idsres</
cmn:jms.inputqueue.connectstring>
            <cmn:jms.outputqueue.connectstring>jms/al1/idsreq</
cmn:jms.outputqueue.connectstring>
            <compcmn:marshaller.class>com.docucorp.messaging.data.ma
rshaller.SOAPMIMEDSIMessageMarshaller</compcmn:marshaller.class>
          </v1:JMS>
        </v1:Properties>
        <v1:IDSRequest>
          <req:DSIMSG>
            <compcmn:MSGVARS>
              <compcmn:VAR NAME="ReqType">sss</compcmn:VAR>
            </compcmn:MSGVARS>
          </req:DSIMSG>
        </v1:IDSRequest>
      </tns:doCallIDSRequestV1>
```

```
    </tns:doCallIDSRequest>
  </soap:Body>
</soap:Envelope>
```

Sending and Receiving File Attachments

Certain rules in Docupresentment expect input file attachments or return output file attachments. The doCallIDS web service operation provides the ability to submit or retrieve these file attachments.

For example, here is a Docupresentment request type that expects an input file attachment of name *EXTRACTFILE* and returns and output file attachment of name *RPOUTPUT*:

```
<section name="ReqType:RPDRUNRP">
    <entry name="function">atcw32->ATCLogTransaction</entry>
    <entry name="function">atcw32->ATCLoadAttachment</entry>
    <entry name="function">atcw32->ATCUnloadAttachment</entry>
    <entry name="function">atcw32-
>ATCSendFile,RPOUTPUT,Printer1,BINARY</entry>
    <entry name="function">atcw32-
>ATCReceiveFile,EXTRACTFILE,EXTRFILE,*.xml,KEEP</entry>
    <entry name="function">dprw32->DPRSetConfig</entry>
    <entry name="function">RPDW32->RPDCheckRPRun</entry>
    <entry name="function">RPDW32->RPDCreateJob</entry>
    <entry name="function">RPDW32->RPDProcessJob</entry>
</section>
```

Here is an example of the corresponding doCallIDS request payload that invokes the RPDRUNRP request type and submits an input file attachment of name EXTRACTFILE and expects an output file attachment of name RPOUTPUT.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cmn="oracle/documaker/schema/common"
xmlns:compcmn="oracle/documaker/schema/ws/composition/common"
xmlns:req="oracle/documaker/schema/ws/composition/doCallIDS/v1/
request"
xmlns:tns="oracle/documaker/schema/ws/composition"
xmlns:v1="oracle/documaker/schema/ws/composition/doCallIDS/v1">
  <soap:Body>
    <tns:doCallIDSRequest>
      <tns:doCallIDSRequestV1>
        <compcmn:timeoutMillis>30000</compcmn:timeoutMillis>
        <v1:IDSRequest>
          <req:DSIMSG>
            <compcmn:MSGVARS>
              <compcmn:VAR NAME="ReqType">RPDRUNRP</compcmn:VAR>
              ...
              ...
            </compcmn:MSGVARS>
            <compcmn:Attachment>
              <cmn:Name>EXTRACTFILE</cmn:Name>
              <cmn:Content>
                <cmn:Binary>UEsDBAoAAAA...</cmn:Binary>
              </cmn:Content>
            </compcmn:Attachment>
          </req:DSIMSG>
        </v1:IDSRequest>
        <v1:ResponseProperties>
```

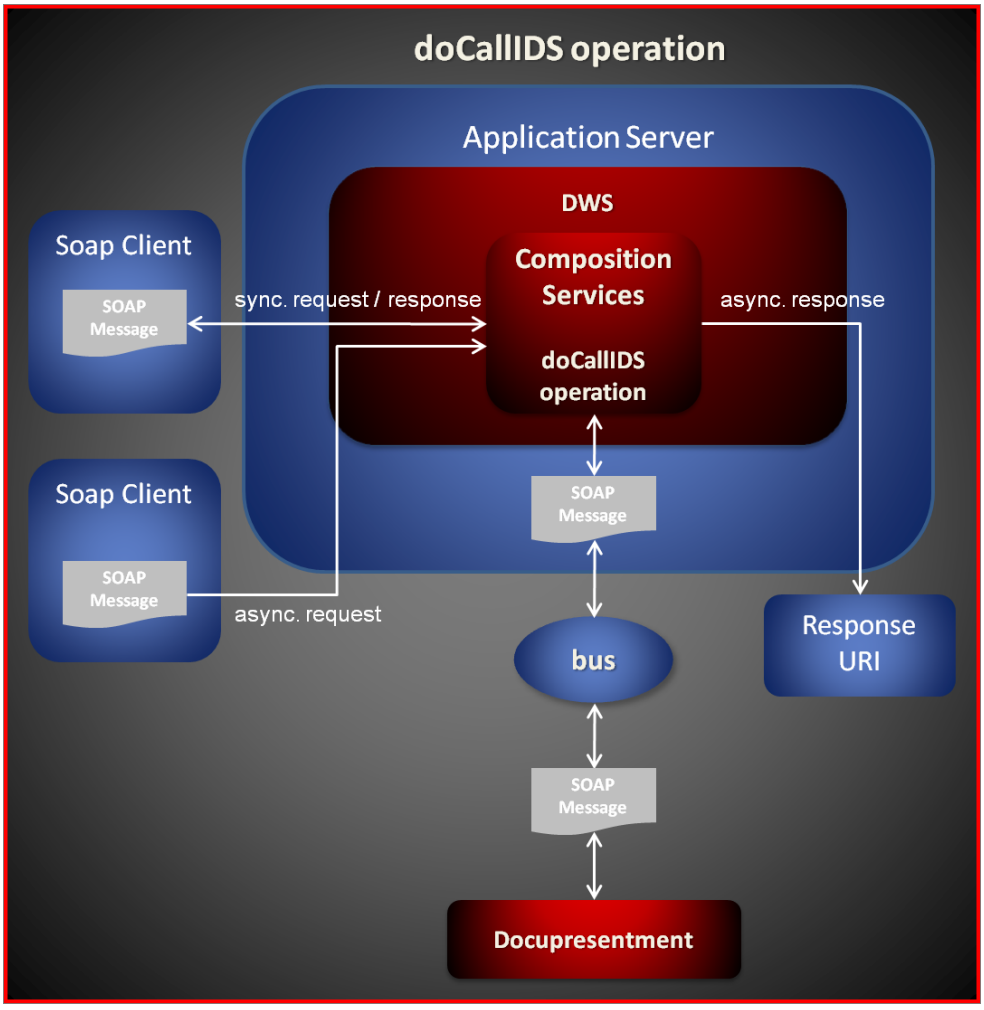
```
            <v1:ResponseAttachment>
              <cmn:Name>RPOUTPUT</cmn:Name>
              <cmn:ReturnType>Binary</cmn:ReturnType>
            </v1:ResponseAttachment>
          </v1:ResponseProperties>
        </tns:doCallIDSRequestV1>
      </tns:doCallIDSRequest>
    </soap:Body>
</soap:Envelope>
```

This example shows an input file named EXTRACTFILE is sent as base64-encoded content (most of the base64-encoded data was omitted for brevity). The example also shows the service operation expects a file attachment of name RPOUTPUT in the response message which is to be returned as binary base64-encoded content.

Synchronous vs. Asynchronous Responses

The doCallIDS web service operation can run as synchronous or asynchronous based on WS-Addressing headers provided in the request payload.

When running as	This happens
A synchronous operation	doCallIDS waits for a response message to be returned to the message bus by Docupresentment and then returns it to the SOAP client.
An asynchronous operation	The web service client submits the request along with a response URI (Uniform Resource Identifier) in a WS-Addressing header and does not wait for the response. The doCallIDS web service operation sends the response message to the URI provided when finished.



Message Schema

The doCallIDS web service operation request and response SOAP messages use the same DSIMSG schema element.

Discussions of the schema elements follow. The Type/Count column in each of these schema tables describes the schema type and occurrence. The schema type can refer to other custom schema types.

If the count is defined as	It means the element is
one (1)	Required.
(0...1)	Optional.
(0...many) or (1...many)	Optional, but more than one element of this type can exist. or Required, but more than one element of this type can exist.

Certain schema elements are defined as (choice) and then contain a list of elements. This means one, but no more than one, of the elements in the list can be used. This is standard schema nomenclature.

Note Some of the schema elements described in these sections, such as DSIMSG, MSGVARS, VAR, ROWSET, ROW, are described in the *Customizing Your System* topic in the Internet Document Server Guide. Please refer to that document for more information.

Here is a list of the doCallIDS schema elements:

doCallIDSRequest

DWS provides web service versioning at the message level. The doCallIDSRequest element contains a schema choice element that provides the ability to select different versions of a request message.

Element	Description	Type/Count
(choice)	Contains one of these elements: doCallIDSRequestV1	choice (1)

doCallIDSRequestV1

The doCallIDSRequestV1 element is the first message version of doCallIDSRequest element. It contains these elements:

Element	Description	Type/Count
timeoutMillis	Specifies how long the service operation should wait for a reply message from Docupresentment. The default is 30,000 milliseconds.	int (1)
Properties	Provides the message bus configuration options that can be used to communicate with Docupresentment. This element overrides the default message bus configuration.	Properties (0...1)
IDSRequest	Contains the request payload for an Internet Docupresentment Server request.	IDSRequest (1)
ResponseProperties	A response properties element that defines how attachments should be returned.	ResponseProperties (0...1)

Properties

Can be used to override the default message bus properties configured in ALCONFIGCONTEXT administration table. This element provides the ability to configure each request payload to talk to a different Docupresentment instance.

Element	Description	Type/Count
(choice)	Contains one of these elements: HTTP MQ MSMQ JMS	choice (1)

Note The web service supports the HTTP, MQ, MSMQ, and JMS options used by Docupresentment and documented in the Internet Document Server Guide. Also, the web service operation WSDL displays defaults for each option selected.

HTTP

A set of HTTP message bus configuration options for communicating with Docupresentment.

Element	Description	Type/Count
queue.factory.class	The fully-qualified class name of the HTTP queue factory class to use. The value of this element is final: com.docucorp.messaging.http.DSIHTTPMessageQueueFactory	string (1)
marshaller.class	The fully-qualified class name of the marshaller class to use. The value of this element is final: com.docucorp.messaging.data.marshaller.SOAPMIMEDSIMessageMarshaller	string (1)
http.url	The URL of the Docupresentment HTTP router or server. The default is http://localhost:49152	string (1)
http.reuse.ports	This option determines if any opened ports should be reused by the client. The default is Yes.	string (0...1)
http.putmessage.tries	How many put message attempts should be made by the client when an error occurs. The default is three (3).	string (0...1)

Note These options are also documented in the Internet Document Server Guide.

MQ

A set of WebSphere MQ message bus configuration options for communicating with Docupresentment.

Element	Description	Type/Count
queue.factory.class	The fully-qualified class name of the MQ queue factory class to use. The value of this element is final: com.docucorp.messaging.mqseries.DSIMQMessageQueueFactory	string (1)

* Only used when *com.docucorp.messaging.mqseries.DSIMQSSLsocketFactory* is specified as the value of the mq.ssl.socketFactory.class option. SSL options should only be used if the queue manager has been configured to support SSL.

Element	Description	Type/Count
marshaller.class	The fully-qualified class name of the marshaller class to use.	MarshallerClass (1)
mq.queue.manager	The name of the MQ queue manager. The value is case-sensitive. The default is queue_manager.	string (1)
mq.tcpip.host	The host name or IP address of the server where the MQ queue manager resides. Omit this option to use a bindings mode connection. Include this option to use client mode.	string (1)
mq.tcpip.port	The port number the MQ queue manager is listening on. Omit this option to use a bindings mode connection. Include this option to use client mode. The default is 1414.	string (0...1)
mq.inputqueue.name	The name of the input queue. The input queue is the queue that is used to read reply messages from Docupresentment, meaning it is the output queue on the Docupresentment server side. The value is case-sensitive. The default is RESULTQ.	string (1)
mq.outputqueue.name	The name of the output queue. The output queue is the queue that is used to send request messages to Docupresentment, meaning it is the input queue on the Docupresentment server side. The value is case-sensitive. The default is REQUESTQ.	string (1)
mq.queue.channel	The name of the MQ Server Connection Channel to use. Omit this option to use a bindings mode connection. Include this option to use client mode. The value is case-sensitive. The default is SYSTEM.DEF.SVRCONN.	string (0...1)
mq.outputqueue.expiry	How long should a message placed in MQ stay around. The default is 1800 seconds. Enter a value of -1 to indicate the message never expires.	string (0...1)
mqseries.exception.logging	This option enables exception logging at the WebSphere MQ level. Acceptable values are Yes or No. The default is Yes.	string (0...1)
mqseries.tracing	Sets the WebSphere MQ tracing level. MQSeriesTracing (1...4), 1 being the lowest level of tracing.	string (0...1)
mqseries.log	Sets the location and name of the WebSphere MQ log file to use when mqseries.exception.logging and/or mqseries.tracing options are enabled.	string (0...1)

* Only used when *com.docucorp.messaging.mqseries.DSIMQSSLsocketFactory* is specified as the value of the mq.ssl.socketFactory.class option. SSL options should only be used if the queue manager has been configured to support SSL.

Element	Description	Type/Count

mq.ccdt.url	This value should contain the URL of a client connection definition table (CCDT) that should be used to derive all the connection information for this factory.
	This property and the mq.queue.channel property are mutually exclusive. If you define both you get an MQSeries 2423 MQRC error. Here are some examples of URL values:
	`file:///c:/mq/ccdt/AMQCLCHL.TAB`
	`file:/c:/mq/ccdt/AMQCLCHL.TAB`
	`ftp://userName:password@myServer/ccdt_files/AMQCLCHL.TAB`
		string (0...1)

mq.ssl.cipherspec	The encryption and hashing algorithm used for SSL communications.	MQSSLCipherspec (0...1)
mq.ssl.peername	The distinguished name (DN) pattern of the SSL certificate used by the queue manager. This is used to validate the queue manager.	string (0...1)
mq.ssl.socketFactory.class	The name of a custom SSL socket factory class that should be used to override the default SSL socket factory used by WebSphere MQ - javax.net.ssl.SSLsocketFactory. This value should contain the package and class name of an SSL socket factory class that extends the javax.net.ssl.SSLsocketFactory class. There is no default value for this property. If this property is not specified and SSL support is enabled, WebSphere MQ uses the javax.net.ssl.SSLsocketFactory class and looks for the java key and trust stores in this way: • Look for them in System properties javax.net.ssl.keyStore and javax.net.ssl.trustStore • Look for their passwords in System properties javax.net.ssl.keyStorePassword and javax.net.ssl.trustStorePassword. If the system properties are not defined, the system looks in the default keyStore/trustStore (named *cacerts*) located in JAVA_HOME\jre\lib\security directory and uses the default password (*changeit*) for them. If you need to load your own keyStore and trust store and do not want to use the system properties, you can define your own SSLsocketFactory class or use the com.docucorp.messaging.mqseries.DSIMQSSLsocketFactory class in DocucorpMsg.jar package by providing the appropriate value for this property. See also the mq.ssl.protocol, mq.ssl.keyStore, mq.ssl.keyStore.type, mq.ssl.keyStore.manager.type, mq.ssl.keyStore.pwd, mq.ssl.trustStore, mq.ssl.trustStore.type, mq.ssl.trustStore.manager.type, and mq.ssl.trustStore.pwd properties.	string (0...1)
mq.ssl.protocol *	The SSL protocol to use with a custom SSL socket factory class. The default is SSLv3.	string (0...1)
mq.ssl.keyStore *	The path and file name of the Java key store where the private keys and public certificates are stored.	string (0...1)

* Only used when *com.docucorp.messaging.mqseries.DSIMQSSLsocketFactory* is specified as the value of the mq.ssl.socketFactory.class option. SSL options should only be used if the queue manager has been configured to support SSL.

Element	Description	Type/Count
mq.ssl.keyStore.type *	mq.ssl.keyStore.type *	string (0...1)
mq.ssl.keyStore.manager.type *	The key store manager type. The default is SunX509.	string (0...1)
mq.ssl.keyStore.pwd *	The password for the SSL key store.	string (0...1)
mq.ssl.trustStore *	The path and file name of the java trust store where the trusted public certificates are stored.	string (0...1)
mq.ssl.trustStore.type *	The type of trust store used. The default is JKS (Java Key Store).	string (0...1)
mq.ssl.trustStore.manager.type *	The trust manager type. The default is SunX509.	string (0...1)
mq.ssl.trustStore.pwd	The password for the SSL trust store.	string (0...1)
mq.ssl.debug	A value of Yes or No enables debug for the SSL session. This is a system-wide (global) property. The default is No.	string (0...1)
mq.Property	Use this option to supply additional MQ provider specific properties. This option is reserved for future use.	Property (0...1)

* Only used when *com.docucorp.messaging.mqseries.DSIMQSSLsocketFactory* is specified as the value of the mq.ssl.socketFactory.class option. SSL options should only be used if the queue manager has been configured to support SSL.

Note These options are also documented in the Internet Document Server Guide in the *Using WebSphere MQ* topic.

MQSeriesTracing

Element	Description	Type/Count
(enum)	The tracing level for WebSphere MQ code. Acceptable values for this option are: • 1 (lowest) • 2 • 3 • 4 (highest)	int (1)

MQSSLCipherspec

Element	Description	Type/Count
(enum)	The SSL encryption and hashing algorithm for WebSphere MQ. Acceptable values for this option are: • DES_SHA_EXPORT • DES_SHA_EXPORT1024 • NULL_MD5 • NULL_SHA • RC2_MD5_EXPORT • RC4_56_SHA_EXPORT1024 • RC4_MD5_US • RC4_MD5_EXPORT • RC4_SHA_US • TRIPLE_DES_SHA_US	string (1)

MSMQ

A set of MSMQ message bus configuration options for communicating with Docupresentment.

Element	Description	Type/Count
queue.factory.class	The fully-qualified class name of the MSMQ queue factory class to use. The value of this element is final: com.docucorp.messaging.msmq.DSIMSMQMessageQueueFactory	string (1)
marshaller.class	The fully-qualified class name of the marshaller class to use.	MarshallerClass (1)
msmq.server.name	The IP address or server name for the MSMQ server. This property is not used when direct format names are used for the input and output queues.	string (0...1)
msmq.inputqueue.name	The name of the input queue. This can be a queue path name or a direct format name. Queue path names are used with the msmq.server.name property and therefore should not include the server name. Direct format names do not use the msmq.server.name property. The default is DIRECT=OS:localhost\PRIVATE$\RESULTQ	string (1)
msmq.outputqueue.name	The name of the output queue. This can be a queue path name or a direct format name. Queue path names are used with the msmq.server.name property and should not include the server name. Direct format names do not use the msmq.server.name property. The default is DIRECT=OS:localhost\PRIVATE$\REQUESTQ	string (1)
msmq.timeout	The timeout interval in milliseconds. This defines how long the system should wait for a message to reach a queue during a send operation. The default is 30000, which equals 30 seconds.	string (0...1)
msmq.expiry	How long a message should remain in the queue before it is deemed expired. This value is used during a send operation. The default is one (1). 800000 ms = 30 minutes.	string (0...1)

Element	Description	Type/Count
msmq.Property	Use this option to supply additional MSMQ provider specific properties. This option is reserved for future use.	string (0...many)

Note These options are also documented in the Internet Document Server Guide in the *Using MSMQ* topic

JMS

A set of JMS message bus configuration options for communicating with Docupresentment.

Element	Description	Type/Count
queue.factory.class	The fully-qualified class name of the JMS JNDI queue factory class to use. The value of this element is final: com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory	string (1)
marshaller.class	The fully-qualified class name of the marshaller class to use.	MarshallerClass (1)
jms.initial.context.factory	The fully-qualified class name of the JMS provider initial context factory. The default is weblogic.jndi.WLInitialContextFactory	string (1)
jms.provider.URL	The JMS provider URL. The default is t3://localhost:7001	string (1)
jms.qcf.name	The JNDI name of the queue connection factory. The default is qcf.	string (1)
jms.inputqueue.connectstring	The JNDI name of the input queue. The default is resultq.	string (1)
jms.outputqueue.connectstring	The JNDI name of the output queue. The default is requestq.	string (1)
jms.security.principal	The account to use when authentication is required.	string (0...1)
jms.security.credentials	The account password to use when authentication is required.	string (0...1)
jms.env.Property	An additional JNDI context property.	Property (0...many)

Note These options are also documented in the Internet Document Server Guide in the *Using the Java Message Service (JMS)* topic.

MarshallerClass

This element provides a selection of the supported Docupresentment message marshallers.

Element	Description	Type/Count
(enum)	The marshaller class used to format a message to/from Docupresentment. Acceptable values for this option are: com.docucorp.messaging.data.marshaller.SerializationDSIMessageMarshaller com.docucorp.messaging.data.marshaller.SOAPMIMEDSIMessageMarshaller	string (1)

Property

Represents a property name/value pair.

Element	Description	Type/Count
Name	The name of the property.	string (1)
Value	The value of the property.	string (1)

IDSRequest

The IDSRequest element contains the DSIMSG element.

Element	Description	Type/Count
DSIMSG	Contains a MSGVARS element.	DSIMSG (1)

ResponseProperties

This element indicates how file attachments should be returned in the response message.

Element	Description	Type/Count
ResponseAttachment	Represents a response attachment element.	ResponseAttachment (0...many)

ResponseAttachment

Element	Description	Type/Count
Name	The name of the attachment that is returned by Docupresentment. This name must match the actual attachment name in the Docupresentment SOAP message. See the example in *Sending and Receiving File Attachments* on page 421 and the *Customizing Your System* topic in the Internet Document Server Guide.	string (1)
ReturnType	Represents an AttachmentReturnType element.	AttachmentReturnType (1)
URI	The file URI to save the return attachment to. Used only when ReturnType element has a value of URI.	string (0...1)

AttachmentReturnType

Represents a file attachment return type choice.

Element	Description	Type/Count
(enum)	Indicates how a file attachment should be returned. Specify one of these options: • URI • Binary	One of these options: URI (1) Binary (1)

doCallIDSResponse

DWS provides web service versioning at the message level. The doCallIDSResponse element contains a schema choice element that lets you select different versions of a response message. A response message, however, will always contain the appropriate message version to match the version in the request message invocation.

Element	Description	Type/Count
(choice)	Contains one of these elements: doCallIDSResponseV1	choice (1)

doCallIDSResponseV1

The doCallIDSResponseV1 element is the first message version of doCallIDSResponse element. It contains these elements:

Element	Description	Type/Count
Result	An integer value that defines the overall result of the service operation. Zero (0) means success. One (1) means failure.	Result (1)
ServiceTimeMillis	How long the service operation took to execute. The elapsed time is returned in milliseconds.	int (1)
IDSResponse	Contains the response payload for an Internet Docupresentment Server response.	IDSResponse (1)
Results	Contains the result code and possibly any error codes returned by a Docupresentment transaction.	Results (1)
ServiceInfo	Returns information about the invoked service operation.	ServiceInfo (1)

IDSResponse

The IDSResponse element contains the DSIMSG element.

Element	Description	Type/Count
DSIMSG	Contains a MSGVARS element.	DSIMSG (1)

DSIMSG

The main element of a Docupresentment SOAP message.

Element	Description	Type/Count
VAR	Represents a name/value pair.	VAR (0...many)
ROWSET	Represents a collection of ROW elements. A ROWSET is basically one or more rows, each row containing one or more name/value pairs.	ROWSET (0...many)

VAR

Represents a name/value pair.

Element	Description	Type/Count
NAME *	The name of the name/value pair.	string (1)
(TextNode) +	The value of the name/value pair.	string (0...1)

* = attribute
+ = text node

ROWSET

Contains one or more rows and each row can contain one or more name/value pairs. A ROWSET is basically one or more rows, each row containing one or more name/value pairs.

Element	Description	Type/Count
NAME *	The name of the row set.	string (1)
ROW	This element represents a row.	ROW (0...many)

* = attribute

ROW

Contains one or more name/value pairs.

Element	Description	Type/Count
NUM *	The row number.	int (1)
VAR	Represents a name/value pair.	VAR (0...many)

* = attribute

Attachment

Represents a file attachment.

Element	Description	Type/Count
Name *	The name of the attachment.	string (0...1)

* = attribute

Element	Description	Type/Count
Content	Represents the file attachment content.	Content (1)

* = attribute

Content

Represents the content of a file attachment.

Element	Description	Type/Count
URI *	A file URI. HTTP URIs are also supported for input request payloads.	string(1)
Binary *	The binary content of the file attachment.	base64Binary (1)

* = URI and Binary elements are mutually exclusive.

Results

Contains the results returned by Docupresentment after processing the request payload.

Element	Description	Type/Count
Result	Represents the result of the invoked Docupresentment transaction.	Result (1)
Errors	Depicts any errors returned by the invoked Docupresentment transaction.	Errors (1)

Result

Contains the result code returned by Docupresentment.

Element	Description	Type/Count
(enum)	Indicates the result of the invoked Docupresentment transaction. Acceptable values are: • 1 (error) • 0 (success)	int (1)

Errors

Contains any error information returned by Docupresentment.

Element	Description	Type/Count
Error	Represents an error returned by the Docupresentment transaction.	Error (0...many)

Error

Represents an error returned by Docupresentment.

Element	Description	Type/Count
Code	The error code returned by Docupresentment.	string (1)
Severity	The severity of the error encountered by Docupresentment.	string (1)
Category	The category of the error encountered by Docupresentment.	string (1)
Description	The description of the error.	string (1)
Diagnosis	A diagnosis element.	Diagnosis (0...many)

Diagnosis

Contains diagnostic information returned by Docupresentment.

Element	Description	Type/Count
Cause	A possible cause of the error.	string (1)
Remedy	A possible resolution for the error.	string (1)

ServiceInfo

Contains information pertaining the service operation invoked.

Element	Description	Type/Count
Operation	The name of the web service operation invoked.	string (1)
Version	Contains information about the version of the service operation invoked.	Version (1...many)

Version

Contains information pertaining the version of the service operation invoked.

Element	Description	Type/Count
Number	The service version number	int (1)
Used	A boolean value that indicates if the current version was used during the service operation invocation. True means this version was used.	boolean (1)

DSIMSG

The main element of a Docupresentment SOAP message.

Element	Description	Type/Count
VAR	Represents a name/value pair.	VAR (0...many)
ROWSET	Represents a collection of ROW elements. A ROWSET is basically one or more rows, each row containing one or more name/value pairs.	ROWSET (0...many)

VAR

Represents a name/value pair.

Element	Description	Type/Count
NAME *	The name of the name/value pair.	string (1)
(TextNode) +	The value of the name/value pair.	string (0...1)

* = attribute
+ = text node

ROWSET

Contains one or more rows and each row can contain one or more name/value pairs. A ROWSET is basically one or more rows, each row containing one or more name/value pairs.

Element	Description	Type/Count
NAME *	The name of the row set.	string (1)
ROW	This element represents a row.	ROW (0...many)

* = attribute

ROW

Contains one or more name/value pairs.

Element	Description	Type/Count
NUM *	The row number.	int (1)
VAR	Represents a name/value pair.	VAR (0...many)

* = attribute

Attachment

Represents a file attachment.

Element	Description	Type/Count
Name *	The name of the attachment.	string (0...1)
Content	Represents the file attachment content.	Content (1)

* = attribute

Content

Represents the content of a file attachment.

Element	Description	Type/Count
URI *	A file URI. HTTP URIs are also supported for input request payloads.	string(1)
Binary *	The binary content of the file attachment.	base64Binary (1)

* = URI and Binary elements are mutually exclusive.

Error Handling

The doCallIDS service operation returns a Composition Fault Exception when there is an error.

CompositionFault Schema

Element	Description	Type/Count
faultInfo	Detailed information about the error. Usually a stack trace.	string (1)
message	Brief information about the error. Usually an application generated message.	string (1)

Note For an example, see *CompositionFault* on page 442.

Example Payloads

Here are examples of the Request, Response, and CompositionFault payloads:

Request Payload

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cmn="oracle/documaker/schema/common"
xmlns:compcmn="oracle/documaker/schema/ws/composition/common"
xmlns:req="oracle/documaker/schema/ws/composition/doCallIDS/v1/
request"
xmlns:tns="oracle/documaker/schema/ws/composition"
xmlns:v1="oracle/documaker/schema/ws/composition/doCallIDS/v1">
  <soap:Body>
    <tns:doCallIDSRequest>
      <tns:doCallIDSRequestV1>
        <compcmn:timeoutMillis>30000</compcmn:timeoutMillis>
        <v1:Properties>
          <v1:HTTP>
            <cmn:queuefactory.class>com.docucorp.messaging.http.DSIH
TTPMessageQueueFactory</cmn:queuefactory.class>
            <cmn:http.url>http://localhost:49952</cmn:http.url>
            <compcmn:marshaller.class>com.docucorp.messaging.data.ma
rshaller.SOAPMIMEDSIMessageMarshaller</compcmn:marshaller.class>
          </v1:HTTP>
        </v1:Properties>
        <v1:IDSRequest>
          <req:DSIMSG>
```

```
            <compcmn:MSGVARS>
              <compcmn:VAR NAME="ReqType">sss</compcmn:VAR>
            </compcmn:MSGVARS>
          </req:DSIMSG>
        </v1:IDSRequest>
      </tns:doCallIDSRequestV1>
    </tns:doCallIDSRequest>
  </soap:Body>
</soap:Envelope>
```

Response Payload

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns5:doCallIDSResponse xmlns:ns6="oracle/documaker/schema/ws/
composition/doCallIDS/v1/response"
    xmlns:ns5="oracle/documaker/schema/ws/composition"
    xmlns:ns4="oracle/documaker/schema/ws/composition/doCallIDS/v1/
request"
    xmlns:ns3="oracle/documaker/schema/ws/composition/doCallIDS/v1"
    xmlns:ns2="oracle/documaker/schema/common"
    xmlns="oracle/documaker/schema/ws/composition/common">
      <ns5:doCallIDSResponseV1>
        <Result>0</Result>
        <ServiceTimeMillis>234</ServiceTimeMillis>
        <ns3:IDSResponse>
          <ns6:DSIMSG>
            <MSGVARS>
              <VAR NAME="ERRORCOUNT">0</VAR>
              <VAR NAME="IDSGUID">2FE81924-6B4A-C4B0-301C-
06B2740B564B</VAR>
              <VAR NAME="IDSHOSTNAME">df121x64</VAR>
              <VAR NAME="LASTRESTART">Apr 28, 2011 2:39:59 PM EDT</
VAR>
              <VAR NAME="RESTARTCOUNT">0</VAR>
              <VAR NAME="RESULTS">SUCCESS</VAR>
              <VAR NAME="SERVERTIMESPENT">0.016</VAR>
              <VAR NAME="SUCCESSCOUNT">1</VAR>
              <VAR NAME="UPTIME">Apr 28, 2011 2:39:59 PM EDT</VAR>
              <VAR NAME="WARNINGCOUNT">0</VAR>
              <ROWSET NAME="LIBRARIES">
                <ROW NUM="1">
                  <VAR NAME="DATE">Feb 24 2011</VAR>
                  <VAR NAME="NAME">dsicrule</VAR>
                  <VAR NAME="TIME">22:15:18</VAR>
                  <VAR NAME="VERSION">200.023.001</VAR>
                </ROW>
                <ROW NUM="2">
                  <VAR NAME="DATE">Feb 24 2011</VAR>
                  <VAR NAME="NAME">DSIOS2</VAR>
                  <VAR NAME="TIME">22:18:02</VAR>
                  <VAR NAME="VERSION">200.023.002</VAR>
                </ROW>
                <ROW NUM="3">
                  <VAR NAME="DATE">Feb 24 2011</VAR>
                  <VAR NAME="NAME">dsijava</VAR>
                  <VAR NAME="TIME">22:18:28</VAR>
                  <VAR NAME="VERSION">200.023.001</VAR>
                </ROW>
                <ROW NUM="4">
                  <VAR NAME="DATE">Feb 24 2011</VAR>
                  <VAR NAME="NAME">jexec</VAR>
                  <VAR NAME="TIME">22:15:13</VAR>
```

```
  <VAR NAME="VERSION">200.023.001</VAR>
</ROW>
<ROW NUM="5">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">ARC</VAR>
  <VAR NAME="TIME">20:34:15</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="6">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">CAR</VAR>
  <VAR NAME="TIME">20:21:03</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="7">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">CRM</VAR>
  <VAR NAME="TIME">20:34:19</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="8">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">DAL</VAR>
  <VAR NAME="TIME">20:35:08</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="9">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">DB</VAR>
  <VAR NAME="TIME">20:19:32</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="10">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">DPR</VAR>
  <VAR NAME="TIME">20:46:20</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="11">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">DS</VAR>
  <VAR NAME="TIME">20:20:02</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="12">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">DTBL</VAR>
  <VAR NAME="TIME">20:28:58</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="13">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">DXM</VAR>
  <VAR NAME="TIME">20:17:52</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="14">
  <VAR NAME="DATE">Apr 19 2011</VAR>
  <VAR NAME="NAME">FAP</VAR>
  <VAR NAME="TIME">20:19:13</VAR>
  <VAR NAME="VERSION">12,1,0,12473</VAR>
</ROW>
<ROW NUM="15">
  <VAR NAME="DATE">Apr 19 2011</VAR>
```

```
            <VAR NAME="NAME">GRF</VAR>
            <VAR NAME="TIME">20:21:07</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="16">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">GUI</VAR>
            <VAR NAME="TIME">20:28:19</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="17">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">INI</VAR>
            <VAR NAME="TIME">20:16:54</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="18">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">LBY</VAR>
            <VAR NAME="TIME">20:35:36</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="19">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">LGN</VAR>
            <VAR NAME="TIME">20:35:52</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="20">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">LOG</VAR>
            <VAR NAME="TIME">20:24:22</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="21">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">PRT</VAR>
            <VAR NAME="TIME">20:20:20</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="22">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">USR</VAR>
            <VAR NAME="TIME">20:34:33</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="23">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">UTL</VAR>
            <VAR NAME="TIME">20:17:28</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
          <ROW NUM="24">
            <VAR NAME="DATE">Apr 19 2011</VAR>
            <VAR NAME="NAME">VMM</VAR>
            <VAR NAME="TIME">20:15:42</VAR>
            <VAR NAME="VERSION">12,1,0,12473</VAR>
          </ROW>
        </ROWSET>
      </MSGVARS>
    </ns6:DSIMSG>
  </ns3:IDSResponse>
  <ns3:Results>
    <Result>0</Result>
```

```
        </ns3:Results>
        <ns3:ServiceInfo>
          <ns2:Operation>doCallIDS</ns2:Operation>
          <ns2:Version>
            <ns2:Number>1</ns2:Number>
            <ns2:Used>true</ns2:Used>
          </ns2:Version>
        </ns3:ServiceInfo>
      </ns5:doCallIDSResponseV1>
    </ns5:doCallIDSResponse>
  </S:Body>
</S:Envelope>
```

CompositionFault

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <S:Fault xmlns:ns4="http://www.w3.org/2003/05/soap-envelope">
      <faultcode>S:Server</faultcode>
      <faultstring>No Response from IDS.</faultstring>
      <detail>
        <CompositionFault:CompositionFault
            xmlns:CompositionFault="oracle/documaker/schema/ws/
composition"
            xmlns="oracle/documaker/schema/ws/composition"
            xmlns:ns2="oracle/documaker/schema/common">
          <faultInfo>java.lang.NullPointerException: No Response fro
m IDS.
            at oracle.documaker.ws.ids.Proxy.doCallIDS(Proxy.java:20
3)
            at java.lang.Thread.run(Thread.java:619)
          </faultInfo>
          <message>No Response from IDS.</message>
        </CompositionFault:CompositionFault>
      </detail>
    </S:Fault>
  </S:Body>
</S:Envelope>
```

USING PUBLISHING SERVICES

Publishing service operations expose Document Factory functionality and Documaker to assemble, publish, and distribute documents.

Document Factory and Documaker Core Run Time

Document Factory is a series of processes in an assembly line that are responsible for assembling, publishing and distributing print streams for one or more documents. Some of the processes in the assembly line use the Documaker core run time for assembly, distribution and publishing. Document Factory must be set up before you can use publishing service operations.

WSDL URLs

Publishing service operations are exposed through these URLs:

SOAP version	URL
1.1	http://IpAddress:Port/DWS/PublishingService?WSDL
1.2	http://IpAddress:Port/DWS/PublishingServiceSoap12?WSDL

Where *IpAddress* and *Port* reflect the IP address and port of the application server hosting DWS.

Service Operations

Here is a list of the service operations provided under publishing services.

Operation	Description
doPublishFromImport	A web service operation that uses the DAO layer of the Document Factory to insert jobs in the Jobs assembly table for the Document Factory and Documaker core run time to assemble, publish and distribute as one or more documents. See *doPublishFromImport* on page 445 for more information.

ERROR HANDLING

Publishing services return a PublishingFault SOAP element with a detailed description of the error encountered. See *PublishingFault Schema* on page 473 and *Example PublishingFault* on page 479 for more information.

CONFIGURING ASSEMBLY LINE

A DWS application instance can only interface with one Document Factory assembly line. To invoke composition service operations, first set up the Document Factory assembly line the DWS application instance should interface with.

This is necessary so composition service operations can retrieve the default Docupresentment message bus configuration from the ALCONFIGCONTEXT Document Factory administration table. This configuration is achieved through web.xml file configuration parameters in WEB-INF directory of the DWS.war file.

Note See *web.xml File* on page 480 for information about JNDI and the assembly line configuration options.

doPublishFromImport

This web service operation uses the Data Access Object (DAO) layer of Document Factory to insert jobs into a Document Factory Jobs assembly table so they can be assembled, published, and distributed as one or more documents via Document Factory's assembly line and Documaker's rules processing engine.

Providing the Extract File for a Job

An extract file is an input file used by the master resource library (MRL) for a Document Factory assembly line to assemble, publish, and distribute as one or more documents. It contains one or more transactions that are broken apart by the Document Factory into the Trns table after being inserted into the Jobs table by this service operation. The format of the extract file can be one of the following:

- Stacked XML file
- Single XML file
- Flat file

The extract file is specific to the MRL for the Document Factory assembly line. There can be only one MRL per Document Factory assembly line.

Note An MRL and extract files are used by Documaker to assemble, publish, and distribute documents and are covered in the Documaker Administration Guide.

Invoking doPublishFromImport

To invoke doPublishFromImport service operation you must submit either an *extract* schema element with a file attachment that contains an extract file with one or more transactions or one or more *transaction* schema elements, each with its own extract data in the form of a file attachment.

Here is an example of a request payload that submits an extract schema element. The request submits the extract data as a URI (Uniform Resource Identifier) element to a file local to the Document Factory. Also, by specifying JobId value in the ResponseType element, the request message indicates the response message should return only the JobId for the job that was imported.

You could also specify the Attachments value for the ResponseType element to indicate the response message should return the output print streams as binary base64 encoded data.

You would typically use the extract element instead of transaction element when the extract data contains more than one transaction that needs to be parsed and separated by Document Factory and you do not need to define any Trns table column values at the request message level.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cmn="oracle/documaker/schema/common"
xmlns:req="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/request"
```

```
xmlns:pubcmn="oracle/documaker/schema/ws/publishing/common"
xmlns:tns="oracle/documaker/schema/ws/publishing"
xmlns:v1="oracle/documaker/schema/ws/publishing/doPublishFromImport/
v1">
  <soap:Body>
    <tns:doPublishFromImportRequest>
      <tns:doPublishFromImportRequestV1>
        <pubcmn:timeoutMillis>110000</pubcmn:timeoutMillis>
        <v1:JobRequest>
          <req:Payload>
            <req:Extract>
              <cmn:Content>
                <cmn:URILocation>
                  <cmn:Location>Server</cmn:Location>
                  <cmn:URI>file:///oracle/oracle_insurance_1/
documaker/mstrres/dmres/input/extrfile.xml</cmn:URI>
                </cmn:URILocation>
              </cmn:Content>
            </req:Extract>
          </req:Payload>
        </v1:JobRequest>
        <v1:ResponseProperties>
          <cmn:ResponseType>JobId</cmn:ResponseType>
        </v1:ResponseProperties>
      </tns:doPublishFromImportRequestV1>
    </tns:doPublishFromImportRequest>
  </soap:Body>
</soap:Envelope>
```

Note See *Input Formats* on page 142 for more information about the accepted input formats of an extract file.

Here is an example of a request payload that submits a transaction schema element. The request submits the transaction's extract data as binary base64 content in the Data element (most of the Binary base64 content has been omitted for brevity).

Also, by specifying Attachments value in the ResponseType element, the request message indicates the response message should return any output print streams as base64 binary content.

You should only use the transaction element when the Data element only contains the extract data for a single transaction. Another advantage of using the transaction element is that it lets you define values that can override the values in the different Trns table columns for a transaction.

Note See *Transaction* on page 458 for more information about the Transaction element.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cmn="oracle/documaker/schema/common"
xmlns:req="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/request"
xmlns:pubcmn="oracle/documaker/schema/ws/publishing/common"
xmlns:tns="oracle/documaker/schema/ws/publishing"
xmlns:v1="oracle/documaker/schema/ws/publishing/doPublishFromImport/
v1">
  <soap:Body>
    <tns:doPublishFromImportRequest>
      <tns:doPublishFromImportRequestV1>
```

```
        <pubcmn:timeoutMillis>90000</pubcmn:timeoutMillis>
        <v1:JobRequest>
          <req:Payload>
            <req:Transaction>
              <req:Data>
                <cmn:Content>
                  <cmn:Binary>PD94bWwgdmVyc...</cmn:Binary>
                </cmn:Content>
              </req:Data>
            </req:Transaction>
          </req:Payload>
        </v1:JobRequest>
        <v1:ResponseProperties>
          <cmn:ResponseType>Attachments</cmn:ResponseType>
    </v1:ResponseProperties>
      </tns:doPublishFromImportRequestV1>
    </tns:doPublishFromImportRequest>
  </soap:Body>
</soap:Envelope>
```

Note Input file attachments can be sent as HTTP URIs or base64 binary file attachments. See *Attachment* on page 434 for more information.

Here is an example, based on a one transaction per job situation, which is typical. First you would check the TRNStatus to make sure it is set to 290. This means it is in WIP ready for Documaker Interactive to access.

Then take the TRN_ID and KeyID values and the Documaker Interactive location (in the demo app, this is referenced in the IP.XML file in the software\temp\config directory) and launch this URL:

```
https://10.140.215.247:7002/idm/faces/
load?taskflow=value&uniqueId=value&docId=value
```

Where the IP is correct for Documaker Interactive and where the task flow values are:

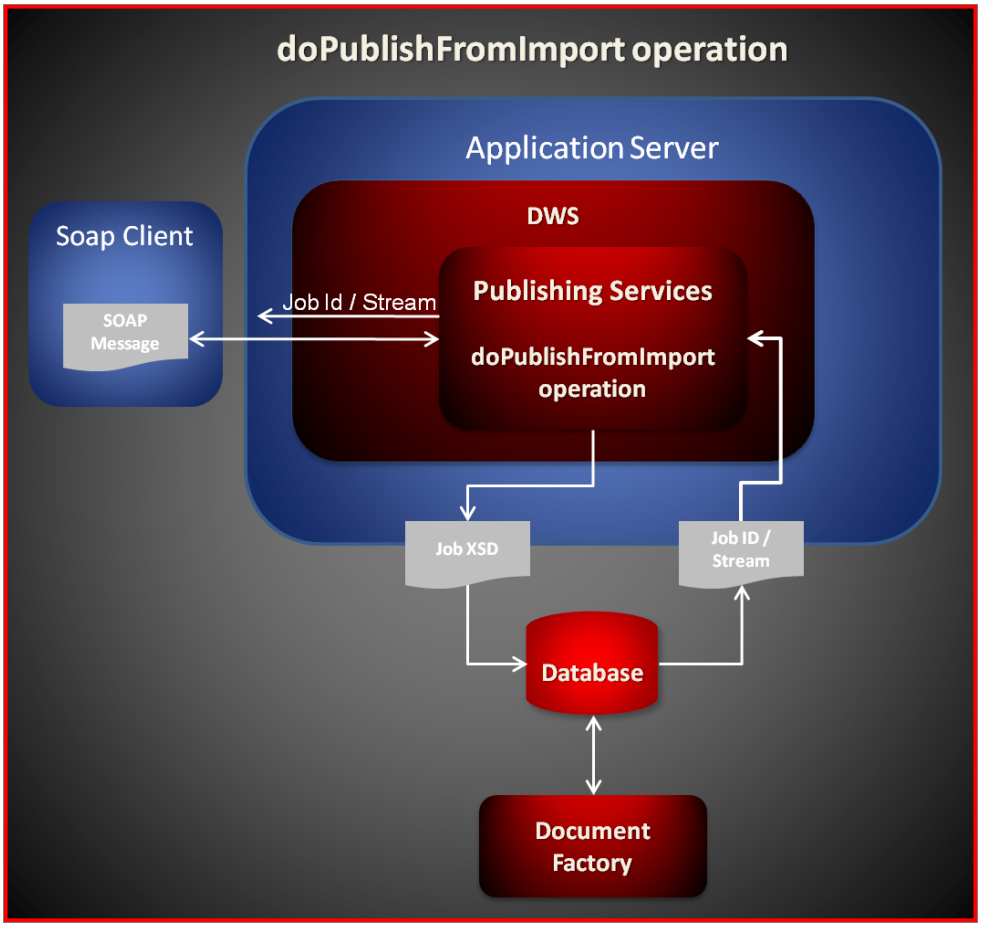
Field	Value
inbox	The Inbox tab
edit	The Forms tab for the particular transaction
compose	The Document tab in the WIP Edit plug-in for the particular transaction
uniqueId	The TRN_ID
docID	The document ID displayed on the tab and indicated by the KeyID

Remember that Documaker Interactive has a tie to the owner or owner group of the transaction so if you do not apply one in the XML data feed, the transaction will appear on the Unassigned tab or be sent to the default user you set up in the AFG2WIP control group in the FSISYS.INI file.

Also remember the system assumes authentication has already taken place. In the demo this is true because the user logs into the demo app and because the demo app and Documaker Interactive are in the same security context, so authentication is successful Documaker Interactive launches. If the authentication process is unsuccessful, Documaker Interactive requires you to log in.

The Response Payload

The response payload varies, and is determined by different input options in the request payload. The doPublishFromImport service operation can return the job ID of the new record created in the Jobs table or the print streams generated by Document Factory.



Here is an example of a response message that returns a transaction with the print streams:

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns5:doPublishFromImportResponse
    xmlns:ns6="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/response"
    xmlns:ns5="oracle/documaker/schema/ws/publishing"
    xmlns:ns4="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/request"
    xmlns:ns3="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1"
    xmlns:ns2="oracle/documaker/schema/common"
    xmlns="oracle/documaker/schema/ws/publishing/common">
      <ns5:doPublishFromImportResponseV1>
        <Result>0</Result>
        <ServiceTimeMillis>14454</ServiceTimeMillis>
        <ns3:JobResponse>
          <ns6:JobBchErr>0</ns6:JobBchErr>
```

```
          <ns6:JobBchProc>1</ns6:JobBchProc>
          <ns6:JobBchSch>2</ns6:JobBchSch>
          <ns6:JobBchStartTime>2011-04-12T15:45:43.260Z</
ns6:JobBchStartTime>
          <ns6:JobBchTotal>3</ns6:JobBchTotal>
          <ns6:JobHistorical>0</ns6:JobHistorical>
          <ns6:JobHistory>1</ns6:JobHistory>
          <ns6:JobPayloadType>0</ns6:JobPayloadType>
          <ns6:JobPriority>10</ns6:JobPriority>
          <ns6:JobRcpErr>0</ns6:JobRcpErr>
          <ns6:JobRcpProc>1</ns6:JobRcpProc>
          <ns6:JobRcpSch>2</ns6:JobRcpSch>
          <ns6:JobRcpStartTime>2011-04-12T15:45:43.260Z</
ns6:JobRcpStartTime>
          <ns6:JobRcpTotal>3</ns6:JobRcpTotal>
          <ns6:JobStartTime>2011-04-12T15:45:39.728Z</
ns6:JobStartTime>
          <ns6:JobStatus>416</ns6:JobStatus>
          <ns6:JobTrnErr>0</ns6:JobTrnErr>
          <ns6:JobTrnProc>0</ns6:JobTrnProc>
          <ns6:JobTrnSch>1</ns6:JobTrnSch>
          <ns6:JobTrnStartTime>2011-04-12T15:45:40.119Z</
ns6:JobTrnStartTime>
          <ns6:JobTrnTotal>1</ns6:JobTrnTotal>
          <ns6:JobTrnWip>0</ns6:JobTrnWip>
          <ns6:JobUnique_Id>1b6d8297-2f5b-48f5-9c11-3ef8a0f5636c</
ns6:JobUnique_Id>
          <ns6:Job_Id>6</ns6:Job_Id>
          <ns6:Payload>
            <ns6:Transaction>
              <ns6:Action>100011</ns6:Action>
              <ns6:ApprovalState>10</ns6:ApprovalState>
              <ns6:CreateTime>2011-04-12T15:45:40.000Z</
ns6:CreateTime>
              <ns6:CurrGroup>3</ns6:CurrGroup>
              <ns6:CurrUser>8</ns6:CurrUser>
              <ns6:Customized>0</ns6:Customized>
              <ns6:Data>
                <ns2:Name>6_1</ns2:Name>
                <ns2:ContentType>message/rfc822</ns2:ContentType>
                <ns2:FileType>htm</ns2:FileType>
                <ns2:Content>
                  <ns2:Binary>TUlNRS12ZX..</ns2:Content>
              </ns6:Data>
              <ns6:Descr>Welcome Packet</ns6:Descr>
              <ns6:FormsetId>1b6d8297-2f5b-48f5-9c11-3ef8a0f5636c</
ns6:FormsetId>
              <ns6:Job_Id>6</ns6:Job_Id>
              <ns6:Key1>Central</ns6:Key1>
              <ns6:Key2>Account_Status</ns6:Key2>
              <ns6:KeyId>0000004</ns6:KeyId>
              <ns6:ModifyTime>2011-04-12T15:45:42.000Z</
ns6:ModifyTime>
              <ns6:OrigUser>8</ns6:OrigUser>
              <ns6:ProcessName>Batcher</ns6:ProcessName>
              <ns6:RecType>00</ns6:RecType>
              <ns6:SecLevel>0</ns6:SecLevel>
              <ns6:StatusCode>P</ns6:StatusCode>
              <ns6:TranCode>null</ns6:TranCode>
              <ns6:TrnBchErr>0</ns6:TrnBchErr>
              <ns6:TrnBchProc>1</ns6:TrnBchProc>
              <ns6:TrnBchSch>2</ns6:TrnBchSch>
              <ns6:TrnBchTotal>3</ns6:TrnBchTotal>
              <ns6:TrnDoLog>0</ns6:TrnDoLog>
```

```
                <ns6:TrnHistorical>0</ns6:TrnHistorical>
                <ns6:TrnHistory>1</ns6:TrnHistory>
                <ns6:TrnRcpErr>0</ns6:TrnRcpErr>
                <ns6:TrnRcpProc>1</ns6:TrnRcpProc>
                <ns6:TrnRcpSch>2</ns6:TrnRcpSch>
                <ns6:TrnRcpTotal>3</ns6:TrnRcpTotal>
                <ns6:TrnStartTime>2011-04-12T15:45:40.119Z</
ns6:TrnStartTime>
                <ns6:TrnStatus>416</ns6:TrnStatus>
                <ns6:Trn_Id>6</ns6:Trn_Id>
                <ns6:Unique_Id>1b6d8297-2f5b-48f5-9c11-3ef8a0f5636c</
ns6:Unique_Id>
              </ns6:Transaction>
            </ns6:Payload>
          </ns3:JobResponse>
          <ns3:ServiceInfo>
            <ns2:Operation>doPublishFromImport</ns2:Operation>
            <ns2:Version>
              <ns2:Number>1</ns2:Number>
              <ns2:Used>true</ns2:Used>
            </ns2:Version>
          </ns3:ServiceInfo>
        </ns5:doPublishFromImportResponseV1>
      </ns5:doPublishFromImportResponse>
    </S:Body>
</S:Envelope>
```

Message Schema

The following schema elements comprise the request and response payload for the doPublishFromImport web service operation. The Type/Count column in each of these schema tables describes the schema type and occurrence. The schema type can refer to other custom schema types.

If the count is defined as	It means the element is
one (1)	Required.
(0...1)	Optional.
(0...many) or (1...many)	Optional, but more than one element of this type can exist. or Required, but more than one element of this type can exist.

Certain schema elements are defined as (choice) and then contain a list of elements. This means one, but no more than one, of the elements in the list can be used. This is standard schema nomenclature.

Discussions of these schema elements follow:

- *Transaction* on page 467
- *ServiceInfo* on page 472
- *Version* on page 472

doPublishFromImportRequest

DWS provides web service versioning at the message level. The doPublishFromImportRequest element contains a schema choice element that provides the ability to select different versions of a request message.

Element	Description	Type/Count
(choice)	Contains one of these elements: doPublishFromImportRequestV1	choice (1)

doPublishFromImportRequestV1

The doPublishFromImportRequestV1 element is the first message version of doPublishFromImportRequest element. It contains these elements:

Element	Description	Type/Count
timeoutMillis	Specifies how long the service operation should wait for a reply message from Document Factory. The default is 30,000 milliseconds.	IDSRequest (1)
Properties	Provides the message bus configuration options that can be used to communicate with Document Factory. This element overrides the default message bus configuration.	Properties (0...1)
JobRequest	Contains the request payload for a Document Factory request.	JobRequest (1)
ResponseProperties	A response properties element that defines how attachments should be returned.	ResponseProperties (0...1)

Properties

Use this element to override the default message bus properties configured in ALCONFIGCONTEXT administration table. This element lets you configure each request payload to talk to a different Document Factory instance.

Element	Description	Type/Count
(choice)	Contains one of these elements: MQ MSMQ JMS	choice (1)

MQ

A set of WebSphere MQ message bus configuration options for communicating with Document Factory.

Element	Description	Type/Count
queue.factory.class	The fully-qualified class name of the MQ queue factory class to use. The value of this element is final: com.docucorp.messaging.mqseries.DSIMQMessageQueueFactory	string (1)
marshaller.class	The fully-qualified class name of the marshaller class to use.	MarshallerClass (1)
mq.queue.manager	The name of the MQ queue manager. The value is case-sensitive. The default is queue_manager.	string (1)
mq.tcpip.host	The host name or IP address of the server where the MQ queue manager resides. Omit this element to use a bindings mode connection. Include this element to use client mode.	string (1)
mq.tcpip.port	The port number the MQ queue manager is listening on. Omit this element to use a bindings mode connection. Include this element to use client mode. The default is 1414.	string (0...1)
mq.inputqueue.name	The name of the input queue. The input queue is the queue that is used to read reply messages from Document Factory, meaning it is the output queue on the Document Factory side. The value is case-sensitive. The default is RESULTQ.	string (1)
mq.outputqueue.name	The name of the output queue. The output queue is the queue that is used to send request messages to Document Factory, meaning it is the input queue on the Document Factory side. The value is case-sensitive. The default is REQUESTQ.	string (1)
mq.queue.channel	The name of the MQ Server Connection Channel to use. Omit this element to use a bindings mode connection. Include this element to use client mode. The value is case-sensitive. The default is SYSTEM.DEF.SVRCONN.	string (0...1)
mq.outputqueue.expiry	How long should a message placed in MQ stay around. The default value is 1800 seconds. Use a value of -1 to indicate the message never expires.	string (0...1)
mqseries.exception.logging	This option enables exception logging at the WebSphere MQ level. Acceptable values are Yes or No. The default is Yes.	string (0...1)
mqseries.tracing	Sets the WebSphere MQ tracing level. MQSeriesTracing (1...4), 1 being the lowest level of tracing.	string (0...1)
mqseries.log	Sets the location and name of the WebSphere MQ log file to use when mqseries.exception.logging and/or mqseries.tracing options are enabled.	string (0...1)

* = Only used when com.docucorp.messaging.mqseries.DSIMQSSLSocketFactory is specified as the value of the mq.ssl.SocketFactory.class option. SSL options should only be used if the queue manager has been configured to support SSL.

Element	Description	Type/Count

mq.ccdt.url	This value should contain the URL of a client connection definition table (CCDT) that should be used to derive all the connection information for this factory. This property and the mq.queue.channel property are mutually exclusive; do not define both or you gets an MQSeries 2423 MQRC error. Here are a few examples of URL values:
	`file:///c:/mq/ccdt/AMQCLCHL.TAB`
	`file:/c:/mq/ccdt/AMQCLCHL.TAB`
	`ftp://userName:password@myServer/ccdt_files/AMQCLCHL.TAB`
		string (0...1)

mq.ssl.cipherspec	The encryption and hashing algorithm used for SSL communications.	MQSSLCipherspec (0...1)
mq.ssl.peername	The distinguished name (DN) pattern of the SSL certificate used by the queue manager. This is used to validate the queue manager.	string (0...1)
mq.ssl.SocketFactory.class	The name of a custom SSL socket factory class that should be used to override the default SSL socket factory used by WebSphere MQ - javax.net.ssl.SSLSocketFactory. This value should contain the package and class name of an SSL socket factory class that extends the javax.net.ssl.SSLSocketFactory class. There is no default. If this property is not specified and SSL support is enabled, WebSphere MQ uses the javax.net.ssl.SSLSocketFactory class and looks for the java key and trust stores in this way: • Look for them in the System properties javax.net.ssl.keyStore and javax.net.ssl.trustStore and look for their passwords in the System properties javax.net.ssl.keyStorePassword and javax.net.ssl.trustStorePassword. • If the system properties are not defined, look in the default keyStore/trustStore named *cacerts* located in the JAVA_HOME\jre\lib\security directory and use the default password (*changeit*) for them. Implementations that need to load their own keyStore and trust store and do not want to use the system properties can either define their own SSLSocketFactory class or use the com.docucorp.messaging.mqseries.DSIMQSSLSocketFactory class in DocucorpMsg.jar package by providing the appropriate value for this property. See also the mq.ssl.protocol, mq.ssl.keyStore, mq.ssl.keyStore.type, mq.ssl.keyStore.manager.type, mq.ssl.keyStore.pwd, mq.ssl.trustStore, mq.ssl.trustStore.type, mq.ssl.trustStore.manager.type, and mq.ssl.trustStore.pwd properties.	string (0...1)
mq.ssl.protocol *	The SSL protocol to use with a custom SSL socket factory class. The default is SSLv3.	string (0...1)
mq.ssl.keyStore *	The path and file name of the java key store where the private keys and public certificates are stored.	string (0...1)
mq.ssl.keyStore.type *	mq.ssl.keyStore.type *	string (0...1)

* = Only used when com.docucorp.messaging.mqseries.DSIMQSSLSocketFactory is specified as the value of the mq.ssl.SocketFactory.class option. SSL options should only be used if the queue manager has been configured to support SSL.

Element	Description	Type/Count
mq.ssl.keyStore.manager.type *	The key store manager type. The default is SunX509.	string (0...1)
mq.ssl.keyStore.pwd *	The password for the SSL key store.	string (0...1)
mq.ssl.trustStore *	The path and file name of the java trust store where the trusted public certificates are stored.	string (0...1)
mq.ssl.trustStore.type *	The type of trust store used. The default is JKS (Java Key Store).	string (0...1)
mq.ssl.trustStore.manager.type *	The trust manager type. The default is SunX509.	string (0...1)
mq.ssl.trustStore.pwd	The password for the SSL trust store.	string (0...1)
mq.ssl.debug	A value of Yes or No enables debug for the SSL session. This is a system-wide (global) property. The default is No.	string (0...1)
mq.Property	Use this option to supply additional MQ provider-specific properties. This option is reserved for future use.	Property (0...1)

* = Only used when com.docucorp.messaging.mqseries.DSIMQSSLSocketFactory is specified as the value of the mq.ssl.SocketFactory.class option. SSL options should only be used if the queue manager has been configured to support SSL.

MQSeriesTracing

Element	Description	Type/Count
(enum)	The tracing level for WebSphere MQ code. Acceptable values for this option are: 1 (lowest) 2 3 4 (highest)	int (1)

MQSSLCipherspec

Element	Description	Type/Count
(enum)	The SSL encryption and hashing algorithm for WebSphere MQ. Acceptable values for this option are: DES_SHA_EXPORT DES_SHA_EXPORT1024 NULL_MD5 NULL_SHA RC2_MD5_EXPORT RC4_56_SHA_EXPORT1024 RC4_MD5_US RC4_MD5_EXPORT RC4_SHA_US TRIPLE_DES_SHA_US	string (1)

MSMQ

A set of MSMQ message bus configuration options for communicating with Document Factory.

Element	Description	Type/Count
queue.factory.class	The fully-qualified class name of the MSMQ queue factory class to use. The value of this element is final: com.docucorp.messaging.msmq.DSIMSMQMessageQueueFactory	string (1)
marshaller.class	The fully-qualified class name of the marshaller class to use.	MarshallerClass (1)
msmq.server.name	The IP address or server name for the MSMQ server. This property is not used when direct format names are used for the input and output queues.	string (0...1)
msmq.inputqueue.name	The name of the input queue. This can be a queue path name or a direct format name. Queue path names are used with the msmq.server.name property and should not include the server name. Direct format names do not use the msmq.server.name property. The default is DIRECT=OS:localhost\PRIVATE$\RESULTQ	string (1)
msmq.outputqueue.name	The name of the output queue. This can be a queue path name or a direct format name. Queue path names are used with the msmq.server.name property and should not include the server name. Direct format names do not use the msmq.server.name property. The default is DIRECT=OS:localhost\PRIVATE$\REQUESTQ	string (1)
msmq.timeout	The timeout interval in milliseconds. This determines how long to wait for a message to reach a queue during a send operation. The default is 30000 ms (30 seconds).	string (0...1)
msmq.expiry	How long should a message remain in the queue before it is deemed expired. This value is used during a send operation. The default is one (1). 800000 ms are equal to 30 minutes.	string (0...1)

Element	Description	Type/Count
msmq.Property	Use this option to supply additional MSMQ provider specific properties. This option is reserved for future use.	string (0...many)

JMS

A set of JMS message bus configuration options for communicating with Document Factory.

Element	Description	Type/Count
queue.factory.class	The fully-qualified class name of the JMS JNDI queue factory class to use. The value of this element is final: com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory	string (1)
marshaller.class	The fully-qualified class name of the marshaller class to use.	MarshallerClass (1)
jms.initial.context.factory	The fully-qualified class name of the JMS provider initial context factory. The default is weblogic.jndi.WLInitialContextFactory	string (1)
jms.provider.URL	The JMS provider URL. The default is t3://localhost:7001	string (1)
jms.qcf.name	The JNDI name of the queue connection factory. The default is qcf.	string (1)
jms.inputqueue.connectstring	The JNDI name of the input queue. The default is resultq.	string (1)
jms.outputqueue.connectstring	The JNDI name of the output queue. The default is requestq.	string (1)
jms.security.principal	The account to use when authentication is required.	string (0...1)
jms.security.credentials	The account password to use when authentication is required.	string (0...1)
jms.env.Property	An additional JNDI context property.	Property (0...many)

JobRequest

Element	Description	Type/Count
JobHistory	A value of zero (0) or one (1) that indicates if the job table data should be copied to the history table upon deletion. This can be set to zero (0) by an application before deletion. The default is one (1) which copies the job table data to the history table.	int (0...1)
JobName	The job name.	string (0...1)

Element	Description	Type/Count
JobPriority	The pick list value for the assigned job priority which affects the order of processing. Jobs with lower values are processed first. Here are some examples: • 0=immediate/highest priority • 10=normal/regular priority • 20=lowest priority	Priority (0...1)
JobName	The date and time for job retention. Jobs that have a value that is less than the current system time qualify to be purged from the system.	dateTime (0...1)
JobStatus	The overall processing status of the job as it is being processed through the system. Only override it if you want to place this job on hold.	int (0...1)
JobUnique_Id	A unique identifier string that can be used by an application to identify the job.	string (0...1)
Payload	The element that contains the extract file data.	Payload (0...1)

Payload

The content of the new job.

Element	Description	Type/Count
(choice)	One of these: Transaction Extract (Data type)	choice (1)

Transaction

A transaction for the job. Use this element to provide values that override a transaction's column values in the Trns table record for the transaction. Use this element with an extract file that only contains a single transaction in the Data child element.

Element	Description	Type/Count
Action	The action value. Use this column to override the default value if you have custom actions that should be performed on the transaction by the system. The default is batch created.	int (0...1)
AgencyId	The agency ID.	string (0...1)
ApprovalState	The data related to the approval process workflow: • Draft = 10 • Pending Approval = 20 • Approved = 30 • Rejected = 40 • Pending Distribution = 50 • Distributed = 60	string (0...1)
CurrGroup	The group of the current user of the transaction.	int (0...1)

Element	Description	Type/Count
CurrRole	The role of the current user of the transaction.	string (0...1)
CurrSuper	The supervisor of the current user of the transaction.	int (0...1)
CurrUser	The current user of the transaction.	int (0...1)
Customized	A value of zero (0) or one (1), where one (1) means Yes. This is used to indicate if the transaction is customized. Reserved for future use.	int (0...1)
Data	The extract data for a transaction. Should only contain a single transaction's extract data. If you need to submit extract data for more than one transaction in an extract file you have these options: • Break the extract data apart for each transaction and then use a separate Transaction element and Data child element for each one. • Use the Extract child element under the JobRequest and Payload elements and provide the multiple transaction extract data.	double (0...1)
Descr	The transaction description.	string (0...1)
DocSubType	The document sub-type of the transaction.	string (0...1)
DocType	The document type of the transaction.	string (0...1)
FromGroup	The group that assigned the transaction.	int (0...1)
FromTime	The date and time the transaction was assigned from a user or group.	dateTime (0...1)
FromUser	The user who assigned the transaction.	int (0...1)
InUse	The in-use flag. Only set this value if you want to lock the transaction.	string (0...1)
Jurisdictn	The transaction jurisdiction code.	string (0...1)
Key1	The key 1 value.	string (0...1)
Key2	The key 2 value.	string (0...1)
Key3	The key 3 value.	string (0...1)
KeyId	The key ID value.	string (0...1)
LocId	The location ID.	string (0...1)
OrigUser	The original user or creator/author of the transaction.	int (0...1)
ProcessName	The process name that created this transaction. This is normally set by the application inserting the job but can be overridden.	string (0...1)
QueueId	The queue identifier.	string (0...1)
Reason_Id	The reason code for routing rejection or processing.	string (0...1)

Element	Description	Type/Count
RecType	The record type.	string (0...1)
Retention	A date and time stamp that indicates how long to retain the transaction.	dateTime (0...1)
RouteDesc	The code that indicates the reason why a document was routed or rejected.	string (0...1)
SecLevel	The security level for the transaction.	int (0...1)
StatusCode	The status code of the transaction. The value is set by the system to be W, E, B, or P as part of Document Factory processing but may be overwritten if provided here.	string (0...1)
SubLocId	The sub-location ID.	string (0...1)
ToGroup	The group the transaction was assigned to.	int (0...1)
ToTime	The date and time the transaction was assigned to a user or group.	dateTime (0...1)
ToUser	The user the transaction was assigned to.	int (0...1)
TranCode	The transaction code for the transaction.	string (0...1)
TrnAppDate001	An application-defined date and time column.	dateTime (0...1)
TrnAppDate002	An application-defined date and time column.	dateTime (0...1)
TrnAppDate003	An application-defined date and time column.	dateTime (0...1)
TrnAppDate004	An application-defined date and time column.	dateTime (0...1)
TrnAppDate005	An application-defined date and time column.	dateTime (0...1)
TrnAppDec001	An application-defined decimal column.	double (0...1)
TrnAppDec002	An application-defined decimal column.	double (0...1)
TrnAppDec003	An application-defined decimal column.	double (0...1)
TrnAppDec004	An application-defined decimal column.	double (0...1)
TrnAppDec005	An application-defined decimal column.	double (0...1)
TrnAppInt001	An application-defined signed integer column.	int (0...1)
TrnAppInt002	An application-defined signed integer column.	int (0...1)
TrnAppInt003	An application-defined signed integer column.	int (0...1)
TrnAppInt004	An application-defined signed integer column.	int (0...1)
TrnAppInt005	An application-defined signed integer column.	int (0...1)
TrnAppStr001	An application-defined string column.	string (0...1)
TrnAppStr002	An application-defined string column.	string (0...1)

Element	Description	Type/Count
TrnAppStr003	An application-defined string column.	string (0...1)
TrnAppStr004	An application-defined string column.	string (0...1)
TrnAppStr005	An application-defined string column.	string (0...1)
TrnAppStr006	An application-defined string column.	string (0...1)
TrnAppStr007	An application-defined string column.	string (0...1)
TrnAppStr008	An application-defined string column.	string (0...1)
TrnAppStr009	An application-defined string column.	string (0...1)
TrnAppStr010	An application-defined string column.	string (0...1)
TrnAppStr011	An application-defined string column.	string (0...1)
TrnAppStr012	An application-defined string column.	string (0...1)
TrnAppStr013	An application-defined string column.	string (0...1)
TrnAppStr014	An application-defined string column.	string (0...1)
TrnAppStr015	An application-defined string column.	string (0...1)
TrnCusDate001	A custom date and time column.	dateTime (0...1)
TrnCusDate002	A custom date and time column.	dateTime (0...1)
TrnCusDate003	A custom date and time column.	dateTime (0...1)
TrnCusDate004	A custom date and time column.	dateTime (0...1)
TrnCusDate005	A custom date and time column.	dateTime (0...1)
TrnCusDate006	A custom date and time column.	dateTime (0...1)
TrnCusDate007	A custom date and time column.	dateTime (0...1)
TrnCusDate008	A custom date and time column.	dateTime (0...1)
TrnCusDate009	A custom date and time column.	dateTime (0...1)
TrnCusDate010	A custom date and time column.	dateTime (0...1)
TrnCusDate011	A custom date and time column.	dateTime (0...1)
TrnCusDate012	A custom date and time column.	dateTime (0...1)
TrnCusDate013	A custom date and time column.	dateTime (0...1)
TrnCusDate014	A custom date and time column.	dateTime (0...1)
TrnCusDate015	A custom date and time column.	dateTime (0...1)
TrnCusDec001	A custom decimal column.	double (0...1)

Element	Description	Type/Count
TrnCusDec002	A custom decimal column.	double (0...1)
TrnCusDec003	A custom decimal column.	double (0...1)
TrnCusDec004	A custom decimal column.	double (0...1)
TrnCusDec005	A custom decimal column.	double (0...1)
TrnCusInt001	A custom signed integer column.	int (0...1)
TrnCusInt002	A custom signed integer column.	int (0...1)
TrnCusInt003	A custom signed integer column.	int (0...1)
TrnCusInt004	A custom signed integer column.	int (0...1)
TrnCusInt005	A custom signed integer column.	int (0...1)
TrnCusStr001	A custom string column.	string (0...1)
TrnCusStr002	A custom string column.	string (0...1)
TrnCusStr003	A custom string column.	string (0...1)
TrnCusStr004	A custom string column.	string (0...1)
TrnCusStr005	A custom string column.	string (0...1)
TrnCusStr006	A custom string column.	string (0...1)
TrnCusStr007	A custom string column.	string (0...1)
TrnCusStr008	A custom string column.	string (0...1)
TrnCusStr009	A custom string column.	string (0...1)
TrnCusStr010	A custom string column.	string (0...1)
TrnCusStr011	A custom string column.	string (0...1)
TrnCusStr012	A custom string column.	string (0...1)
TrnCusStr013	A custom string column.	string (0...1)
TrnCusStr014	A custom string column.	string (0...1)
TrnCusStr015	A custom string column.	string (0...1)
TrnCusStr016	A custom string column.	string (0...1)
TrnCusStr017	A custom string column.	string (0...1)
TrnCusStr018	A custom string column.	string (0...1)
TrnCusStr019	A custom string column.	string (0...1)
TrnCusStr020	A custom string column.	string (0...1)

Element	Description	Type/Count
TrnDoLog	A value of zero (0) or one (1) that is used to control when the TrnsLog table is updated. Set this value to one (1) to enable logging for this and future actions on this transaction into the TrnsLog table unless a subsequent process disables logging. The default is zero (0).	int (0...1)
TrnHistory	A value of zero (0) or one (1) that determines whether the Trns table data is copied to the history table upon deletion. This element can be set to zero (0) by an application before deletion. The default is one (1) which copies the Trns table data to the history table.	int (0...1)
TrnName	The name of the transaction.	string (0...1)
TrnPrtLogName	The logical printer name when the transaction is sent to a predetermined logical printer.	string (0...1)
TrnStatus	A numerical value that indicates the overall status of the transaction as it is being processed through the system.	int (0...1)
Unique_Id	A unique identifier string that can be used for application look up of the transaction.	string (0...1)

Data

The extract data for a transaction.

Element	Description	Type/Count
Name	The name of the data. This can be a file name.	string (0...1)
ContentType	The mime (Multipurpose Internet Mail Extensions) content type for the extract data.	string (0...1)
FileType	The file type for the extract data.	string (0...1)
Content	The content of the data.	Content (1)

Content

Represents the content of a file attachment.

Element	Description	Type/Count
(choice)	Contains one of these elements: URILocation Binary (base64Binary)	choice (1)

URILocation

Provides the URI and location information for a URI.

Element	Description	Type/Count
Location	Defines the location of the URI relative to DWS or to Document Factory	URIType (1)
URI	The URI to the extract data file.	anyURI (1)

URIType

Provides the location for a URI relative to DWS or to Document Factory.

Element	Description	Type/Count
(enum)	Choose one of these values: Client Server	enum (1)

Note Where *Client* means the URI is relative to server where DWS is installed and Server means the URI is relative to the server where Document Factory is installed.

ResponseProperties

This element indicates how file attachments should be returned in the response message.

Element	Description	Type/Count
ResponseType	Defines the type of response that should be returned.	ResponseType (0...1)
URILocation	Defines a location to write any file attachments returned in the response. When this option is not defined the file attachments are returned as inline base64 encoded content.	URILocation (0...1)

ResponseType

Element	Description	Type/Count
(enum)	Choose one of these values: • JobId • Attachments	enum (1)

Note Where *JobId* indicates the response message should only return the job ID and Attachments indicates the response message should return the file attachments.

doPublishFromImportResponse

DWS provides web service versioning at the message level. The doPublishFromImportResponse element contains a schema choice element that provides the ability to select different versions of a response message, however, a response message will always contain the appropriate message version to match the version in the request message invocation.

Element	Description	Type/Count
(choice)	Contains one of these elements: doPublishFromImportResponseV1	choice (1)

doPublishFromImportResponseV1

The doPublishFromImportResponseV1 element is the first message version of doPublishFromImportResponse element. It contains these elements:

Element	Description	Type/Count
Result	An value that defines the overall result of the service operation: • Zero (0) means success • One (1) means failure	Result (1)
ServiceTimeMillis	How long the service operation took to execute. The elapsed time is returned in milliseconds.	int (1)
JobResponse	Contains the response payload.	JobResponse (1)
ServiceInfo	Returns information about the invoked service operation.	ServiceInfo (1)

JobResponse

Element	Description	Type/Count
JobBchEndTime	The date and time when the job batches finished processing.	dateTime (0...1)
JobBchErr	The number of batches for the job that encountered errors while processing.	int (0...1)
JobBchProc	The number of batches for the job that were successfully processed to completion.	int (0...1)
JobBchSch	The number of batches for the job that were scheduled due to configuration settings in the Bchings table.	int (0...1)
JobBchStartTime	The date and time when the job batches started processing.	dateTime (0...1)
JobBchTotal	The total number of batches for a job.	int (0...1)
JobEndTime	The date and time when the job completed processing through the system.	dateTime (0...1)
JobErr_Id	The last error ID found while processing the job through the system.	int (0...1)
JobHistorical	A value of zero (0) or one (1), where one (1) means Yes. This indicate whether the job is in the job history table. The default is zero (0).	int (0...1)
JobHistory	A numerical value of zero (0) or one (1), where one (1) means Yes. This indicates whether the job table data should be copied to the history table upon deletion. Can be set to zero (0) by an application before deletion. The default is one (1).	int (0...1)
JobName	The job name.	string (0...1)

Element	Description	Type/Count
JobPayloadType	The payload type pick list: 0=data in XML data type 1=data in BLOB 2=URI reference	int (0...1)
JobPriority	The pick list value for the assigned job priority which affects the order of processing. Jobs with lower values are processed first. Here are some examples: 0=immediate/highest priority 10=normal/regular priority 20=lowest priority	int (0...1)
JobPrtLogName	The logical printer name to send the job to.	string (0...1)
JobRcpEndTime	The date and time when the job recipients finished processing.	dateTime (0...1)
JobRcpErr	The number of recipients for the job that encountered errors during processing.	int (0...1)
JobRcpProc	The number of recipients for the job that were successfully processed to completion.	int (0...1)
JobRcpSch	The number of recipients for the job that were scheduled due to one or more scheduled batches configured in the Bchings table.	int (0...1)
JobRcpStartTime	The date and time when the job recipients began processing.	dateTime (0...1)
JobRcpTotal	The total number of recipients for the job.	int (0...1)
JobName	The date and time for job retention. Jobs that have a value that is less than the current system time qualify to be purged from the system.	dateTime (0...1)
JobStartTime	The date and time when the job was created.	dateTime (0...1)
JobStatus	The overall processing status of the job as it is being processed through the system. Only override it if you want to place this job on hold.	int (0...1)
JobTrnEndTime	The date and time when the job transactions finished processing.	dateTime (0...1)
JobTrnErr	The number of transactions for the job that encountered errors while processing through the system.	int (0...1)
JobTrnProc	The number of transactions for the job that were successfully processed to completion through the system.	int (0...1)
JobTrnSch	The number of transactions for the job that were scheduled due to one or more scheduled batches configured in the Bchings table.	int (0...1)
JobTrnStartTime	The date and time when the job transactions began processing.	dateTime (0...1)
JobTrnTotal	The total number of transactions for the job.	int (0...1)
JobTrnWip	The number of transactions for the job that were set to manual work in progress status and are awaiting end user input.	int (0...1)

Element	Description	Type/Count
JobUnique_Id	A unique identifier string that can be used by an application to identify the job.	string (0...1)
Job_Id	The job unique identifier for the new Jobs table record.	int (0...1)
Payload	The element that contains the transaction data to return.	Payload (0...1)

Transaction

Element	Description	Type/Count
Action	The action value. Use this column to override the default value if you have custom actions that should be performed on the transaction by the system. The default is batch created.	int (0...1)
AgencyId	The agency ID.	string (0...1)
ApprovalState	The data related to the approval process workflow: Draft = 10 Pending Approval = 20 Approved = 30 Rejected = 40 Pending Distribution = 50 Distributed = 60	string (0...1)
ArcKey	The archive key.	string (0...1)
ArcTime	The archive time.	dateTime (0...1)
BeginTime	The date and time the transaction processing began.	dateTime (0...1)
CreateTime	The date and time the transaction was created.	dateTime (0...1)
CurrGroup	The group of the current user of the transaction.	int (0...1)
CurrRole	The role of the current user of the transaction.	string (0...1)
CurrSuper	The supervisor of the current user of the transaction.	int (0...1)
CurrUser	The current user of the transaction.	int (0...1)
Customized	A numerical flag of zero (0) or one (1), where one (1) means Yes. This is used to indicate if the transaction is customized. Reserved for future use.	int (0...1)
Data	The print streams for a transaction. There may not be any if... • There are errors • The transaction is sent to manual batch • The transaction is scheduled for later processing	double (0...unbounded)
Descr	The transaction description.	string (0...1)
DocSubType	The document sub-type of the transaction.	string (0...1)
DocType	The document type of the transaction.	string (0...1)

Element	Description	Type/Count
EndTime	The date and time the transaction processing ended.	dateTime (0...1)
FormsetId	The form set unique identifier.	string (0...1)
FromGroup	The group that assigned the transaction.	int (0...1)
FromTime	The date and time the transaction was assigned from a user or group.	dateTime (0...1)
FromUser	The user who assigned the transaction.	int (0...1)
InUse	The in-use flag. Only set this value if you wish to lock the transaction.	string (0...1)
Job_Id	The unique identifier for the job this transaction belongs to. This is a foreign key to a job in the Jobs table.	int (0...1)
Jurisdictn	The transaction jurisdiction code.	string (0...1)
Key1	The key 1 value.	string (0...1)
Key2	The key 2 value.	string (0...1)
Key3	The key 3 value.	string (0...1)
KeyId	The key ID value.	string (0...1)
LocId	The location ID.	string (0...1)
ModifyTime	The last date and time the transaction was modified.	dateTime (0...1)
OrigUser	The original user or creator/author of the transaction.	int (0...1)
ProcessName	The process name that created this transaction. This is normally set by the application that inserts the job but can be overridden.	string (0...1)
QueueId	The queue identifier.	string (0...1)
Reason_Id	The reason code for routing rejection or processing.	string (0...1)
RecType	The record type.	string (0...1)
Retention	A date and time stamp that indicates how long to retain the transaction.	dateTime (0...1)
RouteDesc	The code that indicates the reason why a document was routed or rejected.	string (0...1)
SecLevel	The security level for the transaction.	int (0...1)
StatusCode	The status code of the transaction. The value is set by the system to be W, E, B, or P as part of Document Factory processing but can be overwritten if provided here.	string (0...1)
SubLocId	The sub-location ID.	string (0...1)
ToGroup	The group the transaction was assigned to.	int (0...1)

Element	Description	Type/Count
ToTime	The date and time the transaction was assigned to a user or group.	dateTime (0...1)
ToUser	The user the transaction was assigned to.	int (0...1)
TranCode	The transaction code for the transaction.	string (0...1)
TrnAppDate001	An application-defined date and time column.	dateTime (0...1)
TrnAppDate002	An application-defined date and time column.	dateTime (0...1)
TrnAppDate003	An application-defined date and time column.	dateTime (0...1)
TrnAppDate004	An application-defined date and time column.	dateTime (0...1)
TrnAppDate005	An application-defined date and time column.	dateTime (0...1)
TrnAppDec001	An application-defined decimal column.	double (0...1)
TrnAppDec002	An application-defined decimal column.	double (0...1)
TrnAppDec003	An application-defined decimal column.	double (0...1)
TrnAppDec004	An application-defined decimal column.	double (0...1)
TrnAppDec005	An application-defined decimal column.	double (0...1)
TrnAppInt001	An application-defined signed integer column.	int (0...1)
TrnAppInt002	An application-defined signed integer column.	int (0...1)
TrnAppInt003	An application-defined signed integer column.	int (0...1)
TrnAppInt004	An application-defined signed integer column.	int (0...1)
TrnAppInt005	An application-defined signed integer column.	int (0...1)
TrnAppStr001	An application-defined string column.	string (0...1)
TrnAppStr002	An application-defined string column.	string (0...1)
TrnAppStr003	An application-defined string column.	string (0...1)
TrnAppStr004	An application-defined string column.	string (0...1)
TrnAppStr005	An application-defined string column.	string (0...1)
TrnAppStr006	An application-defined string column.	string (0...1)
TrnAppStr007	An application-defined string column.	string (0...1)
TrnAppStr008	An application-defined string column.	string (0...1)
TrnAppStr009	An application-defined string column.	string (0...1)
TrnAppStr010	An application-defined string column.	string (0...1)
TrnAppStr011	An application-defined string column.	string (0...1)

Element	Description	Type/Count
TrnAppStr012	An application-defined string column.	string (0...1)
TrnAppStr013	An application-defined string column.	string (0...1)
TrnAppStr014	An application-defined string column.	string (0...1)
TrnAppStr015	An application-defined string column.	string (0...1)
TrnBchErr	The number of batches for the transaction that encountered errors while processing through the system.	int (0...1)
TrnBchProc	The number of batches for the transaction that were successfully processed to completion by the system.	int (0...1)
TrnBchSch	The number of batches for the transaction that were scheduled.	int (0...1)
TrnBchTotal	The total number of batches for the transaction.	int (0...1)
TrnCusDate001	A custom date and time column.	dateTime (0...1)
TrnCusDate002	A custom date and time column.	dateTime (0...1)
TrnCusDate003	A custom date and time column.	dateTime (0...1)
TrnCusDate004	A custom date and time column.	dateTime (0...1)
TrnCusDate005	A custom date and time column.	dateTime (0...1)
TrnCusDate006	A custom date and time column.	dateTime (0...1)
TrnCusDate007	A custom date and time column.	dateTime (0...1)
TrnCusDate008	A custom date and time column.	dateTime (0...1)
TrnCusDate009	A custom date and time column.	dateTime (0...1)
TrnCusDate010	A custom date and time column.	dateTime (0...1)
TrnCusDate011	A custom date and time column.	dateTime (0...1)
TrnCusDate012	A custom date and time column.	dateTime (0...1)
TrnCusDate013	A custom date and time column.	dateTime (0...1)
TrnCusDate014	A custom date and time column.	dateTime (0...1)
TrnCusDate015	A custom date and time column.	dateTime (0...1)
TrnCusDec001	A custom decimal column.	double (0...1)
TrnCusDec002	A custom decimal column.	double (0...1)
TrnCusDec003	A custom decimal column.	double (0...1)
TrnCusDec004	A custom decimal column.	double (0...1)
TrnCusDec005	A custom decimal column.	double (0...1)

Element	Description	Type/Count
TrnCusInt001	A custom signed integer column.	int (0...1)
TrnCusInt002	A custom signed integer column.	int (0...1)
TrnCusInt003	A custom signed integer column.	int (0...1)
TrnCusInt004	A custom signed integer column.	int (0...1)
TrnCusInt005	A custom signed integer column.	int (0...1)
TrnCusStr001	A custom string column.	string (0...1)
TrnCusStr002	A custom string column.	string (0...1)
TrnCusStr003	A custom string column.	string (0...1)
TrnCusStr004	A custom string column.	string (0...1)
TrnCusStr005	A custom string column.	string (0...1)
TrnCusStr006	A custom string column.	string (0...1)
TrnCusStr007	A custom string column.	string (0...1)
TrnCusStr008	A custom string column.	string (0...1)
TrnCusStr009	A custom string column.	string (0...1)
TrnCusStr010	A custom string column.	string (0...1)
TrnCusStr011	A custom string column.	string (0...1)
TrnCusStr012	A custom string column.	string (0...1)
TrnCusStr013	A custom string column.	string (0...1)
TrnCusStr014	A custom string column.	string (0...1)
TrnCusStr015	A custom string column.	string (0...1)
TrnCusStr016	A custom string column.	string (0...1)
TrnCusStr017	A custom string column.	string (0...1)
TrnCusStr018	A custom string column.	string (0...1)
TrnCusStr019	A custom string column.	string (0...1)
TrnCusStr020	A custom string column.	string (0...1)
TrnDoLog	A value of zero (0) or one (1) that controls when the TrnsLog table is updated. Set this value to one (1) to enable logging for this and future actions on this transaction into the TrnsLog table unless a subsequent process disables logging. The default is zero (0), which disables logging.	int (0...1)
TrnEndTime	The transaction ending date and time.	dateTime (0...1)

Element	Description	Type/Count
trnErr_Id	The transaction error ID when an error was encountered during processing.	int (0...1)
TrnHistorical	A numerical value of zero (0) or one (1), where one (1) means Yes. This indicates whether the transaction is in the TrnsHist history table.	int (0...1)
TrnHistory	A numerical value of zero (0) or one (1), where one (1) means Yes. This controls whether the Trns table data should be copied to the history table upon deletion. It can be set to zero (0) by an application before deletion. The default is one (1).	int (0...1)
TrnName	The name of the transaction.	string (0...1)
TrnRcpErr	The number of recipients for the transaction that encountered errors while processing through the system.	int (0...1)
TrnRcpProc	The number of recipients for the transaction that were successfully processed to completion by the system.	int (0...1)
TrnRcpSch	The number of recipients for the transaction that were scheduled due to one or more of its associated batches being scheduled.	int (0...1)
TrnRcpTotal	The total number of recipients for the transaction.	int (0...1)
TrnStartTime	The transaction starting date and time.	dateTime (0...1)
TrnStatus	A numerical value that indicates the overall status of the transaction as it is being processed through the system.	int (0...1)
Trn_Id	The unique identifier for the transaction. This is the primary key for the Trns table.	int (0...1)
Unique_Id	A unique identifier string that can be used for application look up of the transaction.	string (0...1)

ServiceInfo

Contains information pertaining the service operation invoked.

Element	Description	Type/Count
Operation	The name of the web service operation invoked.	string (1)
Version	Contains information about the version of the service operation invoked.	Version (1...many)

Version

Contains information pertaining the version of the service operation invoked.

Element	Description	Type/Count
Number	The service version number	int (1)

Element	Description	Type/Count
Used	A boolean value that indicates if the current version was used during the service operation invocation. True means this version was used.	boolean (1)

Handling Errors

The doPublishFromImport service operation returns a Publishing Fault Exception when there is an error.

PublishingFault Schema

Element	Description	Type/Count
faultInfo	Detailed information about the error. Usually a stack trace.	string (1)
message	Brief information about the error. Usually an application generated message.	string (1)

Example Payloads

Here are some payload examples:

Request Payload 1

This example shows how to submit an extract schema element with a file attachment that may contain the extract data for more than one transaction. The format of this file can be a single valid XML file, a stacked XML file or a flat file.

See *Input Formats* on page 142 for more information about the supported formats. See the Documaker Administration Guide for more information regarding extract files.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cmn="oracle/documaker/schema/common"
xmlns:req="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/request"
xmlns:pubcmn="oracle/documaker/schema/ws/publishing/common"
xmlns:tns="oracle/documaker/schema/ws/publishing"
xmlns:v1="oracle/documaker/schema/ws/publishing/doPublishFromImport/
v1">
  <soap:Body>
    <tns:doPublishFromImportRequest>
      <tns:doPublishFromImportRequestV1>
        <pubcmn:timeoutMillis>110000</pubcmn:timeoutMillis>
        <v1:JobRequest>
          <req:Payload>
            <req:Extract>
              <cmn:Content>
                <cmn:URILocation>
                  <cmn:Location>Server</cmn:Location>
                  <cmn:URI>file:///oracle/oracle_insurance_1/
documaker/mstrres/dmres/input/extrfile.xml</cmn:URI>
                </cmn:URILocation>
```

```
              </cmn:Content>
            </req:Extract>
          </req:Payload>
        </v1:JobRequest>
        <v1:ResponseProperties>
          <cmn:ResponseType>Attachments</cmn:ResponseType>
        </v1:ResponseProperties>
      </tns:doPublishFromImportRequestV1>
    </tns:doPublishFromImportRequest>
  </soap:Body>
</soap:Envelope>
```

Response Payload 1

In this example, most of the binary base64 encoded data in the Binary element has been omitted for brevity. Also, several Transaction elements have been omitted for brevity.

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns5:doPublishFromImportResponse
    xmlns:ns6="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/response"
    xmlns:ns5="oracle/documaker/schema/ws/publishing"
    xmlns:ns4="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/request"
    xmlns:ns3="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1"
    xmlns:ns2="oracle/documaker/schema/common"
    xmlns="oracle/documaker/schema/ws/publishing/common">
      <ns5:doPublishFromImportResponseV1>
        <Result>0</Result>
        <ServiceTimeMillis>21969</ServiceTimeMillis>
        <ns3:JobResponse>
          <ns6:JobBchErr>0</ns6:JobBchErr>
          <ns6:JobBchProc>6</ns6:JobBchProc>
          <ns6:JobBchSch>12</ns6:JobBchSch>
          <ns6:JobBchTotal>18</ns6:JobBchTotal>
          <ns6:JobHistorical>0</ns6:JobHistorical>
          <ns6:JobHistory>1</ns6:JobHistory>
          <ns6:JobPayloadType>0</ns6:JobPayloadType>
          <ns6:JobPriority>10</ns6:JobPriority>
          <ns6:JobRcpErr>0</ns6:JobRcpErr>
          <ns6:JobRcpProc>6</ns6:JobRcpProc>
          <ns6:JobRcpSch>12</ns6:JobRcpSch>
          <ns6:JobRcpTotal>18</ns6:JobRcpTotal>
          <ns6:JobStartTime>2011-04-12T15:53:03.806Z</
ns6:JobStartTime>
          <ns6:JobStatus>290</ns6:JobStatus>
          <ns6:JobTrnErr>0</ns6:JobTrnErr>
          <ns6:JobTrnProc>0</ns6:JobTrnProc>
          <ns6:JobTrnSch>6</ns6:JobTrnSch>
          <ns6:JobTrnStartTime>2011-04-12T15:53:04.400Z</
ns6:JobTrnStartTime>
          <ns6:JobTrnTotal>12</ns6:JobTrnTotal>
          <ns6:JobTrnWip>6</ns6:JobTrnWip>
          <ns6:JobUnique_Id>9110e261-c40a-4cd2-ac5c-aee54e09d656</
ns6:JobUnique_Id>
          <ns6:Job_Id>14</ns6:Job_Id>
          <ns6:Payload>
            <ns6:Transaction>
              <ns6:Action>100011</ns6:Action>
              <ns6:ApprovalState>40</ns6:ApprovalState>
```

```
              <ns6:CreateTime>2011-04-12T15:53:04.000Z</
ns6:CreateTime>
              <ns6:CurrGroup>3</ns6:CurrGroup>
              <ns6:CurrUser>8</ns6:CurrUser>
              <ns6:Customized>0</ns6:Customized>
              <ns6:Descr>Welcome Packet</ns6:Descr>
              <ns6:FormsetId>0a503761-599e-42ca-b4b5-abf34f699eb7</
ns6:FormsetId>
              <ns6:Job_Id>14</ns6:Job_Id>
              <ns6:Key1>Central</ns6:Key1>
              <ns6:Key2>Account_Status</ns6:Key2>
              <ns6:KeyId>0000001</ns6:KeyId>
              <ns6:ModifyTime>2011-04-12T15:53:05.000Z</
ns6:ModifyTime>
              <ns6:OrigUser>8</ns6:OrigUser>
              <ns6:ProcessName>Identifier</ns6:ProcessName>
              <ns6:RecType>00</ns6:RecType>
              <ns6:RouteDesc>DM20030:   the following required field
s are missing data: AGENTCITYSTATEZIP.</ns6:RouteDesc>
              <ns6:SecLevel>0</ns6:SecLevel>
              <ns6:StatusCode>W</ns6:StatusCode>
              <ns6:TranCode>NB</ns6:TranCode>
              <ns6:TrnBchErr>0</ns6:TrnBchErr>
              <ns6:TrnBchProc>0</ns6:TrnBchProc>
              <ns6:TrnBchSch>0</ns6:TrnBchSch>
              <ns6:TrnBchTotal>0</ns6:TrnBchTotal>
              <ns6:TrnDoLog>0</ns6:TrnDoLog>
              <ns6:TrnHistorical>0</ns6:TrnHistorical>
              <ns6:TrnHistory>1</ns6:TrnHistory>
              <ns6:TrnRcpErr>0</ns6:TrnRcpErr>
              <ns6:TrnRcpProc>0</ns6:TrnRcpProc>
              <ns6:TrnRcpSch>0</ns6:TrnRcpSch>
              <ns6:TrnRcpTotal>0</ns6:TrnRcpTotal>
              <ns6:TrnStartTime>2011-04-12T15:53:04.197Z</
ns6:TrnStartTime>
              <ns6:TrnStatus>290</ns6:TrnStatus>
              <ns6:Trn_Id>15</ns6:Trn_Id>
              <ns6:Unique_Id>0a503761-599e-42ca-b4b5-abf34f699eb7</
ns6:Unique_Id>
            </ns6:Transaction>
            <ns6:Transaction>
              <ns6:Action>100011</ns6:Action>
              <ns6:ApprovalState>40</ns6:ApprovalState>
              <ns6:CreateTime>2011-04-12T15:53:04.000Z</
ns6:CreateTime>
              <ns6:CurrGroup>3</ns6:CurrGroup>
              <ns6:CurrUser>8</ns6:CurrUser>
              <ns6:Customized>0</ns6:Customized>
              <ns6:Descr>Welcome Packet</ns6:Descr>
              <ns6:FormsetId>0bd3df4c-e2fe-4e4a-ba76-293ccd9bcea0</
ns6:FormsetId>
            <ns6:Transaction>
              <ns6:Action>100011</ns6:Action>
              <ns6:ApprovalState>10</ns6:ApprovalState>
              <ns6:CreateTime>2011-04-12T15:53:04.000Z</
ns6:CreateTime>
              <ns6:CurrGroup>3</ns6:CurrGroup>
              <ns6:CurrUser>8</ns6:CurrUser>
              <ns6:Customized>0</ns6:Customized>
              <ns6:Data>
                <ns2:Name>30_9</ns2:Name>
                <ns2:ContentType>message/rfc822</ns2:ContentType>
                <ns2:FileType>htm</ns2:FileType>
                <ns2:Content>
```

```
                    <ns2:Binary>TUlNRS12Z...</ns2:Content>
                </ns6:Data>
                <ns6:Descr>Welcome Packet</ns6:Descr>
                <ns6:FormsetId>26de30ad-8d65-440a-90e1-03af4bc3c323</
ns6:FormsetId>
                <ns6:Job_Id>14</ns6:Job_Id>
                <ns6:Key1>Central</ns6:Key1>
                <ns6:Key2>Account_Status</ns6:Key2>
                <ns6:KeyId>0000007</ns6:KeyId>
                <ns6:ModifyTime>2011-04-12T15:53:06.000Z</
ns6:ModifyTime>
                <ns6:OrigUser>8</ns6:OrigUser>
                <ns6:ProcessName>Batcher</ns6:ProcessName>
                <ns6:RecType>00</ns6:RecType>
                <ns6:SecLevel>0</ns6:SecLevel>
                <ns6:StatusCode>P</ns6:StatusCode>
                <ns6:TranCode>null</ns6:TranCode>
                <ns6:TrnBchErr>0</ns6:TrnBchErr>
                <ns6:TrnBchProc>1</ns6:TrnBchProc>
                <ns6:TrnBchSch>2</ns6:TrnBchSch>
                <ns6:TrnBchTotal>3</ns6:TrnBchTotal>
                <ns6:TrnDoLog>0</ns6:TrnDoLog>
                <ns6:TrnHistorical>0</ns6:TrnHistorical>
                <ns6:TrnHistory>1</ns6:TrnHistory>
                <ns6:TrnRcpErr>0</ns6:TrnRcpErr>
                <ns6:TrnRcpProc>1</ns6:TrnRcpProc>
                <ns6:TrnRcpSch>2</ns6:TrnRcpSch>
                <ns6:TrnRcpTotal>3</ns6:TrnRcpTotal>
                <ns6:TrnStartTime>2011-04-12T15:53:04.260Z</
ns6:TrnStartTime>
                <ns6:TrnStatus>416</ns6:TrnStatus>
                <ns6:Trn_Id>21</ns6:Trn_Id>
                <ns6:Unique_Id>26de30ad-8d65-440a-90e1-03af4bc3c323</
ns6:Unique_Id>
            </ns6:Transaction>
            ...
            ...
        </ns6:Payload>
        </ns3:JobResponse>
        <ns3:ServiceInfo>
          <ns2:Operation>doPublishFromImport</ns2:Operation>
          <ns2:Version>
            <ns2:Number>1</ns2:Number>
            <ns2:Used>true</ns2:Used>
          </ns2:Version>
        </ns3:ServiceInfo>
      </ns5:doPublishFromImportResponseV1>
    </ns5:doPublishFromImportResponse>
  </S:Body>
</S:Envelope>
```

Request Payload 2

This example shows how to submit a Transaction schema element with a file attachment that contains the extract data for that transaction. The format of the extract data must be a single valid XML file or flat file data for one transaction only.

Most of the binary base64 encoded data in the Binary element has been omitted for brevity.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cmn="oracle/documaker/schema/common"
xmlns:req="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/request"
xmlns:pubcmn="oracle/documaker/schema/ws/publishing/common"
xmlns:tns="oracle/documaker/schema/ws/publishing"
xmlns:v1="oracle/documaker/schema/ws/publishing/doPublishFromImport/
v1">
  <soap:Body>
    <tns:doPublishFromImportRequest>
      <tns:doPublishFromImportRequestV1>
        <pubcmn:timeoutMillis>90000</pubcmn:timeoutMillis>
        <v1:JobRequest>
          <req:Payload>
            <req:Transaction>
              <req:Data>
                <cmn:Content>
                  <cmn:Binary>PD94bWw...</cmn:Binary>
                </cmn:Content>
              </req:Data>
            </req:Transaction>
          </req:Payload>
        </v1:JobRequest>
        <v1:ResponseProperties>
          <cmn:ResponseType>Attachments</cmn:ResponseType>
    </v1:ResponseProperties>
      </tns:doPublishFromImportRequestV1>
    </tns:doPublishFromImportRequest>
  </soap:Body>
</soap:Envelope>
```

Response Payload 2

In this example, most of the binary base64 encoded data in the Binary element has been omitted for brevity.

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns5:doPublishFromImportResponse xmlns:ns6="oracle/documaker/
schema/ws/publishing/doPublishFromImport/v1/
response" xmlns:ns5="oracle/documaker/schema/ws/
publishing" xmlns:ns4="oracle/documaker/schema/ws/publishing/
doPublishFromImport/v1/request" xmlns:ns3="oracle/documaker/schema/
ws/publishing/doPublishFromImport/v1" xmlns:ns2="oracle/documaker/
schema/common" xmlns="oracle/documaker/schema/ws/publishing/common">
      <ns5:doPublishFromImportResponseV1>
        <Result>0</Result>
        <ServiceTimeMillis>14454</ServiceTimeMillis>
        <ns3:JobResponse>
          <ns6:JobBchErr>0</ns6:JobBchErr>
          <ns6:JobBchProc>1</ns6:JobBchProc>
          <ns6:JobBchSch>2</ns6:JobBchSch>
          <ns6:JobBchStartTime>2011-04-12T15:45:43.260Z</
ns6:JobBchStartTime>
          <ns6:JobBchTotal>3</ns6:JobBchTotal>
          <ns6:JobHistorical>0</ns6:JobHistorical>
          <ns6:JobHistory>1</ns6:JobHistory>
          <ns6:JobPayloadType>0</ns6:JobPayloadType>
          <ns6:JobPriority>10</ns6:JobPriority>
          <ns6:JobRcpErr>0</ns6:JobRcpErr>
          <ns6:JobRcpProc>1</ns6:JobRcpProc>
          <ns6:JobRcpSch>2</ns6:JobRcpSch>
```

```
          <ns6:JobRcpStartTime>2011-04-12T15:45:43.260Z</
ns6:JobRcpStartTime>
          <ns6:JobRcpTotal>3</ns6:JobRcpTotal>
          <ns6:JobStartTime>2011-04-12T15:45:39.728Z</
ns6:JobStartTime>
          <ns6:JobStatus>416</ns6:JobStatus>
          <ns6:JobTrnErr>0</ns6:JobTrnErr>
          <ns6:JobTrnProc>0</ns6:JobTrnProc>
          <ns6:JobTrnSch>1</ns6:JobTrnSch>
          <ns6:JobTrnStartTime>2011-04-12T15:45:40.119Z</
ns6:JobTrnStartTime>
          <ns6:JobTrnTotal>1</ns6:JobTrnTotal>
          <ns6:JobTrnWip>0</ns6:JobTrnWip>
          <ns6:JobUnique_Id>1b6d8297-2f5b-48f5-9c11-3ef8a0f5636c</
ns6:JobUnique_Id>
          <ns6:Job_Id>6</ns6:Job_Id>
          <ns6:Payload>
            <ns6:Transaction>
              <ns6:Action>100011</ns6:Action>
              <ns6:ApprovalState>10</ns6:ApprovalState>
              <ns6:CreateTime>2011-04-12T15:45:40.000Z</
ns6:CreateTime>
              <ns6:CurrGroup>3</ns6:CurrGroup>
              <ns6:CurrUser>8</ns6:CurrUser>
              <ns6:Customized>0</ns6:Customized>
              <ns6:Data>
                <ns2:Name>6_1</ns2:Name>
                <ns2:ContentType>message/rfc822</ns2:ContentType>
                <ns2:FileType>htm</ns2:FileType>
                <ns2:Content>
                  <ns2:Binary>TUlNRS12...</ns2:Binary>
                </ns2:Content>
              </ns6:Data>
              <ns6:Descr>Welcome Packet</ns6:Descr>
              <ns6:FormsetId>1b6d8297-2f5b-48f5-9c11-3ef8a0f5636c</
ns6:FormsetId>
              <ns6:Job_Id>6</ns6:Job_Id>
              <ns6:Key1>Central</ns6:Key1>
              <ns6:Key2>Account_Status</ns6:Key2>
              <ns6:KeyId>0000004</ns6:KeyId>
              <ns6:ModifyTime>2011-04-12T15:45:42.000Z</
ns6:ModifyTime>
              <ns6:OrigUser>8</ns6:OrigUser>
              <ns6:ProcessName>Batcher</ns6:ProcessName>
              <ns6:RecType>00</ns6:RecType>
              <ns6:SecLevel>0</ns6:SecLevel>
              <ns6:StatusCode>P</ns6:StatusCode>
              <ns6:TranCode>null</ns6:TranCode>
              <ns6:TrnBchErr>0</ns6:TrnBchErr>
              <ns6:TrnBchProc>1</ns6:TrnBchProc>
              <ns6:TrnBchSch>2</ns6:TrnBchSch>
              <ns6:TrnBchTotal>3</ns6:TrnBchTotal>
              <ns6:TrnDoLog>0</ns6:TrnDoLog>
              <ns6:TrnHistorical>0</ns6:TrnHistorical>
              <ns6:TrnHistory>1</ns6:TrnHistory>
              <ns6:TrnRcpErr>0</ns6:TrnRcpErr>
              <ns6:TrnRcpProc>1</ns6:TrnRcpProc>
              <ns6:TrnRcpSch>2</ns6:TrnRcpSch>
              <ns6:TrnRcpTotal>3</ns6:TrnRcpTotal>
              <ns6:TrnStartTime>2011-04-12T15:45:40.119Z</
ns6:TrnStartTime>
              <ns6:TrnStatus>416</ns6:TrnStatus>
              <ns6:Trn_Id>6</ns6:Trn_Id>
```

```
              <ns6:Unique_Id>1b6d8297-2f5b-48f5-9c11-3ef8a0f5636c</
ns6:Unique_Id>
            </ns6:Transaction>
          </ns6:Payload>
        </ns3:JobResponse>
        <ns3:ServiceInfo>
          <ns2:Operation>doPublishFromImport</ns2:Operation>
          <ns2:Version>
            <ns2:Number>1</ns2:Number>
            <ns2:Used>true</ns2:Used>
          </ns2:Version>
        </ns3:ServiceInfo>
      </ns5:doPublishFromImportResponseV1>
    </ns5:doPublishFromImportResponse>
  </S:Body>
</S:Envelope>
```

Example PublishingFault

Here is an example PublishingFault:

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <S:Fault xmlns:ns4="http://www.w3.org/2003/05/soap-envelope">
      <faultcode>S:Server</faultcode>
      <faultstring>Unable to validate request payload!</faultstring>
      <detail>
        <PublishingFault:PublishingFault
            xmlns:PublishingFault="oracle/documaker/schema/ws/
publishing"
            xmlns="oracle/documaker/schema/ws/publishing"
            xmlns:ns2="oracle/documaker/schema/common"
            xmlns:ns3="oracle/documaker/schema/tables/jobs"
            xmlns:ns4="oracle/documaker/schema/tables/trns"
            xmlns:ns5="oracle/documaker/schema/ws/publishing/
requests">
          <faultInfo>
            cvc-
attribute.3: The value '2.5' of attribute 'schemaVersion' on element
            'tns:doPublishFromImportRequest' is not valid with respe
ct to its type,
            'schemaVersion'.
          </faultInfo>
          <message>Unable to validate request payload!</message>
        </PublishingFault:PublishingFault>
      </detail>
    </S:Fault>
  </S:Body>
</S:Envelope>
```

CONFIGURING DWS

Use the following files and tables to set up Documaker Web Services.

web.xml File

The web.xml deployment descriptor file in WEB-INF directory inside DWS.war file contains several configuration options.

Entry	Description
CONFIG_DS_JNDI_NAME	The JNDI Name of the data source used by composition and publishing service operations to read configuration information from the ALCONFIGCONTEXT and APPCONFIGCONTEXT administration tables for a Document Factory assembly line. The data source must be set up in the application server before deploying DWS.
FACTORY_DS_JNDI_NAME +	The JNDI Name of the data source used by publishing service operations to interface with the tables for a Document Factory assembly line. The data source must be set up in the application server before deploying DWS.
FACTORY_CATALOG +	(Optional) The database catalog name for the Document Factory assembly line tables. The name is case sensitive.
FACTORY_SCHEMA +	(Optional) The database schema name for the Document Factory assembly line tables. The name is case sensitive.
SYSTEM_ID	The system ID value for the administration tables in the Document Factory assembly line that DWS should interface with.
AL_ID	The assembly line ID value for the administration tables in the Document Factory assembly line that DWS should interface with.
CONFIGURATION_FACTORY_CLASS	The implementation class name of the Configuration interface. This is the class that is used to retrieve configuration information from the Document Factory administration tables in an assembly line. You can choose from: • oracle.documaker.config.xml.XMLConfiguration • oracle.documaker.config.jpa.JPAConfiguration • oracle.documaker.config.db.DataSourceConfiguration The default is the fully-qualified class name of the JPAConfiguration class.
XML_DELIMITER +	The XML delimiter for stacked XML files. Here is an example of the format of the search mask: `search-mask,line offset` The default is <?xml,0.
XPATH_DELIMITER +	The xPath delimiter for a valid XML file that contains multiple XML transactions. The default is //DOCUMENT.
TEXT_DELIMITER +	The text delimiter for flat extract files. Here is an example of the format of the search mask: `search-mask,line offset` The default is HEADERREC,10.

* = used by composition services.
+ = used by publishing services.

Here is an example:

```
<context-param>
    <param-name>CONFIG_DS_JNDI_NAME</param-name>
    <param-value>jdbc/DMKRConfig</param-value>
</context-param>
<context-param>
    <param-name>FACTORY_DS_JNDI_NAME</param-name>
    <param-value>jdbc/DMKRFactory</param-value>
</context-param>
<context-param>
    <param-name>FACTORY_SCHEMA</param-name>
    <param-value>DMKR_ASLINE</param-value>
</context-param>
<context-param>
    <param-name>FACTORY_CATALOG</param-name>
    <param-value>DMKR_ASLINE</param-value>
</context-param>
<context-param>
    <param-name>CONFIGURATION_FACTORY_CLASS</param-name>
    <param-value>oracle.documaker.config.jpa.JPAConfiguration</
param-value>
</context-param>
<context-param>
    <param-name>SYSTEM_ID</param-name>
    <param-value>1</param-value>
</context-param>
<context-param>
    <param-name>AL_ID</param-name>
    <param-value>1</param-value>
</context-param>
<context-param>
    <param-name>XML_DELIMITER</param-name>
    <param-value><?xml,0</param-value>
</context-param>
<context-param>
    <param-name>XPATH_DELIMITER</param-name>
    <param-value>//DOCUMENT</param-value>
</context-param>
<context-param>
    <param-name>TEXT_DELIMITER</param-name>
    <param-value>HEADERREC,10</param-value>
</context-param>
```

Note Make sure the value of CONFIG_DS_JNDI_NAME matches the non-jta-data-source value of the persistence.xml file inside persistence-config.jar file in WEB-INF/lib directory of the DWS.war file.

log4j.xml File

The log4j.xml file is located in WEB-INF\classes directory and it contains loggers for producing diagnostic output.

Logger	Description
oracle.documaker.ws.handler.LoggingHandler	Logs the input and output SOAP messages for a transaction.
oracle.documaker.ws.server.CompositionService	Logs error/debug information for Composition service operations.

Logger	Description
oracle.documaker.ws.server.PublishingService	Logs error/debug information for Publishing service operations.
oracle.documaker.ws.ids.Proxy	Logs error/debug information for the Docupresentment proxy.
oracle.documaker.ws.config.DWSConfiguration	Logs error/debug information for the DWS configuration.
oracle.documaker.dao.AbstractDAO	Logs error/debug information for the AbstractDAO object.
oracle.documaker.db.Query	Logs error/debug information for JDBC queries.
http.debug	Logs HTTP error/debug information.
mqseries.debug	Logs WebSphere MQ error/debug information.
msmq.debug	Logs WebSphere MQ error/debug information.
jms.debug	Logs JMS error/debug information.

Note Change the Priority value for a logger from *error* to *debug* to produce diagnostic output. You must restart the Documaker Web Service application for any log4j.xml file changes to take effect.

Here is an example of a logger:

```
<category name="jms.debug" additivity="false">
    <priority value="error"/>
    <appender-ref ref="stdout"/>
    <appender-ref ref="roll"/>
</category>
```

ALCONFIGCONTEXT Table

Read from the bus GROUP_NAME column by composition service operations. This is the Document Factory administration table that is installed and configured when a Document Factory assembly line is installed, which is a pre-requisite for composition and publishing services. You can access this table using Documaker Administrator.

Group_Name	Property	Value
Bus	queuefactory.class	Any queue factory class supported by Docupresentment. For example com.docucorp.messaging.mqseries.DSIMQMessageQueueFactory.
Bus	IDSRequestQueue	The name of the Docupresentment request queue.
Bus	IDSResultQueue	The name of the Docupresentment response queue.
Bus	*, where (*) means any other message bus property supported by Docupresentment.	The value of the corresponding property.

Here is an example (only the PROPERTY and VALUE columns are shown):

Property	Value
queuefactory.class	com.docucorp.messaging.jms.DSIJMSJNDIMessageQueueFactory
jms.initial.context.factory	WebLogic.jndi.WLInitialContextFactory
jms.provider.URL	t3://10.140.212.152:7001
jms.qcf.name	jms/qcf
IDSRequestQueue	jms/requestq
IDSResultQueue	jms/resultq
TimeoutSeconds	5

Note See the Internet Document Server Guide for details about the message buses supported and their configuration options.

DEPLOYING DWS

These application servers are supported:

Application server	Version
Oracle WebLogic	11.3.4

Note Note DWS is deployed with the installation of Document Factory and Documaker Interactive: Correspondence. Follow the instructions below if you need to deploy a second instance of these services.

DEPLOYING TO WEBLOGIC

Deploying to WebLogic involves performing these tasks in the WebLogic Administration Console:

- *Creating the JNDI Data Sources* on page 485
- *Deploying the DWS.ear File* on page 492

Note Add the message bus packages Docupresentment uses to the WebLogic DWS.war file if other than WebLogic JMS queues are used.

Creating the JNDI Data Sources

Follow these steps to create the JNDI data sources in the WebLogic Administration Console:

1. Create the JNDI data sources in the WebLogic Administration Console.

   Make sure the JNDI names for the JNDI data sources created in the WebLogic container match the JNDI names for FACTORY_DS_JNDI_NAME and CONFIG_DS_JNDI_NAME web.xml context parameters in WEB-INF directory of the DWS.war file. Here is an example (defaults shown):

```
<context-param>
    <param-name>CONFIG_DS_JNDI_NAME</param-name>
    <param-value>jdbc/DMKRConfig</param-value>
</context-param>

<context-param>
    <param-name>FACTORY_DS_JNDI_NAME</param-name>
    <param-value>jdbc/DMKRFactory</param-value>
</context-param>
```

   Make sure the JNDI names for the JNDI data sources created in the WebLogic container match the res-ref-name web.xml context parameters in WEB-INF directory of the DWS.war file. Here is an example (defaults shown):

```
<resource-ref>
    <res-ref-name>jdbc/DMKRConfig</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
    <res-sharing-scope>Shareable</res-sharing-scope>
</resource-ref>
<resource-ref>
    <res-ref-name>jdbc/DMKRFactory</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
    <res-sharing-scope>Shareable</res-sharing-scope>
</resource-ref>
```

   Make sure the value of CONFIG_DS_JNDI_NAME in web.xml file in WEB-INF directory of the DWS.war file matches the non-jta-data-source value of the persistence.xml file inside persistence-config.jar file in WEB-INF/lib directory of the DWS.war file.

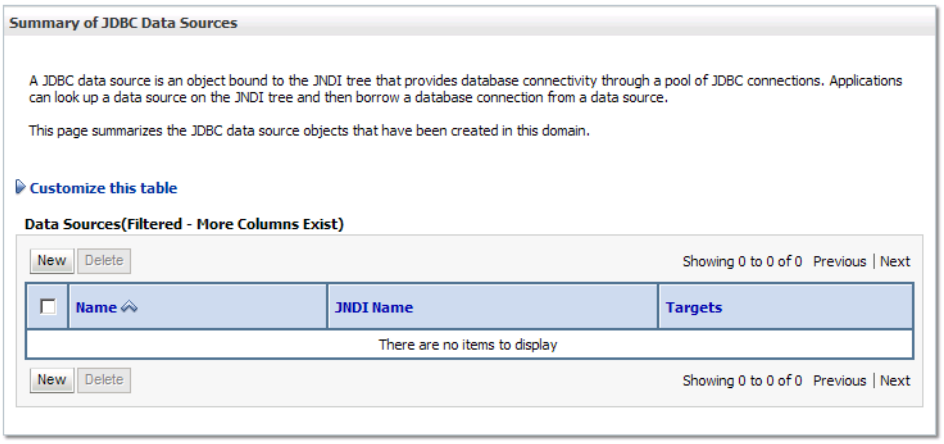
Note Do not add the JDBC driver package to the DWS.war file as you are using container provided JNDI data sources.

2. Log into the WebLogic Administration Console. The URL is typically:

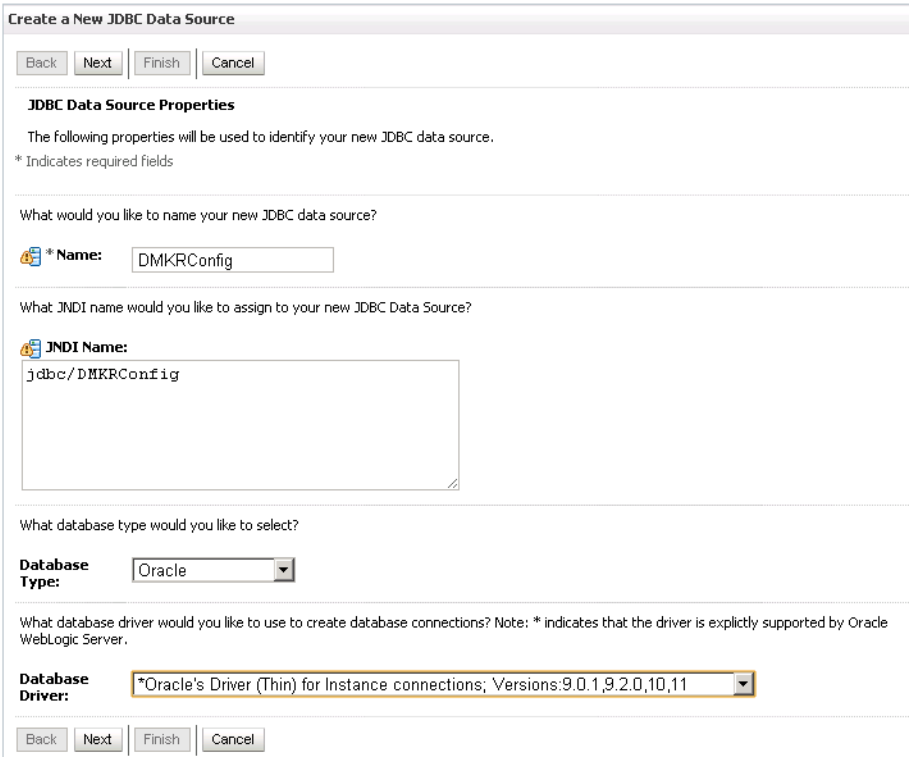
   ```
   http://IpAddress:7001/console
   ```

   where *IpAddress* is the IP address of your container. You will need to provide your credentials to log in.

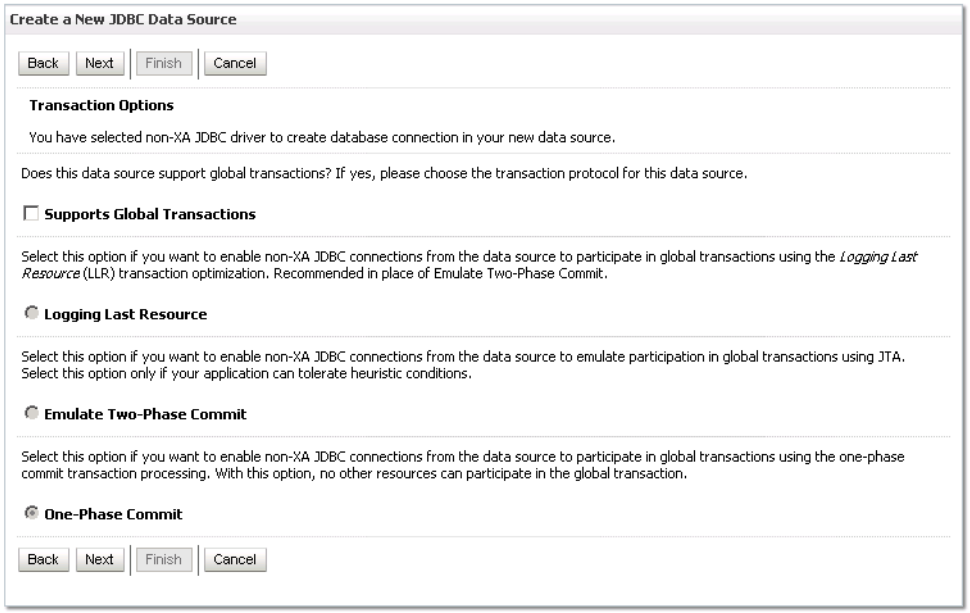
3. Create the DMKRConfig JNDI data source to interface with the Document Factory administration tables. On the left panel of the WebLogic Administration Console, expand Services, JDBC, and click the Data Sources link. On the right panel, click New.



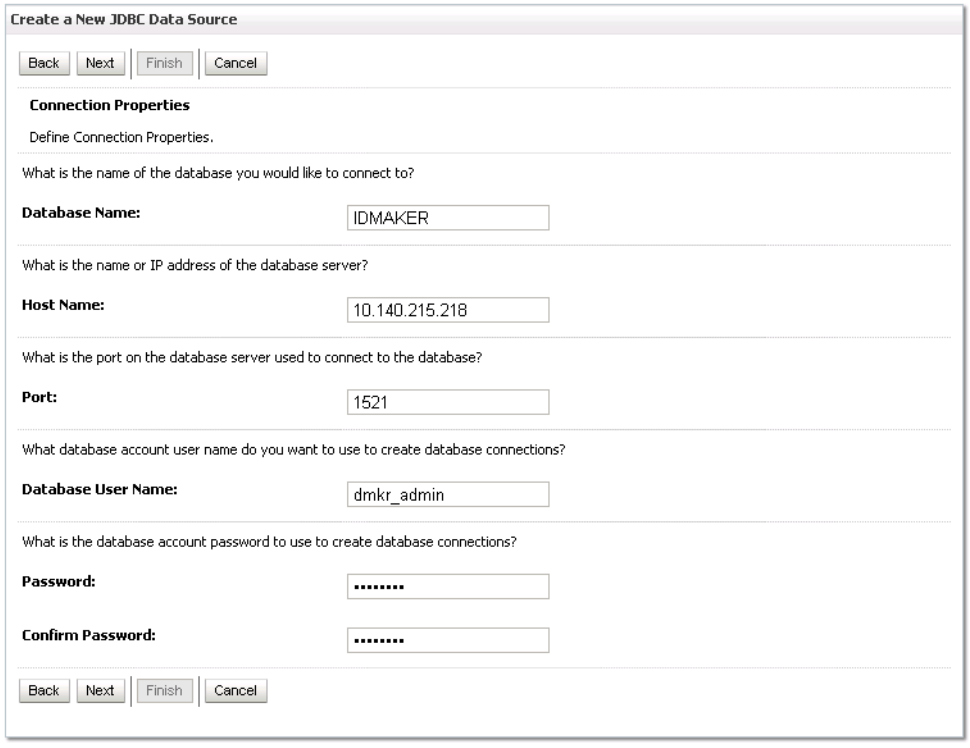
4. Enter values for the Name, JNDI Name, Database Type, and Database Driver fields and click Next.



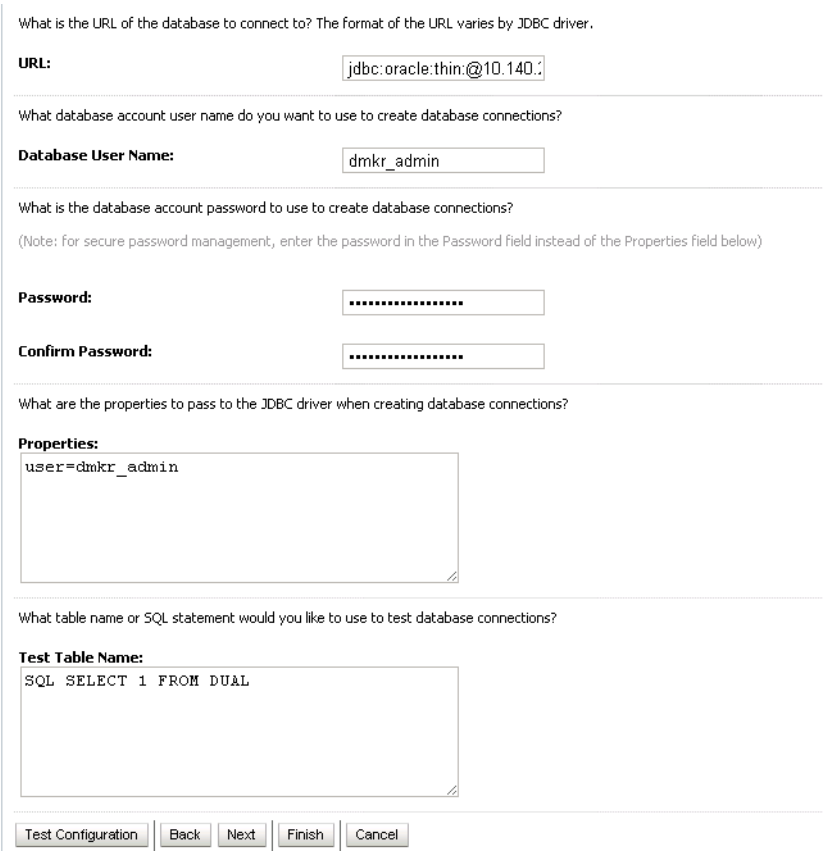
5. Select the options for transaction support and click Next.



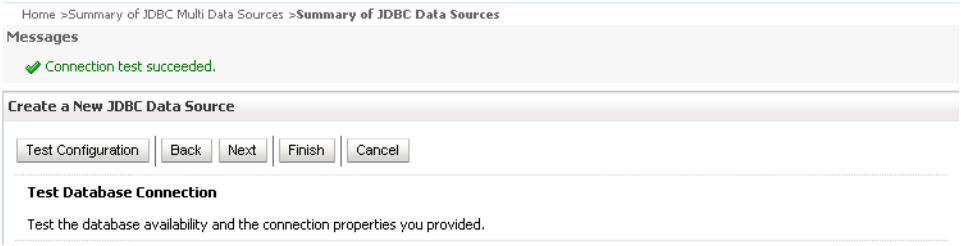
6. Enter the values for the Database Name, Host Name, Port, Database User Name, and Password fields and click Next.



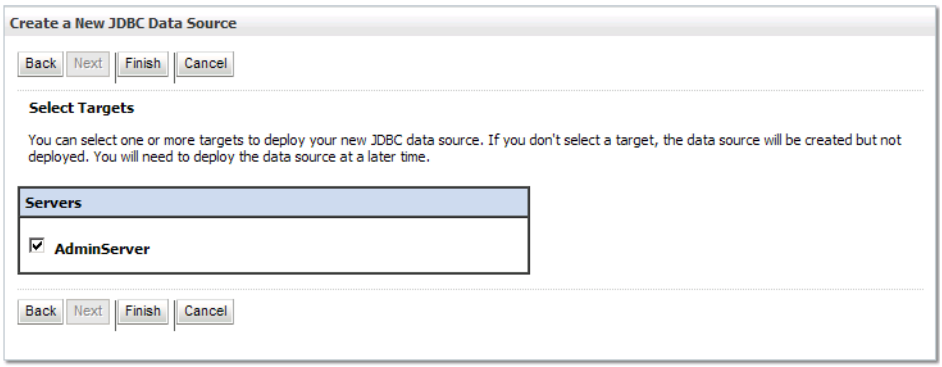
7. Click the Test Configuration button.



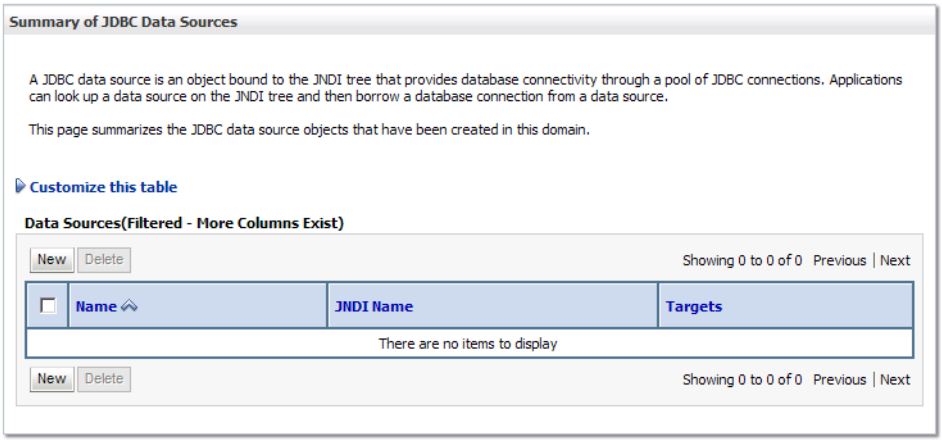
8. Verify the test results were successful and click Next.



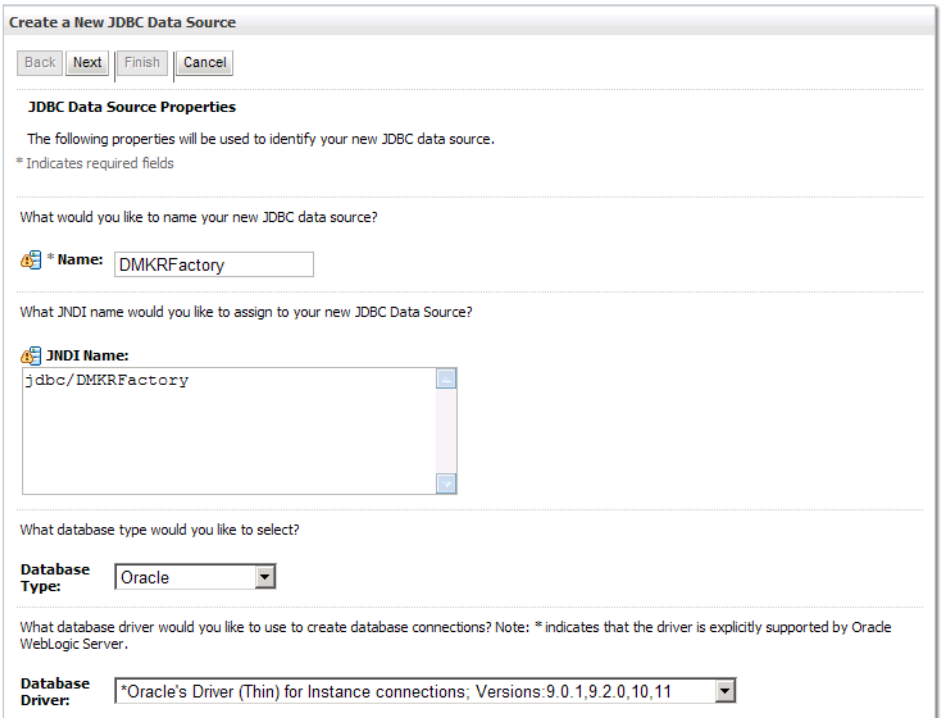
9. Select the target server for the JNDI data source and click Finish.



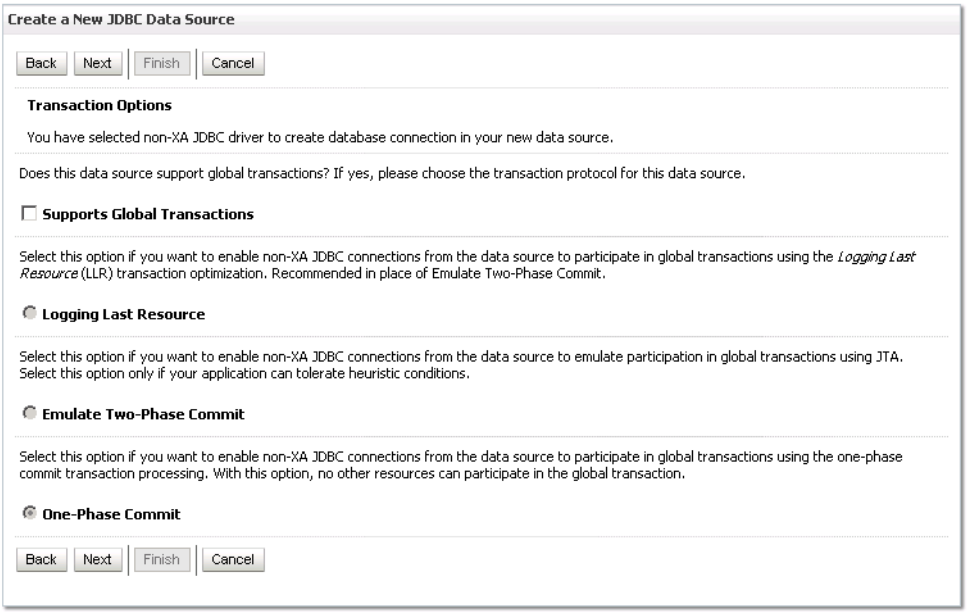
10. Create the DMKRFactory JNDI data source to interface with the Document Factory assembly tables. On the left panel of the WebLogic Administration Console, expand Services, JDBC and click the Data Sources link. On the right panel, click New.



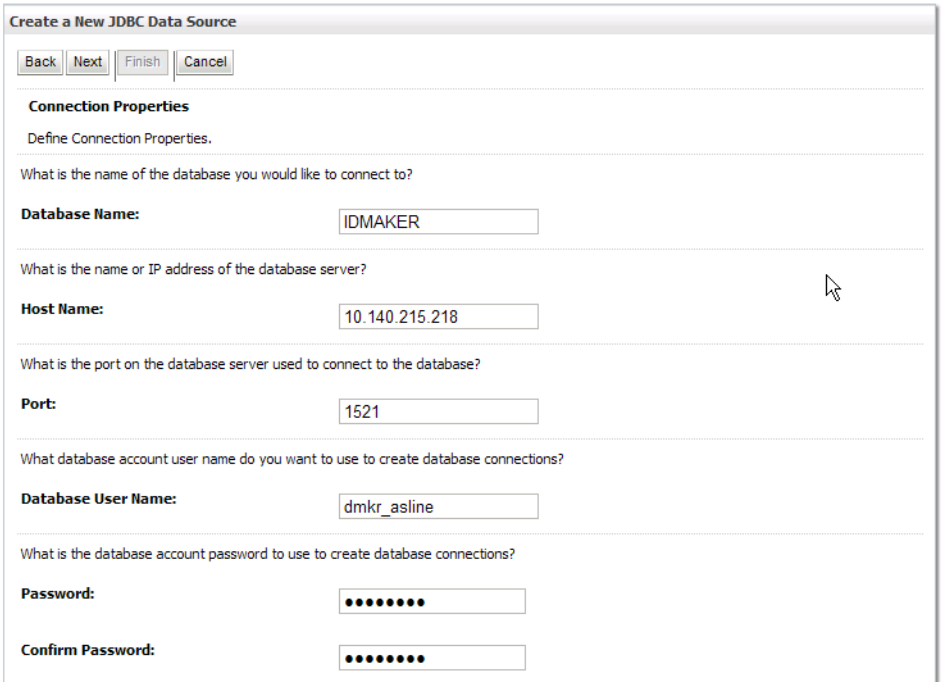
11. Enter the values for the Name, JNDI Name, Database Type, and Database Driver fields and click Next.



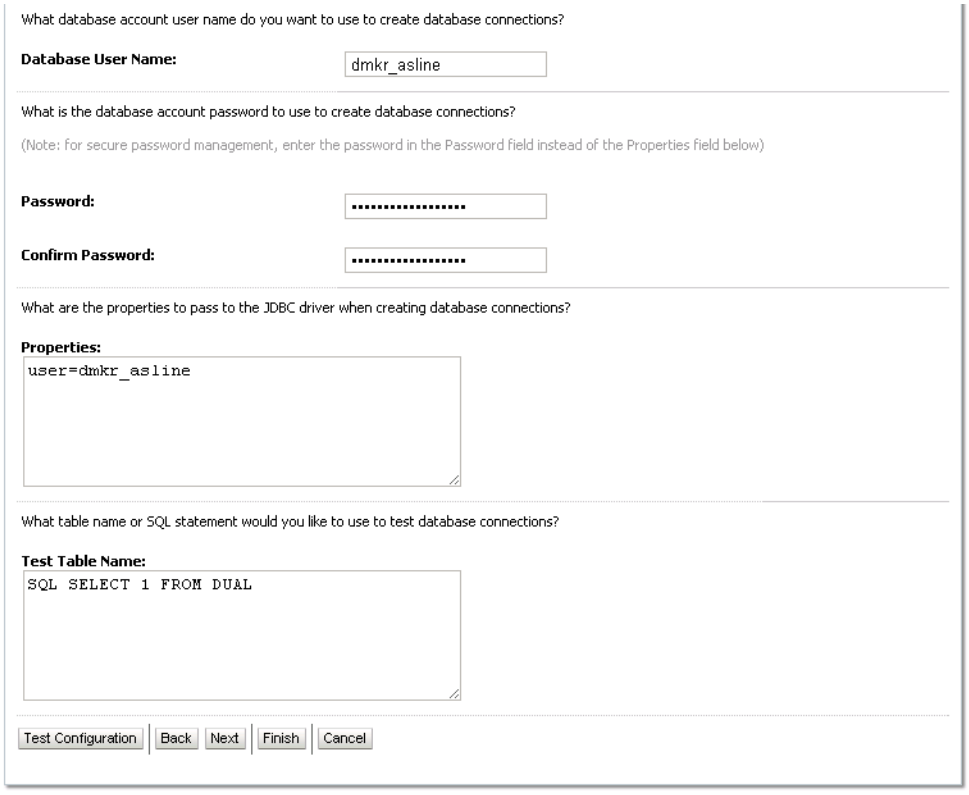
12. Select the options for transaction support and click Next.



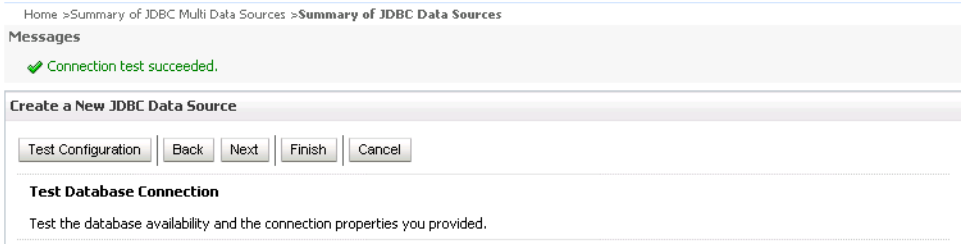
13. Enter the values for the Database Name, Host Name, Port, Database User Name, and Password fields and click Next.



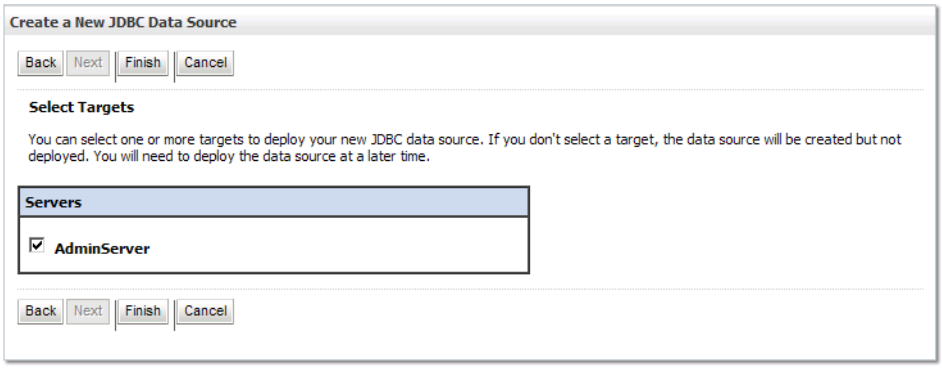
14. Click the Test Configuration button.



15. Verify the test results were successful and click Next.



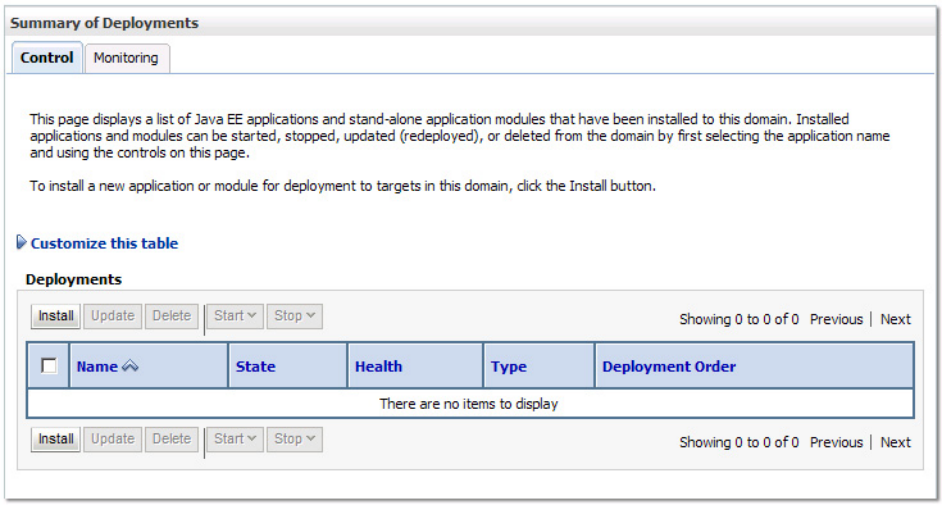
16. Select the target server for the JNDI data source and click Finish.



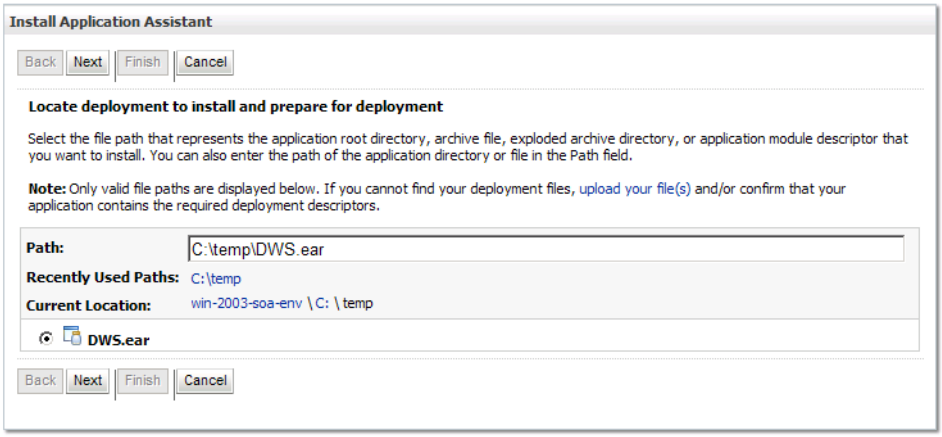
Deploying the DWS.ear File

Follow these steps to deploy the DWS.ear file in the WebLogic Administration Console.

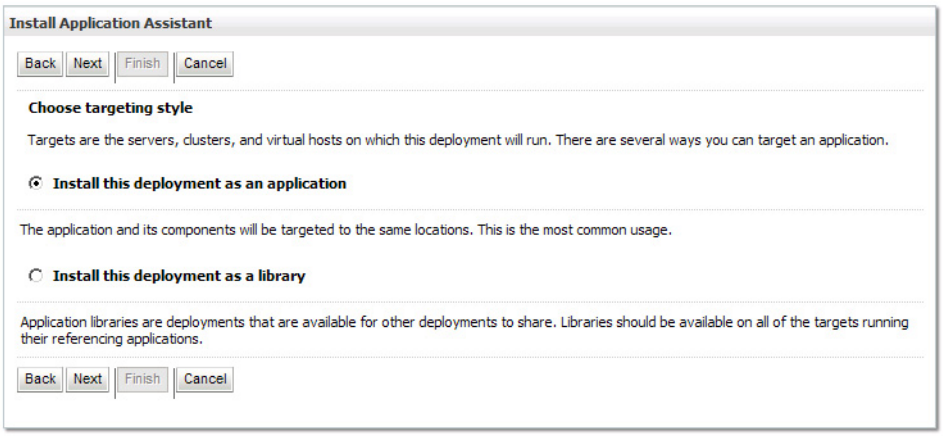
1. Click the Deployments link on the left panel of the WebLogic Administration Console to display the Deployments right panel. Click the Install button.



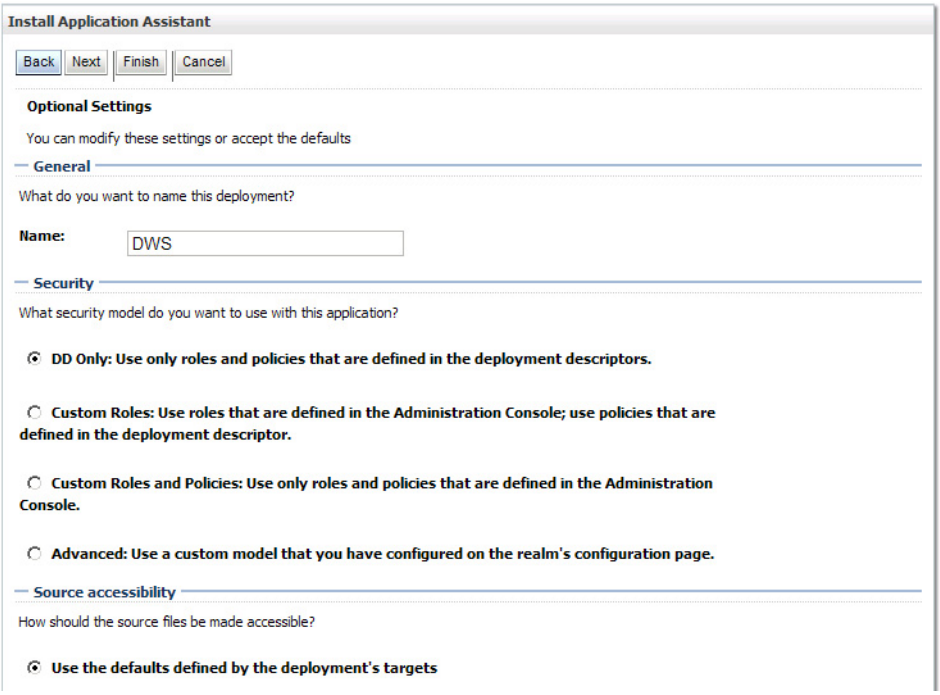
2. Browse to the location of the DWS.ear file and make sure it is selected. Click Next.



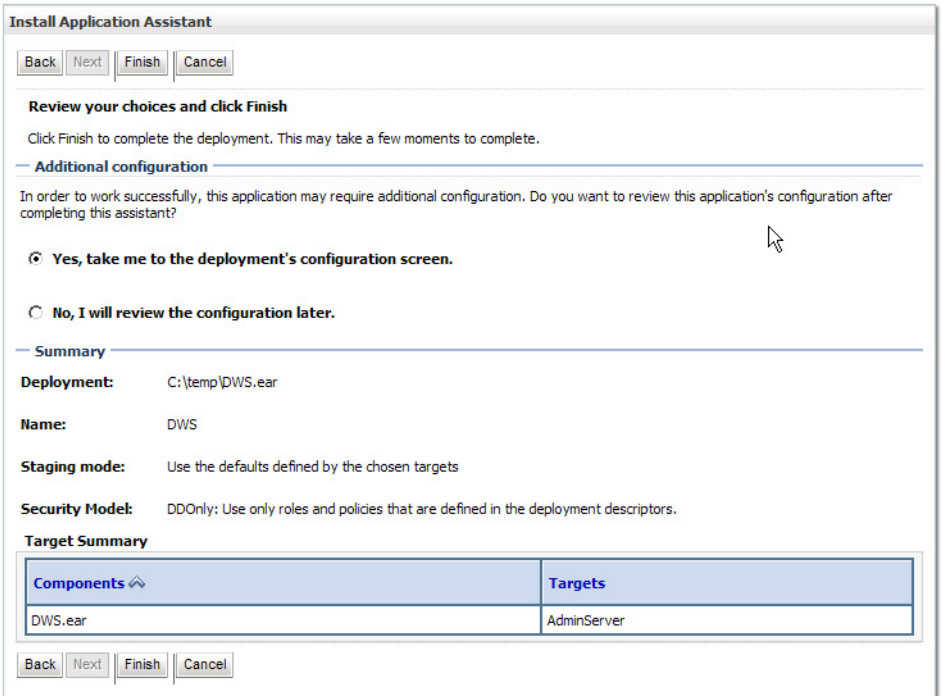
3. Select the *Install this deployment as an application* option and click Next.



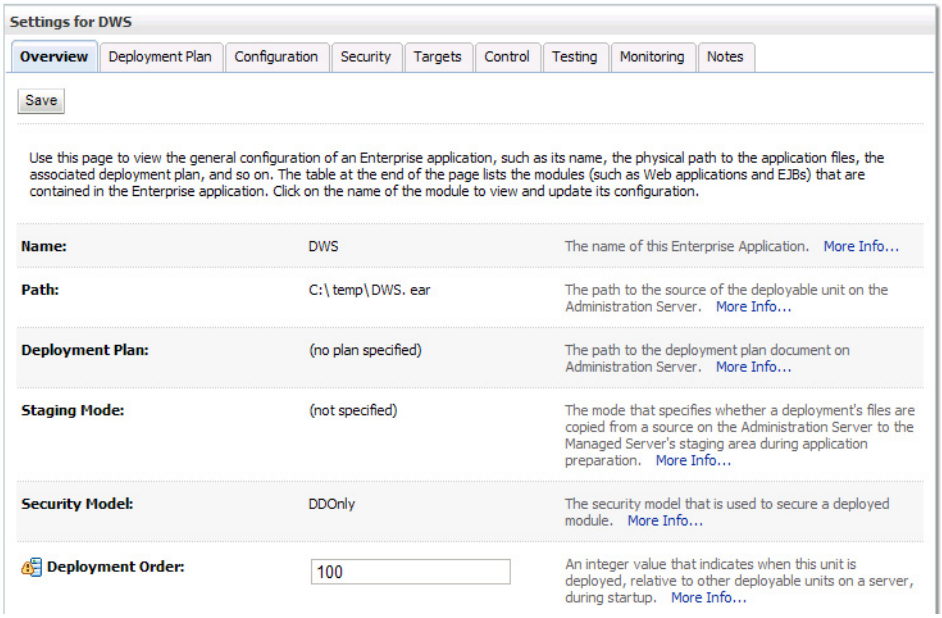
4. Accept the default options and click Next.



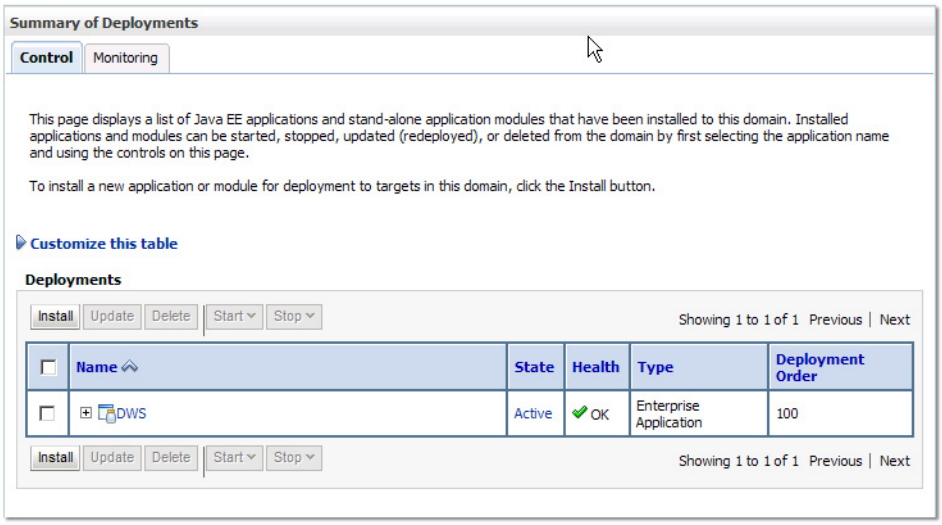
5. To finalize the deployment, select the default options and click Finish.



6. To save the deployment, select the default options and click Save.



7. Go to the Deployments right panel and verify the state is Active and the health is Ok for the DWS application.



When Using an Oracle Database

Keep in mind...

- *Do not* include these files in the WEB-INF/lib directory of the DWS.war file inside the DWS.ear file:
  - ojdbc6.jar
  - xdb.jar
  - xmlparserv2.jar
  - xmltype.jar
- Add these files to the /oracle/middleware/wlserver_10.x/server/lib directory:
  - ojdbc6.jar
  - xdb.jar

  Do not add the xmlparserv2.jar JAR file — the container already includes this file.
- Edit the setDomainEnv.cmd file as follows to add the xdb.jar file to the WebLogic class path:

```
set POST_CLASSPATH=/oracle/middleware/wlserver_10.3/server/lib/
xdb.jar;%POST_CLASSPATH%
```

TESTING YOUR IMPLEMENTATION

These test client programs are provided (including the source code):

Program	Use the	Download from:
jaxws-client	JAX-WS client program to submit a client request to DWS through the Dispatch interface or a Service proxy.	http://IpAddress:Port/DWS/download-examples/jaxws-client.zip
wcf-client	.NET WCF client program to submit a client request through the Dispatch interface or a Service proxy.	http://IpAddress:Port/DWS/download-examples/wcf-client.zip
DWS-JSPClient	JAX-WS JSP client WAR file with sample request types for publishing and composition operations.	http://IpAddress:Port/DWS/downloadexamples/DWSJSPClient.war

Note You must deploy the DWS.ear or DWS.war file before you download the client programs. Replace *IpAddress* and *Port* with the ones for your container.

USING THE JAX-WS CLIENT PROGRAM

You can use the JAX-WS client program to test the Dispatch and Proxy interfaces as well as MTOM, WS-RM, and WS-Addressing.

Follow these steps to set up the JAX-WS client program:

1. Download the JAX-WS framework from this web site:

   `https://jax-ws.dev.java.net/ri-download.html`
2. Copy the jar files to the lib directory.
3. Download the log4j-1.2.15.jar file from this web site:

   `http://logging.apache.org/log4j/1.2/index.html`
4. Copy the jar file to the lib directory.
5. Extract the Documaker-Util.jar and Documaker-Schema.jar files from the WEB-INF/lib directory of DWS.war file and copy them to the lib directory.

Note Modify the *.bat files with your paths. Make sure you use Java version1.6.

JAX-WS Dispatch Interface

The JAX-WS Dispatch interface supports these options:

Option	Description

url	The service endpoint. Here are some examples:
	`http://localhost:8080/DWS/PublishingService?wsdl`
	`http://localhost:8080/DWS/CompositionService?wsdl`

Option	Description
file	An XML file that contains the payload. The default values are composition-request.xml and publishing-request.xml.
replyuri	A reply URI when testing WS-Addressing. Here is an example: `http://localhost:8080/DWS/echo.jsp`
threads	How many threads should invoke the service with the same payload. Here is an example: `threads=1`

The following JAX-WS Dispatch interface scripts are provided for convenience: dispatch-client.bat composition-dispatch-client.bat publishing-dispatch-client.bat

Note You can invoke dispatch-client.bat with a /? to see the usage information.

JAX-WS Service Proxy

The Service Proxy supports these options:

Option	Description

url	The service endpoint. Here are some examples:
	`url=http://localhost:8080/DWS/CompositionService?wsdl`
	`url=http://localhost:8080/DWS/PublishingService?wsdl`

operation	The name of the CompositionService or PublishingService web service operation to invoke. The default is doCallIDS.
threads	How many threads should invoke the service with the same payload. Here is an example: `threads=1`
mtom	A flag that indicates if the Message Transfer Optimization Mechanism (MTOM) should be used for the payload and any attachments. Here is an example: `mtom=Yes`
validate	A flag that indicates if schema validation should be performed at the client side before sending the request. Here is an example: `validate=Yes`
validatevalue	The value to use for the validation test when the validate flag is set to Yes. Here is an example: `validatevalue=2.0`
fastinfoset	A flag that indicates if fastinfoset should be used for the message transfer. Here is an example: `fastinfoset=Yes`
compression	A flag that indicates if http compression should be used for the message transfer. Here is an example: `compression=Yes`

*=option specific to CompositionService-doCallIDS operation.
+=option specific to PublishingService-doPublishFromImport operation.

Option	Description
streaming	A flag that indicates if MTOM attachments should be streamed. Here is an example: `streaming=Yes`
addressing	A flag that indicates if WS-Addressing should be used. Here is an example: `addressing=Yes`
replyuri	The reply URI the service should send the response message to when the addressing flag is enabled. Here is an example: `replyuri=http://localhost:8080/DWS/echo.jsp`
timeout	How long (in milliseconds) should the client wait for a response from the service. Here is an example which results in a 30 second timeout interval: `timeout=30000`
rm	A flag that indicates if WS-RM should be used. Here is an example: `rm=Yes` Configure the Binding in the WSDL to use a WS-RM policy before you use this option.
file1 *	A file to send as an attachment. Here is an example: `file1=test.xml`
file2 *	A second file to send as an attachment. Here is an example `file2=test.pdf`
async *	A flag that indicates if the client call should be asynchronous and wait for a callback. Here is an example: `async=Yes`
oneway *	A flag that indicates if the service should be invoked as a one-way (fire-and-forget) operation. Here is an example: `oneway=Yes`
ini *	The name of a file that contains the name value pairs to send to Docupresentment. Here is an example: `ini=test.ini`
file +	A file to send as an attachment. Here is an example: `file=test.xml`

*=option specific to CompositionService-doCallIDS operation.
+=option specific to PublishingService-doPublishFromImport operation.

These JAX-WS Service Proxy scripts are provided:

- service-client.bat
- composition-service-client.bat
- publishing-service-client.bat

Note You can invoke service-client.bat with a /? to see the usage information.

USING THE WCF CLIENT PROGRAM

You can use the WCF client program to test the Dispatch and Proxy interfaces as well as MTOM, WS-RM, and WS-Addressing.

Make sure you are using .NET Framework version 3.5 or later. Also make sure the ServiceClient.exe.config file is configured with the correct endpoints for your container and DWS web service when using the Service Proxy.

WCF Dispatch Interface

The WCF Dispatch interface supports these options:

Option	Description

url	Indicates the service endpoint. Here are some examples:
	`http://localhost:8080/DWS/PublishingService?wsdl`
	`http://localhost:8080/DWS/CompositionService?wsdl`

operation	Indicates the service operation to invoke. The default is doCallIDS. Here is an example: `operation=doCallIDS`
file	An XML file that contains the payload. The default values are composition-request.xml and publishing-request.xml.
threads	Indicates how many threads should invoke the service with the same payload. Here is an example: `threads=1`
rm	Indicates if WS-RM should be used. You can enter Yes or No. The default is No.
soap	Indicates whether SOAP 1.1 or SOAP 1.2 should be used. You can enter 1.1 or 1.2. The default is 1.1.

These WCF Dispatch interface scripts are provided for convenience: composition-dispatch-client.bat publishing-dispatch-client.bat

Note You can invoke dispatchclient.exe with a /? to see the usage information.

WCF Service Proxy

The WCF Service proxy supports these options:

Option	Description
service	The service name. Acceptable values are CompositionService or PublishingService.
operation	The service operation to invoke. The default is doCallIDS. Here is an example: `operation=doCallIDS`
threads	How many threads should invoke the service with the same payload. Here is an example: `threads=1`

*=option specific to CompositionService-doCallIDS operation.
+=option specific to PublishingService-doPublishFromImport operation.

Option	Description
file1	The name of a file to send as an attachment. Here is an example: `file1=extrfile.dat`
file2	The name of a file to send as an attachment. Here is an example: `file2=test.xml`
rm	A boolean flag that indicates if WS-RM should be used. Enter Yes or No. The default is No.
soap	A value that indicates if SOAP 1.1 or SOAP 1.2 should be used. Acceptable values are 1.1 or 1.2. The default is 1.1.
ini *	The name of a file that contains the name value pairs to send to Docupresentment. Here is an example: `ini=test.ini`
file +	The name of a file to send as an attachment. Here is an example: `file=extrfile.dat`

*=option specific to CompositionService-doCallIDS operation.
+=option specific to PublishingService-doPublishFromImport operation.

The WCF Service proxy also uses the ServiceClient.exe.config file for configuration options that specify timeout intervals, endpoint URLs, and so on. Make sure you configure your container and service endpoints correctly.

These WCF Service proxy scripts are provided for convenience:

- composition-service-client.bat
- publishing-service-client.bat

Note You can invoke ServiceClient.exe with a /? to see the usage information.

USING THE DWS-JSPCLIENT

You can extract and deploy the DWS-JSPClient.war file to the same container hosting DWS or to a separate container. The DWS-JSPClient.war file contains an input subdirectory with sample requests for publishing or composition operations.

Once you deploy the WAR file, you can invoke this URL:

```
http://IpAddress:Port/DWS-JSPClient/dipatch.html
```

Where *IpAddress* and *Port* should be replaced with the values for your container.

You can then use the dispatch.html page to upload one of the sample request files from the input subdirectory to test DWS. You do not need to specify values for the IP Address and Port input boxes on dispath.html page if the DWS-JSPCLlient.war file is deployed in the same application server as DWS.

Appendix A

Error Messages

This appendix contains a listing of error messages you may receive.

ERROR MESSAGE LISTING

Here is a list of common error messages:

Code	Message	To resolve
10292	Error in <GENDocFactory()>: Unable to <RULXMLExtract>.	Make sure the input XML file is valid.
10701	Error in <GenDocFactory():>: Unable to FAPFlushFormset().	Contact your support representative.
10705	Error in RunSetRcpTbl(): Unable to DSDelZeroCopyForms(pRPS->CurrentFapFormsetH,<KEY1>,<KEY2>,<KEY3>	Contact your support representative.
10954	Error in RULLoadXtrRecs(): Unable to RULGetDocSetNames	Check other error messages for the specific issue. Check the DocSetNames INI control group entries.
30056	Error in <GenDocFactory>: Unable to WIPUnloadFormset for Transaction ID	Check the JDBC connect setting in the .binding file for XML form set data and the ODBC connection setting in the INI file for native form set data.
30120	Error in <Function Name> unable to load transaction record	Check the transaction ID in the ODBC connection.
30121	Error in <Function Name> Unable to load extract data from transaction record	Check other error messages for the specific issue. Check the transaction data type.
30122	Error in <Function Name> unable to load form set data from transaction record	Check other error messages for the specific issue. Check the JDBC connect setting in the .binding file for XML form set data and the ODBC connection setting in the INI file for native form set data.
30123	Error in <Function Name> unable to load image <section name>	Make sure the requested section exists.
30124	Error in <GenDocFactory> unable to update recipient records	Check other error messages for the specific issue. The failure occurred in RULUpdateRecips.
30125	Error in <PrtDocFactory> unable to initialize print environment	Check other error messages for the specific issue. The failure in occurred in RULInitPrintEnv.
30126	Error in <Function Name> unable to load global fonts	Check the XRFFile option setting in the MasterResource control group.
30127	Error in <PrtDocFactory> unable to install callback functions	Check the INI callback function values.
30128	Error in <Function Name> required global variable not found<gvm name>	Make sure the specified global variable is defined in the DFD file. The BCH_ID and BCHSTATUS should be defined in the batch table DFD file. The TRN_ID and TRNSTATUS should be defined in the TRNS table DFD file.
30129	Error in <PrtDocFactory> unable to load batch table definition file.	Check the batch table DFD INI setting in the BCHSDFD option in the WIPData control group.
30130	Error in <PrtDocFactory> unable to open batch table	Check the table/ODBC setting for the batch table. The table is defined in the BCHS option in the WIPData control group. If you omit the BCHS option, the default table name (BCHS) is used.

Code	Message	To resolve
30131	Error in <PrtDocFactory> unable to allocate batch table record	Check your system resources and contact your support representative.
30132	Error in <PrtDocFactory> unable to locate batch record for batch id<"BCH_ID">	Make sure the batch record the batch ID exists.
30133	Error in <PrtDocFactory> unable to process print batches for ID<"BCH_ID")	This indicates a failure in the RULProcessPrintBatch. Check other error messages for the specific failure.
30211	Error in PrtDocFactory> unable to close print stream for batch <"batch name">	This is typically caused by a database update failure. The specific failure should be noted in other error messages, usually from DBLIB or SQLIB.
30212	Error in RULGenDocFactory: No form for current transaction	The form set was empty after transaction processing. This indicates a form triggering issue. Make sure the AFGJOB rules include a triggering rule (if not running import) and that the triggers are set correctly for your groups, forms, and sections. Also verify the Key1 and Key2 option settings in your INI file.
30213	Error in <Function Name> unable to parse the < gendata:JLog > ColumnNames INI section.	Check the ColumnNames section for the process name: < docfactory_assembler:JLog > < docfactory_distributor:JLog > < docfactory_presenter:JLog >
30215	Document does not have a selected recipient	The transaction did not trigger a valid recipient or did not have a selected addressee recipient.
30216	Document does not have a selected recipient assigned to an output batch	During distributor processing, no recipients were assigned to output batches. Review your batching rules and criteria.
IDF0400100004	(invalid message queue error)	Messages generated by workers which indicate an invalid message queue, such as error IDF0400100004 for the Identifier, can occur if a queue is unavailable or was configured incorrectly. This can occur when WebLogic is unavailable during a stop and restart process while the ODDF service (or Document Factory service) is still running. Once the queues are available, Document Factory should function as expected.
SUP0402300001	Failed to remove <your path> docfactory\temp	This error occurs if another application or process has locked the temp directory. It can be caused by file editors holding a lock on a file in the temp directory or if a docfactory_*.exe instance was left running after the Supervisor shut down. You should check for possible causes but this error should not prevent Documaker Factory from restarting and functioning as expected.

IDS-related errors

Code	Message	To resolve, do this
IDS_10002	`IDS connection error`	Review the log files for more information.
IDS_20002	`Misconfigured appContextBean`	Look at the log files for details on the specific request that failed. If you are trying to add an item to the TRNS table, make sure IDS can connect to the database using the 'dbpool' connection settings and is sending the appropriate items needed for the request to the Java rules.
IDS_30002	`Bad data in Inbox table`	Review the log files for more information.
IDS_40002	`Attached file ## could not be loaded`	Make sure the file exists.
IDS_50002	`No default locale set`	Specify the default locale.
IDS_60002	`Validation error`	This validation error maps to the BPEL rule validation error codes such as Primary Addressee Required and so on.
IDS_80002	`Generic exception occurred in IDS`	See the LOGS table in the Assembly Line schema for the associated <Transaction Id> entry (in the LOGTAG_ID and the LOGMESSAGE columns). Using the LOGMESSAGE column, see the Docupresentment error codes for information about the entry.

Appendix B

Migrating to Document Factory

This appendix outlines how to migrate a Documaker 11.x master resource library (MRL) to Documaker Document Factory. This includes configuring your implementation to process the MRL in a Document Factory environment.

This appendix includes these topics:

OVERVIEW

Migrating a Documaker 11.x master resource library (MRL) to Document Factory involves these steps:

- Migrate the existing environment to Documaker 12.0 Documaker Server processing. This will help confirm that you have successfully updated the MRL to an ODBC-compliant database and will serve as a baseline for validating mapping and triggering changes that may be needed. This step is not required but will help to give a level of comfort with the upgrade process.
- Update a copy of the configuration files to support running in a Document Factory environment. The details about the changes you need to make are included in this appendix. You can also use this information when creating new MRLs that run in a Document Factory environment.

This table provides information about the resources that can be shared or re-used between Documaker Server and Document Factory implementations:

Resource	Reusable?	Comments
Content stored in the MRL and user tables (DBF and MDX)	Yes	Tables should be migrated to TLK files within Studio. See the Documaker Studio User Guide for more information.
FSISYS.INI file	Yes	Will need to add portions as outlined in the following topics.
FSIUSER.INI file	No	Migrate content from the FSIUSER.INI to FSISYS.INI and then use FSIUSER_1.INI, FSIUSER_2.INI, and FSIUSER_3.INI for processing.
AFGJOB.JDT file	No	Use the provided AFGJOB_1.JDT, AFGJOB_2.JDT, and AFGJOB_3.JDT files. Custom rules are not supported.
WIP	No	The WIP content must be cleared and will be redefined to follow the TRNS structure.
Input Files	Yes	(Optional) If you want to associate individual users or groups with transactions, add this information to the input file.

Note As noted, the pre-Documaker version 12.0 WIP and archive structure has been updated in Documaker version 12.0. You should process all documents in WIP *before* you migrate those documents. You can expect to reuse MRL content in Document Factory processing.

PREPARING YOUR MRL

Documaker version 12.0 requires the MRL be stored in an ODBC-compliant database tables. If you are using Documaker Enterprise Edition with Document Factory and Documaker Interactive processing, these tables must also reflect the Assembly Line schema. The Documaker Enterprise installation creates this schema and the needed tables, so you should perform these steps *after* you install Documaker Enterprise.

1. Confirm the DMRES* tables for library resources are in the Assembly Line schema. You have these options when adding the MRL to the Assembly Line schema:

   - Remove the existing sample resources and use the tables clean. This will mean you no longer have access to the sample forms and resources from the Correspondence example provided with the installation unless you re-run the DEPLOY_SAMPLE_MRL.BAT script.
   - Share the installed tables with your new resources (in this case, be sure they use different resource names, particularly BDF names and that you set the correct BDF name reference in the FSISYS.INI file).
   - If the DBA allows both sets of resources to be retained, create a set of tables with the same structure but different names to house the new library. The original set of tables was created in the installation process by running DMKR_ASLINE.SQL against the database.

   The preferred approach is the last option, where you create tables with the same structure, but with different names.

   To create a new set of tables, create a copy of the DMKR_ASLINE.SQL file for the needed tables and rename as necessary.

   This is the database repository where the MRL content is stored and accessed for Documaker Studio, Documaker Server, and Document Factory processing. If you update the table names, you will need to modify the ODBC_FileConvert control group in the FSISYS.INI file with the updated names. Here is the information from the DMRK_ASLINE.SQL file that you would need to modify, assuming you want to leave the DMRES_DMUSER table in place.

```
CREATE TABLE "DMKR_ASLINE"."DMRES_LBYI" ("FILETYPE" CHAR(3),
"FILESTYP" CHAR(3),"FILENAME" CHAR(100),"RESOURCE"
CHAR(25),"DESCRIPT" CHAR(100),"EFFECTIV" CHAR(10),"MODIFYTM"
CHAR(10),"FILEINDX" CHAR(8),
"RECSTAT" CHAR(3),"VERSION" CHAR(5),"REVISION" CHAR(5),"USERID"
CHAR(64), "USRLEVL" CHAR(2),"PASSWD" CHAR(64),"UNIQUE_ID"
CHAR(26),"ARCKEY" CHAR(18), "MODE" CHAR(25),"STATUS" CHAR(25),"CLASS"
CHAR(25),"PROJECT" CHAR(25) );
CREATE INDEX "DMKR_ASLINE"."DMRES_LBFILEINDX" ON "DMRES_LBYI" (
"FILETYPE", "FILESTYP", "FILENAME", "VERSION", "REVISION" );
CREATE INDEX "DMKR_ASLINE"."DMRES_LBUNIQUE_ID" ON "DMRES_LBYI" (
"UNIQUE_ID" );

CREATE TABLE "DMKR_ASLINE"."DMRES_LBYD" ( "ARCKEY" CHAR(18) NOT
NULL,"SEQ_NUM" CHAR(5) NOT NULL,"CONT_FLAG" CHAR(1),"TOTAL_SIZE"
NUMBER(38),"CARDATA" BLOB, CONSTRAINT "DMRES_LBARCKEY" UNIQUE(
"ARCKEY", "SEQ_NUM" ) );
```

```
CREATE INDEX "DMKR_ASLINE"."DMRES_LBSEQ_NUM" ON "DMRES_LBYD" (
"SEQ_NUM" );
CREATE INDEX "DMKR_ASLINE"."DMRES_LBCAR_KEY" ON "DMRES_LBYD" (
"ARCKEY" );

CREATE TABLE "DMKR_ASLINE"."DMRES_LBYC" ( "CATALOGID"
CHAR(10),"CARFILE" CHAR(8),"MEDIAID" CHAR(11),"LASTNUM"
CHAR(8),"STATUS" CHAR(1) );
CREATE INDEX "DMKR_ASLINE"."DMRES_DECATALOGKEY" ON "DMRES_LBYC" (
"CATALOGID" );
CREATE INDEX "DMKR_ASLINE"."DMRES_DECARFILEKEY" ON "DMRES_LBYC" (
"CARFILE" );
CREATE INDEX "DMKR_ASLINE"."DMRES_DELASTNUMKEY" ON "DMRES_LBYC" (
"LASTNUM" );

CREATE TABLE "DMKR_ASLINE"."DMRES_LBYL" ( "DATE" CHAR(8),"TIME"
CHAR(10),"LIBNAME" CHAR(129),"ACTION" CHAR(20),"FILENAME"
CHAR(100),"FILETYPE" CHAR(3),"VERSION" CHAR(5),"REVISION"
CHAR(5),"EFFECTIV" CHAR(10),"MODE" CHAR(25),"STATUS"
CHAR(25),"CLASS" CHAR(25),"PROJECT" CHAR(25),"USERID"
CHAR(64),"PROCESS" CHAR(20) );
CREATE INDEX "DMKR_ASLINE"."DMRES_DELOGTAG" ON "DMRES_LBYL" ( "DATE",
"TIME" );
CREATE INDEX "DMKR_ASLINE"."DMRES_DEUNIQTAG" ON "DMRES_LBYL" (
"DATE", "TIME", "LIBNAME", "ACTION", "FILENAME", "FILETYPE",
"VERSION", "REVISION", "EFFECTIV" );
```

2. If you are using a Studio installation outside of the Document Factory installation application server to update and modify the resources in the Assembly Line, you must also establish an ODBC connection from the Studio computer to the DMKR_ASLINE schema and make sure appropriate permissions to the table are in place.
3. Use Documaker Studio version 12.0 to access, update, and promote resources from the existing version MRL or create new MRL resources. You can either start from scratch by accessing these tables or promote an existing set of resources into the Assembly Line tables. The promotion can be completed through Studio or by using the DEPLOY_SAMPLE_RESOURCE.BAT file on the Document Factory installation application server.

Note Using the DEPLOY_SAMPLE_RESOURCES.BAT file assumes that you have a pre 12.0 MRL in xBase format called *master.lby* in the dmres\deflib directory on the installed application server. If you want to change the source for the promotion process, be sure to configure the INI files as needed.

CONFIGURING THE RUNTIME ENVIRONMENT

After you install Documaker Enterprise, you can modify the following files to configure the runtime environment.

The sample INI files are based on a standard directory layout for the Documaker resources. This layout is explained in the Documaker Enterprise Installation Guide.

Note If your directory structure does not follow this standard layout, modify it now to make implementation easier.

Here is an overview of the new and updated files:

File type	File name	Comments
INI files		References a specific AFGJOB.JDT file which contains processing rules and information for Document Factory processing.
	FSISYS.INI	
	FSIUSER_1.INI	Used by the Assembler.
	FSIUSER_2.INI	Used by the Distributor.
	FSIUSER_3.INI	Used by the Presenter.
	FSIUSER.INI	Applicable if using Documaker Interactive.
DFD files		
	BCHS.DFD	Defines the layout of the BCHS table in the Assembly Line. Do not modify.
	BCHS_RCPS.DFD	The BCHS_RCPS table.
	CARFILEORA.DFD	Defines the layout of the transaction history form set data. Do not modify.
	DOCDATA.DFD	Define the layout of the document data during WIP processing. Do not modify.
	DSDATA.DFD	Defines the layout of the form set data during document processing. Do not modify.
	JOBS.DFD	The JOBS table definition. Do not modify.
	PUBS.DFD	The PUBS table definition. Do not modify.
	PUBSINFO.DFD	The publication data layout. Do not modify.
	RCBDOCF.DFD	The RCPS table definition. Do not modify.
	RCPSPRT.DFD	The layout of the RCPSPRT table. Do not modify.
	TRNDFDFL.DFD	The layout of the TRNS record if it is running outside of Document Factory. Do not modify.
	TRNSDF.DFD	The layout of the TRNS table in the assembly line schema. Do not modify.

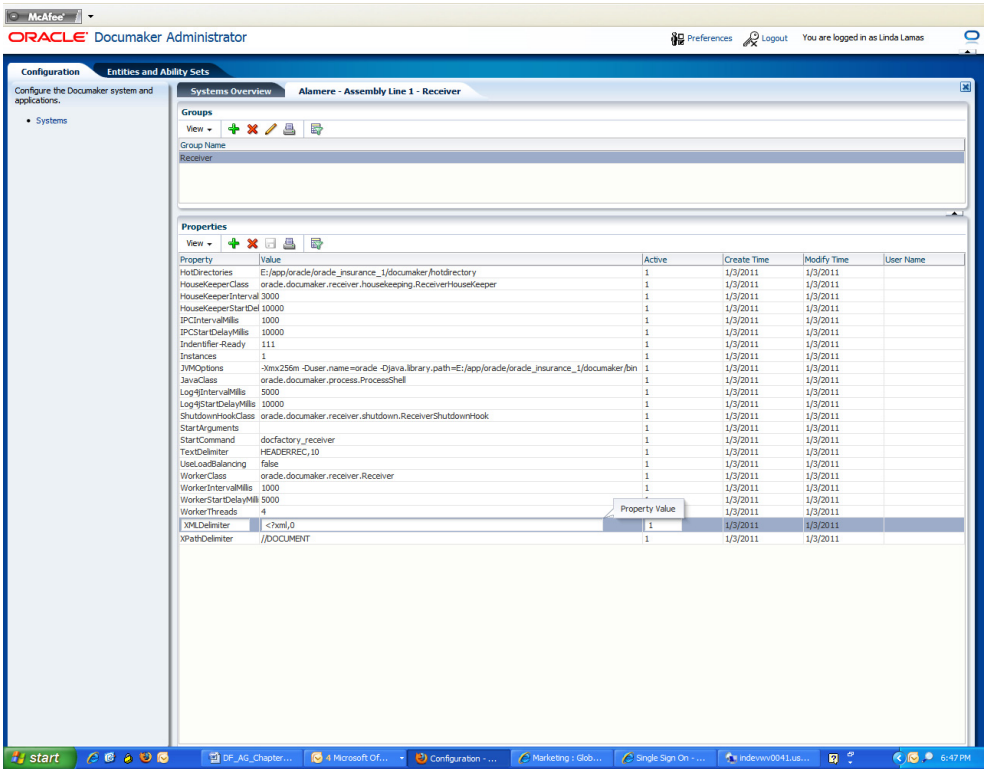
File type	File name	Comments
JDT files		
	AFGJOB_1.JDT	Contains the rules processed by the Assembler to trigger forms and map data.
	AFGJOB_2.JDT	Contains the rules processed by the Distributor to identify form recipients.
	AFGJOB_3.JDT	Contains the rules processed by the Presenter to create print streams.
	Note: You can add rules to these files, but do so with care.	

FSISYS.INI File

Oracle recommends using the reference implementation INI files installed with Documaker Enterprise as a starting point for creating your Document Factory processing environment. If you use the sample FSISYS.INI file, here are the control groups you may want to update to meet you implementation's needs. Be sure to also check the pathing.

Group and Option	Current Value	Use
< AutoFields >	-	Populate with the fields and values you want to set into the forms data in Documaker Interactive.
< DefaultFields >	-	Populate with the fields and values you want to default into the forms data in Documaker Interactive.
< BatchingByRecip >	BATCH1	The FSISYS.INI sample is configured to have one print batch (Batch1), an error batch, and a manual batch. If you want Document Factory to define additional batch groups, define them in this control group.
< Configurations >	Correspondence	Specify the name you want to assign to the MRL Check the value of the StandardType option in the DocFactory control group in the FSIUSER_2.INI file. This value should match the batch type defined in the BATCHINGS configuration for Doucment Factory processing. The default value for the reference implementation is ADDRESSEE.
< Config:*Correspondence* >	Correspondence	Set to Config:*MRL* name. Use DMRes for these options: BDFFile, DALFile, DDTFile, FORFile, FormFile, GRPFile, LogoFile, and XDDFile. Make sure the BaseDef option points to the BDF file name you want to use. Make sure the XRFFile option points to the name of the FXR file you want to use.
< ExtractKeyField > SearchMask	1,<?xml	Set this to be the delimiter to define each transaction in a stacked XML file. Also update the APPCONFIGCONTEXT property, XMLDelimiter (or XMLPathDelimiter or TextDelimiter), within the Receiver control group. Use Documaker Administrator under the assembly line's Receiver worker's configuration settings to make these changes.
< MasterResource >	Correspondence	Set to the < Config:*MRL* > name you want to use.
< PrtType: >	Includes all print types Document Factory supports	If you want to add other, custom print types, add them here along with the appropriate configuration information.

Group and Option	Current Value	Use
< RunMode >	XMLExtract = Yes	Set to No if you are processing flat file extract data.
< TRN_FIELDS >	10 GVM values mapped to sample XML based input data.	Use these but update with the correct mapping information. If using any of the TRNCUS* fields defined in the TRNS table (TRNSDF.DFD), add to the provided values. Also be sure to uncomment the needed fields from the TRNSDF.DFD if you are using any of the TRNCUS* fields to map implementation specific GVM values.



The installed FSISYS.INI file contains all of the Document Factory processing configuration you need, so modifying the items listed above is the recommended approach. If, however, you prefer to use your existing FSISYS.INI files, here are the Document Factory processing settings you must add.

Note These settings came from the RPEX2 resources so you may already have some of these items in your existing configuration.

1. Add a reference to make sure the MRL can recognize the configuration stored in the database when processing.

```
< DocFactory >
   bindings = \oracle\oracle_insurance_1\documaker\docfactory
\config\context
```

2. Make sure Document Factory processing is enabled by setting the DocFactory option to Yes in the RunMode control group.

```
< RunMode >
   DocFactory = Yes
```

3. Add the PrtType:PDF control group to define the default output type for Document Factory processing.

```
< PrtType:PDF >
   Class = PDF
   DownloadFonts = N,Disabled
   Module = PDFOS2
   OverlayExt = .ovl
   PageNumbers = Yes
   PaperSize = 0
   PrintFunc = PDFPrint
   PrintViewOnly = No
   SendColor = Yes
   SendOverlays = No
   SplitPercent = 50
   SplitText = No
```

4. Replace the Trigger2WIP INI group with the one shown here to match the layout of the TRNS:

```
< Trigger2WIP >
   Key1 = KEY1
   Key2 = KEY2
   KeyID = KEYID
   Desc = DESC
   CurrGroup = CURRGROUP
   ApprovalState = APPROVALSTATE
   Action = ACTION
   TRNName = TRNNAME
```

   Add any other GVMs you want mapped to the TRNS table values here. Be sure to uncomment the TRNCUS* field references in the TRNSDF.DFD and TRNSIDS.DFD files if needed.

5. Add the AFG2WIP control group to allow a user to be associated with each individual transaction.

```
< AFG2WIP >
   UserID = ~GVM ORIGUSER
```

6. Replace the TRN_FIELDS group as shown here to create the needed GVMs, including the OrigUser referenced by the AFG2WIP file for each transaction. The current sample TRN_FIELDS section contains these options:

```
< TRN_Fields >
   Key1          = !/location of value in extract
   Key2          = !/location of value in extract
   KeyID         = !/location of value in extract
   TRNName       = !/location of value in extract
   CurrGroup     = !/location of value in extract
   OrigUser      = !/location of value in extract
   Desc          = !/location of value in extract
   ApprovalState= !/location of value in extract
   Action        = !/location of value in extract
```

   Here is a brief description of the options of particular interest:

Option	Description
OrigUser	Used to track documents by user and group, and particularly needed for Documaker Interactive processing.

Option	Description
ApprovalState	Used by Documaker Interactive processing to identify the state of the document within the processing flow.
Action	Specifies an initial action taken by the document on input into the system. The default is Batch Created. You can customize this for your implementation to more specifically define the activity or source of a given transaction.

7. Set the DFD values in the base DFD files as needed. Keep in mind that these will be overwritten for Document Factory processing by references in the FSIUSER_*.INI files.

```
< Data>
    RcbDfdFile = rcbdfdfl.dfd
    TrnDfdFile = trndfdfl.dfd
```

Note Documaker Server and Document Factory processing both rely on the TRNDFDFL.DFD and the RCBDFDFL.DFD files for properly defining and creating internal GVMs. These files, however, are not used to define the layout of either the TRNS or RCPS tables. So, keep these files referenced and available but note that only any common fields between TRNDFDFL.DFD and RCBDFDFL.DFD and TRNS.DFD and RCBDOCF.DFD respectively will be retained as GVM data. Also, do not modify the DFD files or table layouts from the sample MRL – these are configured to be used in specific implementations.

8. Update the DBHandler and DBTables settings as shown to establish the ODBC to Document Factory tables:

```
< DBHandler:ODBC_DMKR_ASLINE >
    Class = ODBC
    Description = Oracle ODBC Assembly Line
    Passwd = ********
    Server = dmkr_asline
    UserID = dmkr_asline
    CreateTable = No
    CreateIndex = No
    Debug = No
```

These DBTable entries are used to define the name and ODBC handler to use to access the resource library tables within the Assembly Line schema. For more information on these tables, see the Documaker Studio User Guide.

```
< Library:DMRES >
    DBTable = DMRESD
    CATALOG = DMRESC
    Description = sample resources
    LBYLogFile = DMRESL
    USERFile = DMRES_DMUSER
< DBTable:DMRES >
    DBHandler = ODBC_DMKR_ASLINE
< DBTable:DMRESC >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = CATALOGID
< DBTable:DMRESD >
    DBHandler = ODBC_DMKR_ASLINE
    DFD = \oracle\oracle_insurance_1\documaker\mstrres\deflib
\carfileora.dfd
    UniqueTag = ARCKEY+SEQ_NUM
< DBTable:DMRESL >
    DBHandler = ODBC_DMKR_ASLINE
```

```
    UniqueTag = DATE+TIME
< DBTable:DMRES_DMUSER >
    DBHandler = ODBC_DMKR_ASLINE
    DefaultTag = UNIQUEIDTAG
    UniqueIDTag = UNIQUEIDTAG
    UniqueTag = IDTAG
< USERINFO >
    FILE = DMRES_DMUSER
```

These table entries are for the Document Factory processing tables.

The extract file table (EXTR) is used by the Assembler to identify that the extract data for the transaction is contained in the database table identified by the DBHandler, in this case the TRNS table:

```
< DBTable:EXTR >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = TRN_ID
```

The Jobs table is the initial processing table for Document Factory.

```
< DBTable:JOBS >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = JOBUNIQUE_ID
```

The transaction status table (TRNSTATUS) is used by the Assembler, Batcher, and Distributor to determine the location of the status information to update for each transaction processed.

```
< DBTable:TRNSTATUS >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = TRN_ID
```

The WIP and WIPData (NA and POL file data) tables – in version 12.0 and Document Factory processing the WIP data – NA and POL file information is stored in the TRNS table along with the transaction key information. In Document Factory, this data is stored in XML format.

```
< DBTable:WIP >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = FORMSETID
< DBTable:WIPData >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = FORMSETID
```

The RCPS group defines the recipients for each transaction.

```
< DBTable:RCPS >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = RCP_ID
```

The BCHS group defines the active batches for document processing.

```
< DBTable:BCHS >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = BCH_ID
```

The PUBS and PUBSINFO groups define the publications (print streams) for each batch.

```
< DBTable:PUBS >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = PUBUNIQUE_ID
< DBTable:PUBSINFO >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = PUBUNIQUE_ID
```

The BCHS_RCPS, BCH_RCPS_UPD, and RCBSPRT groups provide linking information to reference the recipients with each unique batch and each unique printed output.

```
< DBTable:BCHS_RCPS >
    DBHandler = ODBC_DMKR_ASLINE
< DBTable:BCHS_RCPS_UPD >
    DBHandler = ODBC_DMKR_ASLINE
< DBTable:RCBSPRT >
    DBHandler = ODBC_DMKR_ASLINE
    UniqueTag = RCP_ID
```

The ODBC_FileConvert group converts the logical table name into a physical table.

```
< ODBC_FileConvert >
    WIP = TRNS
    WIPData = TRNS
    EXTR = TRNS
    TRNStatus = TRNS
    RCBSPRT = RCPS
    PUBSInfo = PUBS
    BCHS_RCPS_UPD = BCHS_RCPS
    DMRes = DMRES_LBYI
    DMResC = DMRES_LBYC
    DMResD = DMRES_LBYD
    DMResL = DMRES_LBYL
    DMRes_DMUser = DMRES_DMUSER
< ODBC_FieldConvert >
    Desc = DESCR
```

9. Make sure the WIPData group contains the required WIP data entries:

```
< WIPData >
    DatabaseWIP = Yes
    DocFactory = Yes
    File = WIP
    Path = <CONFIG:CORRESPONDENCE> WIPPath =
    Jobs = JOBS
    JobsDFD = \oracle\oracle_insurance_1\documaker\mstrres\dmres
\deflib\jobs.dfd
    BCHS = BCHS
    BCHSDFD = \oracle\oracle_insurance_1\documaker\mstrres\dmres
\deflib\BCHS.dfd
    WIPDFDFile = .\deflib\trnsdf.dfd
    WIPDataDFD = .\deflib\docdata.dfd
    WIPDsDataDFD = .\deflib\dsdata.dfd
```

*Where *Correspondence* is the name of the MRL you are updating.

The WIPDFDFile defines the layout for the WIP or TRNS table. Do not modify this entry. The WIPDataDFD defines the layout of the WIP content if stored as an XML data type in the TRNS table. WIPDSDataDFD defines the layout of the WIP content if stored as combined NA/POL data in BLOB format in the TRNS table. The default for NA/POL data is XML.

Note Batching within Document Factory is a two-step process, expanding the grouping options available to you. The first step in the process is to identify a batching group – via rules within the AFGJOB_2.JDT and the FSISYS.INI file as you would for typical Documaker Server processing. To centralize configuration, you should put all documents into one batch within the FSISYS.INI file and handle the batching logic entirely in the Document Factory.

The Distributor process refines the FSISYS.INI defined batches, or batch groupings, based on the rules and options set in the BCHINGS table, and is controlled via the Documaker Administrator.

FSIUSER_1.INI File

The FSIUSER_1.INI file is used during the Assembler process, as defined by the APPCONFIGCONTEXT StartArguments property for the Assembler.

If you are migrating from a previous implementation, copy the contents of the existing FSIUSER.INI files and paste them into the FSISYS.INI file. The sample installed FSIUSER_1.INI file should be used or you can create a new one from the content listed here.

```
< Configurations >
   Config = Your config name
< Environment >
   FSISYSINI = FSISYS.INI
   FSITEMP = temp
   JLOG_Enabled = No
```

In the Data control group, the AFGJobFile option points to the Assembler AFGJOB_1.JDT file and the recipient record layout used matches the RCPS table.

```
< Data >
   AFGJobFile = afgjob_1.jdt
   RCBDFDFile = rcbdocf.dfd
```

These control groups are required to log error messages to the Document Factory tables:

```
< docfactory_assembler:JLog >
   LogLogger = LogLogger
   ErrorLogger = ErrorLogger
   ColumnNames =
JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_
ID,PUB_ID=DF_PUB_ID
;  BufferSize = 2000
   Debug = No
   LogWarning = No
   LogError = Yes
< GenData:JLog >
   LogLogger = LogLogger
   ErrorLogger = ErrorLogger
   ColumnNames =
JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_
ID,PUB_ID=DF_PUB_ID
;  BufferSize = 2000
   Debug = No
   LogWarning = No
   LogError = Yes
```

FSIUSER_2.INI File

This file is included in the sample Correspondence MRL. The FSIUSER_2.INI file is used during the Distributor process – as defined by the APPCONFIGCONTEXT StartArguments property for the Distributor.

If you are migrating from a previous implementation, copy the contents of the existing FSIUSER.INI files and paste that content into the FSISYS.INI file. You can use the sample FSIUSER_2.INI file or you can create a new one from the content listed here.

The FSIUSER_2.INI file includes these settings:

```
< Configurations >
    Config = Your config name
< Environment >
    FSISYSINI = FSISYS.INI
    FSITemp = temp
    JLOG_Enabled = Yes
< DocFactory >
    StandardType = ADDRESSEE
< Data >
    AFGJobFile = afgjob_2.jdt
    RCBDFDFile = rcbdocf.dfd
< docfactory_distributor:JLog >
    LogLogger = LogLogger
    ErrorLogger = ErrorLogger
    ColumnNames =
JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_
ID,PUB_ID=DF_PUB_ID
;   BufferSize = 2000
    Debug = No
    LogWarning = No
    LogError = Yes
< gendata:JLog >
    LogLogger = LogLogger
    ErrorLogger = ErrorLogger
    ColumnNames =
JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_
ID,PUB_ID=DF_PUB_ID
;   BufferSize = 2000
    Debug = No
    LogWarning = No
    LogError = Yes
```

FSIUSER_3.INI File

This file is included with sample Correspondence MRL. The FSIUSER_3.INI file is used during the Presenter process, as defined by the APPCONFIGCONTEXT Arguments property in the Presenter configuration.

```
< Configurations >
    Config = Your config name
< Environment >
    FSISYSINI = FSISYS.INI
    FSITEMP = temp
    JLOG_Enabled = Yes
< RULImagePrintName >
    Font = 10006
    Red = 256
    Green = 0
    Blue = 0
< RunMode >
```

```
    LoadFAPBitmap = Yes
< Data >
    AfgJobFile = afgjob_3.jdt
    RcbDfdFile = rcbdocf.dfd
< docfactory_presenter:JLog >
    LogLogger = LogLogger
    ErrorLogger = ErrorLogger
    ColumnNames =
JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_
ID,PUB_ID=DF_PUB_ID
;   BufferSize = 2000
    Debug = No
    LogWarning = No
    LogError = Yes
< gendata:JLog >
    LogLogger = LogLogger
    ErrorLogger = ErrorLogger
    ColumnNames =
JOB_ID=DF_JOB_ID,TRN_ID=DF_TRAN_ID,BCH_ID=DF_BATCH_ID,RCP_ID=DF_RCP_
ID,PUB_ID=DF_PUB_ID
;   BufferSize = 2000
    Debug = No
    LogWarning = No
    LogError = Yes
```

AFGJOB_1.JDT File

The AFGJOB_1.JDT file is used in the Assembler phase. The Assembler phase performs the function of Documaker Server's GenData program. It triggers forms and maps data onto those forms. It is also responsible for updating the transaction's key values as defined by the TRN_FIELDS in the FSISYS.INI file.

The Assembler phase, however, does not write the recipient records. That job is performed by the Distributor. Each of the AFGJOB files provided should be uses as is within Document Factory. Additional rules or custom modules may not be supported.

The AFGJOB_1.JDT file is similar to a Documaker Server AFGJOB.JDT file except the RULStandardTransactionProc and LoadExtractData rules are replaced by the GenDocFactory rule.

```
<Base Form Set Rules>
/*;UnitTestDocFactory is only used for testing of individual
transactions*/
/*;UnitTestDocFactory;2;TRN_ID=151;
;GenDocFactory;2;DocFactory Phase 1;
```

Note The SetOvFlwSym entries in the AFGJOB_1.JDT file work with the sample MRL provided. They are not necessary for processing documents in other MRLs.

AFGJOB_2.JDT File

The AFGJOB_2.JDT file is used in the Distributor phase. The Distributor phase creates the recipient batch table records (RCPS) or, in the case of running under Documaker Server, the recipient records are written out to the BCH files as defined in the FSISYS.INI file. This Document Factory version of the AFGJOB_2.JDT file includes the batch assignment rules and the new RcpDocFactory rule.

Note Record the BCH file names and understand the logic defined in the FSISYS.INI file as you will want to have this material available when configuring the final output batches via the Documaker Administrator.

```
/* Every form set in this base uses these rules. */
<Base Form Set Rules>
;RcpDocFactory;2;DocFactory Phase 2;

/* Every section in this base uses these rules. */
<Base Image Rules>
;WIPImageProc;;;

/* Every field in this base uses these rules. */
<Base Field Rules>
;WIPFieldProc;;;
```

AFGJOB_3.JDT File

The AFGJOB_3.JDT file is used in the Presenter phase. The Presenter performs the same function as Documaker Server's GenPrint program. The Presenter uses the PrtDocFactory rule.

Since printing is combined with the AFGJOB_2.JDT file in Documaker Server processing, there is no version of this file for testing. Use the GenData program with the AFGJOB_2.JDT file or use the GenPrint program when testing Documaker Server equivalent of Document Factory.

```
/* Every form set in this base uses these rules. */
<Base Form Set Rules>
;PrtDocFactory;2;DocFactory Phase 3;

/* Every image in this base uses these rules. */
<Base Image Rules>
;WIPImageProc;3;Always the first image level rule;

/* Every field in this base uses these rules. */
<Base Field Rules>
;WIPImageProc;4;Always the first field level rule;
```

CONFIGURING DOCUMAKER INTERACTIVE

At this point, Document Factory is now configured to receive JOBS submitted for processing within the updated MRL. If, however, you are using Documaker Interactive, there are a few more steps to update the new resources.

1. Configure Docupresentment to recognize the new library by CONFIG name. To do this, update the DAP.INI file with the name of the new configuration.

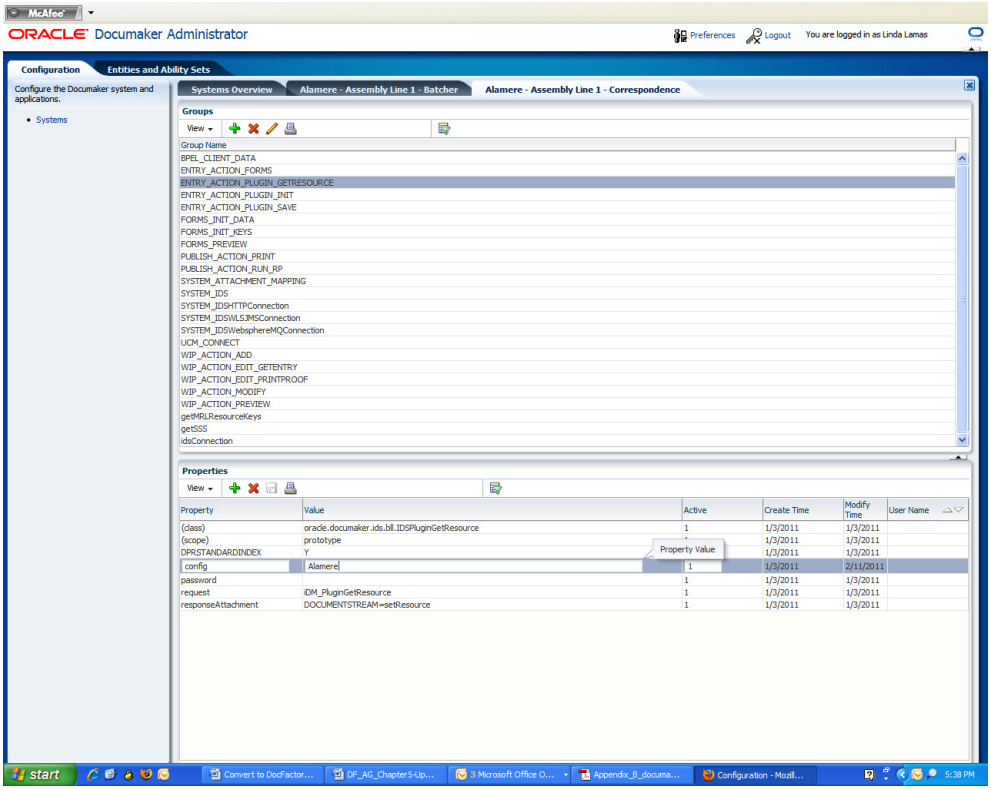
```
< Configurations >
    Config = Correspondence
```

   You can find the DAP.INI file in the Docupresentment directory.

2. Update Docupresentment to define the location of the new Config library. To do this, update the Configurations and Config control groups in the FSIUSER.INI file, just as you did in the FSISYS.INI file.

Group and Option	Current Value	Use
<Configurations>	Correspondence	Set to the name you want to call the MRL
<Config:*Correspondence*>	Correspondence	Set to Config:*MRL* name. Use DMRES for these entries: BDFFile, DALFile, DDTFile, FORFile, FormFile, GRPFile, LogoFile, XDDFile. Make sure the BaseDef points to the BDF file name you want to use. Make sure the XRFFile option points to the name of the FXR file you want to use.

3. Next, update the WIP Edit plug-in to use the new Config control group settings.



4. Use Documaker Administrator to modify the configuration properties in the following groups within the Correspondence application configuration to match the name of the new configuration:
   - ENTRY_ACTION_PLUGIN_GETRESOURCE
   - ENTRY_ACTION_PLUGIN_INIT
   - ENTRY_ACTION_PLUGIN_SAVE
   - FORMS_PREVIEW
   - PUBLISH_ACTION_RUN_RP
   - WIP_ACTION_EDIT_PRINTPROOF
5. Modify the configList and defaultConfig properties in the SYSTEM_IDS group to match the name of the new configuration.

ADDING FORMS TO THE RESOURCE LIBRARY

Using Documaker Studio, library administrators update the resources used by the Document Factory. These resources are stored in the Assembly Line schema within the library tables, by default, prefixed with *dmres*.

Documaker Interactive also uses these resources to build form sets, display documents, and editing. Documaker Interactive, however, uses its own set of index tables for searching and filtering the forms list when adding/editing a document.

The Studio promotion and update process modifies the content of the dmres tables but not of the tables referenced by Documaker Interactive for form selection.

If there are updates to the master resource library in the DMRK_ASLINE schema that you want Documaker Interactive to recognize during form selection, you must restart the idm_server, as well as the ODDF Supervisor and Docupresentment Services.

INDEX

V

W