Oracle® Universal Content Management
Services Reference Guide
10g Release 3 (10.1.3.3.0)

March 2007

ORACLE

Services Reference Guide, 10g Release 3 (10.1.3.3.0)
Copyright © 2007, Oracle. All rights reserved.

Contributing Authors: Will Harris, Jean Wilson

Contributors: Eva Cordes, Alec Kloss, Sam White

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Table of Contents

Chapter 1: Introduction

About This Guide 1-1

Organization 1-2

Audience 1-2

Conventions 1-3

Chapter 2: What's New

Overview 2-1

Revised Documentation 2-1

Changes to Services 2-1

New Services Interface 2-2

Chapter 3: Using Services

Overview 3-1

Service Overview 3-1

- Service Requests and Responses 3-2
  - Internal Service Requests 3-2
  - External Service Requests 3-2
  - Request Parameters 3-3
  - Date and Time Formatting 3-3
  - Case Sensitivity Considerations 3-4
- Page Retrieval 3-4
- Content Server Search Services 3-5
- Integration Methods 3-5
- Calling Services Using Persistent URLs 3-6
- Customizing Locale Parameters 3-8
- Forcing Authentication Challenges 3-10

Custom Application Example 3-10

Redirecting Template Page for Response Output 3-18

Basic Concepts 3-18
Creating a HCST Page 3-18
Reformatting the Search Results Page 3-19
Additional Options 3-19

Chapter 4: Customizing Services

Overview 4-1
Service Structure Overview 4-1
Name 4-2
Attributes 4-3
Service Class 4-4
Access Level 4-6
Template Page 4-8
Service Type 4-9
Subjects Notified 4-9
Error Message 4-11
Actions 4-12
Action Type 4-13
Action Name 4-14
Action Parameters 4-15
Action Control Mask 4-15
Action Error Message 4-18
Service Example 4-18
DOC_INFO Service Definition 4-19
DOC_INFO Attributes 4-20
DOC_INFO Actions 4-21
DOC_INFO Template 4-26
Creating a Service Resource 4-29
Creating a Custom Service Manually 4-29
Creating a Custom Service using Component Wizard 4-32

Chapter 5: Core Content Server Services

Overview 5-1
About Core Content Server Services 5-2
General Services (Core Content Server) 5-3
ADD_DOC_ACCOUNT 5-4
ADD_DOCEXTENSION 5-4
ADD_DOCFORMAT 5-5
ADD_DOCTYPE 5-6

APPLET_DOCINFO 5-6
CONFIG_INFO 5-7
DELETE_DOC_ACCOUNT 5-7
DELETE_DOCEXTENSION 5-7
DELETE_DOCFORMAT 5-8
DELETE_DOCTYPE 5-8
EDIT_DOCEXTENSION 5-8
EDIT_DOCFORMAT 5-9
EDIT_DOCTYPE 5-10
EDIT_TRACE_OPTIONS 5-10
GET_DATARESULTSET 5-10
GET_DOCEXTENSIONS 5-11
GET_DOCFORMATS 5-11
GET_DOCTYPES 5-11
GET_FILELIST 5-12
GET_METADEFS 5-12
GET_RESULT_OPTIONS 5-12
GET_SYSTEM_AUDIT_INFO 5-13
GET_TABLE 5-13
GET_USER_METADEFS 5-14
JAVA_PROPERTIES 5-14
LM_BUILD_WEB_STRING_FILES 5-14
LM_LOAD_LAYOUTS 5-14
LM_LOAD_LAYOUTS_SUB 5-14
LOAD_DOC_ENVIRONMENT 5-14
LOGIN 5-15
MERGE_TABLE 5-16
PING_SERVER 5-19
QUERY_DOC_ACCOUNTS 5-20
SOAP_FAULT 5-20
Doc Services (Core Content Server) 5-21
ASSIGN_DOCINFO_FORM 5-24
CACHE_CHECKIN_NEW 5-24
CACHE_CHECKIN_SEL 5-24
CACHE_SUBMIT_HTML_FORM 5-24
CACHE_WORKFLOW_CHECKIN 5-25
CHECKIN_ARCHIVE_NO_NOTIFY 5-25
CHECKIN_BYNAME 5-26
CHECKIN_CONFIRM_FORM 5-30
CHECKIN_LIST 5-30
CHECKIN_NEW 5-31
CHECKIN_NEW_FORM 5-37

CHECKIN_NEW_SUB 5-38
CHECKIN_SEL 5-40
CHECKIN_SEL_FORM 5-45
CHECKIN_SEL_SUB 5-47
CHECKIN_SIMILAR_FORM 5-47
CHECKIN_UNIVERSAL 5-48
CHECKOUT 5-52
CHECKOUT_BY_NAME 5-55
CHECKOUT_OK 5-56
CHECKOUT_SUB 5-57
CONTINUE_CHECKIN 5-59
CONTINUE_SUBMIT_HTML_FORM 5-59
CREATE_SUBSCRIPTION_TYPE 5-60
DELETE_BYCLASS 5-60
DELETE_BYNAME 5-60
DELETE_BYREV 5-61
DELETE_BYREV_REVISION 5-61
DELETE_CHECKIN_CACHE 5-61
DELETE_DOC 5-61
DELETE_REV 5-62
DELETE_REV_EX 5-63
DELETE_SUBSCRIPTION_TYPE 5-64
DOC_FORMATS_WIZARD 5-64
DOC_INFO 5-64
DOC_INFO_BY_NAME 5-68
DOC_INFO_LATESTRELEASE 5-70
DOC_INFO_SIMPLE 5-71
DOC_INFO_SIMPLE_BYREV 5-72
DOC_SUBS_LIST 5-73
EDIT_DOC_FORMATS 5-75
FORM_PROCESS 5-76
FORM_SUBMIT 5-76
GET_CACHED_CHECKIN_INFO 5-77
GET_DOC_CONFIG_INFO 5-77
GET_DOC_SUBSCRIBERS 5-78
GET_EXPIRED 5-78
GET_PACKAGE_ENVIRONMENT_PAGE 5-79
GET_UPDATE_FORM 5-80
NOTIFY_INDEXER 5-81
ODMA_DOC_INFO_SIMPLE 5-81
PACKAGE_ENVIRONMENT 5-81
REMOVE_METAFILE_SUB 5-81

REPLACE_METAFILE_SUB 5-82
RESUBMIT_FOR_CONVERSION 5-82
REV_HISTORY 5-84
SELECTDOC 5-84
SUBMIT_HTML_FORM 5-85
SUBSCRIBE 5-86
SUBSCRIBE_DOC_USER 5-88
SUBSCRIBE_EX 5-89
SUBSCRIBE_FORM 5-90
SUBSCRIPTION_LIST 5-91
UNDO_CHECKOUT 5-92
UNDO_CHECKOUT_BY_NAME 5-95
UNSUBSCRIBE 5-95
UNSUBSCRIBE_FORM 5-97
UNSUBSCRIBE_FROM_LIST 5-98
UNSUBSCRIBE_FROM_LIST_EX 5-100
UPDATE_BYREV 5-101
UPDATE_DOCINFO 5-102
UPDATE_DOCINFO_BYFORM 5-105
UPDATE_DOCINFO_BYREV 5-109
UPDATE_DOCINFO_METAFILE_BYREV 5-109
UPDATE_DOCINFO_STATUS 5-109
UPDATE_DOCINFO_SUB 5-110
UPDATE_SUBSCRIPTION_NOTIFY 5-110
UPDATE_SUBSCRIPTION_TYPE 5-110
UPDATE_SUBSCRIPTION_USED 5-111
UPDATE_METADATA 5-112
VALIDATE_DOCINFO 5-112
WORK_IN_PROGRESS 5-112
Doc Profile Services (Core Content Server) 5-113
ADD_DOCPROFILE 5-113
ADD_DOCRULE 5-114
DELETE_DOCPROFILE 5-114
DELETE_DOCRULE 5-115
DOCPROFILE_PREVIEW 5-115
EDIT_DOCPROFILE 5-115
EDIT_DOCPROFILE_TRIGGER 5-116
EDIT_DOCRULE 5-116
GET_DOCPROFILE 5-116
GET_DOCPROFILES 5-117
GET_DOCRULE 5-117
GET_DOCRULES 5-117

File Services (Core Content Server) 5-117
ADD_WEB_APP 5-118
APPEND_FILE_CACHING_INFO 5-118
GET_DYNAMIC_CONVERSION. 5-118
GET_DYNAMIC_CONVERSION_SUB 5-120
GET_DYNAMIC_URL 5-121
GET_EXTERNAL_DYNAMIC_CONVERSION 5-122
GET_TEMPLATE_CONVERSIONS 5-122
GET_WEB_APP_STATUS 5-122
LOAD_RESOURCE_FILE. 5-122
REMOVE_WEB_APP 5-124
SAVE_TEMPLATE_CONVERSIONS 5-124
Indexer Services (Core Content Server) 5-124
CANCEL_SEARCH_INDEX 5-124
CONTROL_SEARCH_INDEX. 5-125
GET_FILE 5-129
START_SEARCH_INDEX. 5-132
Internal Services (Core Content Server) 5-133
CANCEL_COMPONENT_INSTALL 5-134
CLEAR_SERVER_OUTPUT 5-134
DOWNLOAD_COMPONENT 5-134
GET_COMPONENT_CONFIG 5-134
GET_COMPONENT_INSTALL_FORM. 5-134
GET_COMPONENT_INSTALL_PROMPTS_FORM. 5-134
GET_COMPONENT_INSTALL_SETTINGS’. 5-134
GET_LOCAL_REGISTRATION_FORM 5-135
GET_MANIFEST_INFO. 5-135
GET_SERVER_OUTPUT 5-135
PROXIED_REQUEST 5-135
UNINSTALL_COMPONENT 5-135
UPDATE_COMPONENT_CONFIG. 5-135
UPDATE_LICENSE. 5-135
UPLOAD_NEW_COMPONENT 5-135
Meta Services (Core Content Server) 5-135
ADD_METADEF 5-136
DEL_METADEF. 5-138
EDIT_METADEF 5-138
GET_ADVANCED_SEARCH_OPTIONS 5-139
GET_DOC_METADATA_INFO 5-139
GET_OPTION_LIST 5-140
GET_USER_METADATA_INFO 5-140

MOVE_METADEF 5-140
UPDATE_ADVANCED_SEARCH_OPTIONS 5-141
UPDATE_USER_META 5-141
UPDATE_USER_META_TABLE 5-141
UPDATE_META_TABLE 5-142
UPDATE_OPTION_LIST 5-142
Miscellaneous Services (Core Content Server) 5-143
CHUNKED_UPLOAD 5-143
DOWNLOAD_LISTBOX_ITEMS 5-144
LOAD_USER_LOCALIZATION 5-144
Page Handler/Page Request Services (Core Content Server) 5-144
DELETE_RESULT_TEMPLATE 5-145
GET_ADMIN_PAGE 5-145
GET_DOC_PAGE 5-146
GET_DYNAMIC_PAGE 5-149
GET_PORTAL_PAGE 5-150
GET_SECURE_PAGE 5-151
LOAD_GLOBALINCLUDES 5-153
PAGE_HANDLER 5-153
PNE_SAVE_QUERY 5-154
PNE_UPDATE_PERSONAL_URLS 5-154
PNE_UPDATE_PORTAL_INFO 5-155
SAVE_GLOBALINCLUDES 5-155
UPDATE_RESULT_TEMPLATE 5-155
Provider Manager Services (Core Content Server) 5-157
ADD_EDIT_PROVIDER 5-157
APPEND_DATABASE_AUDIT_INFO 5-158
DELETE_PROVIDER 5-158
ENABLE_DISABLE_PROVIDER 5-159
GET_ADD_EDIT_PROVIDER_FORM 5-160
GET_ALL_PROVIDERS 5-161
GET_PROVIDER_INFO 5-161
NOTIFY_CHANGE 5-161
REQUEST_SECURITYINFO 5-162
TEST_PROVIDER 5-162
Schema Services (Core Content Server) 5-162
ADD_SCHEMA_FIELD 5-163
ADD_SCHEMA_RELATION 5-164
ADD_SCHEMA_VIEW 5-164
ADDOREDIT_SCHEMA_TABLE 5-164
CONTROL_SCHEMA 5-164

DELETE_SCHEMA_FIELD 5-164
DELETE_SCHEMA_RELATION 5-165
DELETE_SCHEMA_TABLE 5-165
DELETE_SCHEMA_VIEW 5-165
EDIT_SCHEMA_FIELD 5-165
EDIT_SCHEMA_NODE 5-166
EDIT_SCHEMA_RELATION 5-166
EDIT_SCHEMA_VIEW 5-166
EDIT_SCHEMA_VIEW_VALUES 5-167
GET_SCHEMA_FIELD_INFO 5-167
GET_SCHEMA_FIELDS 5-167
GET_SCHEMA_RELATIONS 5-167
GET_SCHEMA_STATS 5-167
GET_SCHEMA_TABLE_INFO 5-168
GET_SCHEMA_TABLES 5-168
GET_SCHEMA_VIEW_EDIT_INFO 5-168
GET_SCHEMA_VIEW_FRAGMENT 5-168
GET_SCHEMA_VIEW_INFO 5-169
GET_SCHEMA_VIEW_VALUES 5-169
GET_SCHEMA_VIEWS 5-169
PUBLISH_SCHEMA 5-169
Search Services (Core Content Server) 5-169
APPEND_SEARCH_AUDIT_INFO 5-170
GET_EXTERNAL_DOC_INFO 5-170
GET_EXTERNAL_HIGHLIGHT_INFO 5-171
GET_EXTERNAL_XML_HIGHLIGHT_INFO 5-171
GET_HIGHLIGHT_INFO 5-172
GET_SEARCH_RESULTS 5-174
GET_SEARCH_RESULTS_FORCELOGIN 5-175
GET_XML_HIGHLIGHT_INFO 5-175
PNE_GET_SEARCH_RESULTS 5-176
VIEW_DOC 5-177
User Services (Core Content Server) 5-178
ADD_ALIAS 5-179
ADD_GROUP 5-181
ADD_ROLE 5-181
ADD_USER 5-182
CHANGE_USER_AUTH_TYPE 5-184
CHECK_USER_CREDENTIALS 5-185
DELETE_ALIAS 5-189
DELETE_GROUP 5-190
DELETE_ROLE 5-190

DELETE_USER 5-191
EDIT_ALIAS 5-191
EDIT_GROUP 5-194
EDIT_ROLE 5-194
EDIT_USER 5-195
EDIT_USER_PROFILE 5-198
GET_ALIASES 5-200
GET_FILTER_ADMIN_PAGE 5-200
GET_SELF_REGISTER_PAGE 5-200
GET_USER_INFO 5-200
GET_USERS 5-202
LOAD_PNE_PORTAL 5-202
LOAD_USER_TOPIC 5-203
QUERY_GROUP 5-203
QUERY_USER_ATTRIBUTES 5-203
REGISTER_USER 5-204
SAVE_USER_TOPICS 5-205
UPDATE_FILTER_INFO 5-209
UPDATE_USEROPTION_LIST 5-209

Chapter 6: Workflow Services

Overview 6-1
About Workflow Services 6-1
Doc and General Services (Workflows) 6-2
ADD_PROBLEMREPORT 6-3
DELETE_PROBLEMREPORT 6-4
GET_CRITERIA_WORKFLOWS_FOR_GROUP 6-5
GET_DOCUMENT_PROBLEMREPORTS 6-5
GET_PROBLEMREPORTS_SEARCH_FORM 6-6
GET_PROBLEMREPORTS_SEARCH_RESULTS 6-6
GET_UPDATE_PROBLEMREPORT_FORM 6-7
GET_WORKFLOWDOCUMENTS 6-7
GET_WORKFLOWS_FOR_ALL 6-8
LOAD_WORKFLOW_QUEUE 6-8
NOTIFY_CONTRIBUTOR 6-9
PROBLEMREPORT_INFO 6-9
RESEND_PROBLEMREPORT 6-9
REVIEW_WORKFLOW_DOC 6-10
UPDATE_PROBLEMREPORT 6-10
WORKFLOW_CHECKIN_SUB 6-11
WORKFLOW_EDIT_REV 6-11

WORKFLOW_NEW_REV 6-12
WORKFLOW_REJECT_FORM. 6-12
Workflow Template Services 6-13
ADD_WF_TEMPLATE. 6-13
DELETE_WF_TEMPLATE 6-13
EDIT_WF_TEMPLATE 6-14
GET_WF_TEMPLATE. 6-20
GET_WF_TEMPLATES 6-20
Workflow Services 6-20
ADD_WORKFLOW 6-22
ADD_WORKFLOW_SCRIPT 6-23
ADD_WORKFLOW_TOKEN 6-23
ADD_WORKFLOWALIASES. 6-24
ADD_WORKFLOWDOCUMENT. 6-24
ADD_WORKFLOWDOCUMENT_SUB 6-25
ADD_WORKFLOWDOCUMENTS 6-25
ADD_WORKFLOWSTEP 6-26
CRITERIAWORKFLOW_DISABLE 6-27
CRITERIAWORKFLOW_DISABLE_SUB 6-28
CRITERIAWORKFLOW_ENABLE 6-28
DELETE_WFCONTRIBUTORS 6-28
DELETE_WORKFLOW 6-29
DELETE_WORKFLOW_SCRIPT 6-29
DELETE_WORKFLOW_TOKEN. 6-30
DELETE_WORKFLOWCRITERIA 6-30
DELETE_WORKFLOWDOCUMENTS 6-30
DELETE_WORKFLOWSTEP 6-32
EDIT_WORKFLOW. 6-32
EDIT_WORKFLOW_SCRIPT 6-33
EDIT_WORKFLOW_TOKEN. 6-34
EDIT_WORKFLOWCRITERIA 6-34
EDIT_WORKFLOWSTEP 6-35
GET_ACTIVE_WORKFLOWS 6-37
GET_ALL_WORKFLOWDOCREVISIONS 6-37
GET_CRITERIA_WORKFLOWS_FOR_GROUP 6-38
GET_WF_COMPANION_INFO 6-38
GET_WORKFLOW 6-39
GET_WORKFLOW_INFO 6-39
GET_WORKFLOW_INFO_BYNAME 6-40
GET_WORKFLOW_SCRIPT. 6-42
GET_WORKFLOWDOCREVISIONS 6-42
GET_WORKFLOWS 6-44

TEST_WORKFLOW_SCRIPT 6-44
WORKFLOW_APPROVE 6-45
WORKFLOW_CANCEL 6-45
WORKFLOW_CHECKIN 6-45
WORKFLOW_EDIT_APPROVE 6-50
WORKFLOW_REJECT 6-50
WORKFLOW_START 6-53

Chapter 7: Archiving Services

Overview 7-1
About Archiving Services 7-1
Archive Services 7-3
ADD_ARCHIVE 7-5
ADD_COLLECTION 7-5
ADD_PROXIEDCOLLECTION 7-6
CANCEL_ARCHIVE 7-6
CHECKIN_ARCHIVE 7-6
COPY_ARCHIVE 7-11
DELETE_ARCHIVE 7-11
DELETE_BATCH_FILE 7-12
DELETE_BATCH_FILE_DOCUMENTS 7-12
DELETE_BATCH_FILE_TABLES 7-13
EDIT_ARCHIVE 7-13
EDIT_ARCHIVEDATA 7-14
EDIT_EXPORTERS 7-14
EDIT_TRANSFEROPTIONS 7-15
EXECUTE_BATCH 7-15
EXPORT_ARCHIVE 7-15
GET_ARCHIVECOLLECTIONS 7-17
GET_ARCHIVETABLECONTENT 7-17
GET_ARCHIVED_FILE 7-17
GET_ARCHIVES 7-19
GET_ARCHIVERELATIONQUERY 7-19
GET_BATCH_FILE_DOCUMENTS 7-19
GET_BATCH_PROPERTIES 7-20
GET_BATCH_SCHEMA 7-20
GET_BATCH_VALUES 7-21
GET_BATCHFILES 7-21
GET_PROXIED_ARCHIVECOLLECTIONS 7-22
GET_PROXIEDSERVERS 7-22
GET_REPLICATION_DATA 7-22

GET_TABLECOLUMNLIST 7-23
GET_TARGET_INFO 7-23
GET_TARGET_TRANSFER_STATUS 7-23
GET_TRANSFER_SOURCE_INFO 7-24
IMPORT_ARCHIVE 7-24
IMPORT_ARCHIVE_START_AUTOMATED 7-26
IMPORT_BATCHFILE 7-26
IMPORT_DOCUMENT 7-26
IMPORT_TABLE_ENTRY 7-27
INSERT_NATIVE 7-27
INSERT_NEW 7-28
REGISTER_IMPORTER 7-28
REMOVE_COLLECTION 7-29
REMOVE_EXPORTER 7-29
REMOVE_IMPORTER 7-30
REMOVE_PROXIEDTRANSFER 7-30
REMOVE_QUEUED_IMPORT 7-30
REMOVE_TRANSFER 7-31
REQUEST_TRANSFER 7-31
TRANSFER_ARCHIVE 7-31
UPDATE_TARGET_TOTALS 7-32
UPDATE_TRANSFER_STATUS 7-32
UPLOAD_ARCHIVE_TRANSFER 7-32

Chapter 8: Folders Services

Overview 8-1

About Folder Services 8-1

Folder Services 8-2

COLLECTION_ADD 8-5
COLLECTION_ADD_LINK 8-6
COLLECTION_ADD_MOUNT 8-6
COLLECTION_BROWSE 8-7
COLLECTION_CHECKIN_NEW 8-7
COLLECTION_CHECKIN_REVISION 8-7
COLLECTION_CHECKIN_SEL_SUB 8-8
COLLECTION_COPY_ALL 8-8
COLLECTION_COPY_COLLECTION 8-8
COLLECTION_COPY_ITEM 8-9
COLLECTION_COPY_LOT 8-9
COLLECTION_DELETE_ALL 8-9
COLLECTION_DELETE_COLLECTION 8-9

COLLECTION_DELETE_ITEM 8-9
COLLECTION_DELETE_LOT. 8-10
COLLECTION_DISPLAY. 8-10
COLLECTION_EDIT 8-11
COLLECTION_GET_ADMIN_CONFIG. 8-11
COLLECTION_GET_ADMIN_INHERIT_CONFIG 8-11
COLLECTION_GET_ADMIN_MARKED_CONFIG 8-11
COLLECTION_GET_ADMIN_META_CONFIG. 8-12
COLLECTION_GET_ADMIN_METADATA_DEFAULTS 8-12
COLLECTION_GET_ADMIN_MOUNTED_CONFIG. 8-12
COLLECTION_GET_ARCHIVE 8-12
COLLECTION_GET_BRANCH 8-13
COLLECTION_GET_COLLECTION 8-13
COLLECTION_GET_COLLECTIONS. 8-13
COLLECTION_GET_CONTENT_FILE 8-13
COLLECTION_GET_CONTENTS. 8-14
COLLECTION_GET_FILE. 8-14
COLLECTION_GET_INFO 8-14
COLLECTION_GET_LINKS 8-14
COLLECTION_GET_META_MAPPING 8-14
COLLECTION_GET_PROFILE_METADATA_DEFAULTS. 8-15
COLLECTION_GET_PROFILE_METADATA_REVISION_DEFAULTS .8-15
COLLECTION_GET_REFERENCE 8-15
COLLECTION_GET_SEARCH_FORM. 8-15
COLLECTION_GET_SYSTEM_FILE 8-16
COLLECTION_GET_USER_CONFIG 8-16
COLLECTION_INDEX_MOUNTS 8-16
COLLECTION_INFO 8-16
COLLECTION_ISVALID_META 8-17
COLLECTION_LOCK 8-17
COLLECTION_MOVE_ALL. 8-17
COLLECTION_MOVE_COLLECTION 8-17
COLLECTION_MOVE_ITEM. 8-17
COLLECTION_MOVE_LOT 8-18
COLLECTION_NEW 8-18
COLLECTION_PROFILE_UPDATE_COLUMNS 8-19
COLLECTION_REBUILD_MOUNT 8-19
COLLECTION_REMOVE_MOUNT 8-19
COLLECTION_RESTORE_COLLECTION 8-19
COLLECTION_RESTORE_ITEM 8-20
COLLECTION_SEARCH_CONTENT 8-20
COLLECTION_SEARCH_RESULTS 8-20

COLLECTION_SET_ARCHIVE 8-20
COLLECTION_SET_USER_CONFIG 8-20
COLLECTION_UNLOCK 8-21
COLLECTION_UPDATE 8-21
COLLECTION_UPDATE_ADMIN_CONFIG 8-22
COLLECTION_UPDATE_ADMIN_INHERIT_CONFIG 8-22
COLLECTION_UPDATE_ADMIN_METADATA_DEFAULTS 8-22
COLLECTION_UPDATE_ALL 8-22
COLLECTION_UPDATE_ITEM 8-22
COLLECTION_UPDATE_META 8-23
COLLECTION_UPDATE_META_TABLE 8-23
COLLECTION_UPDATE_MOUNTS 8-23
COLLECTION_UPDATE_PROFILE_METADATA_DEFAULTS 8-23
COLLECTION_UPDATE_PROFILE_METADATA_REVISION_DEFAULTS 8-24
COLLECTION_UPDATE_STRUCTURE 8-24
GOTO_COLLECTION 8-24
GOTO_ROOT_COLLECTION 8-24
GET_OPTION_LISTS 8-24
GET_ENVIRONMENT 8-25

Chapter 9: Content Publisher Services

Overview 9-1
About Content Publisher Services 9-1
Content Publisher Services 9-2
CHECKIN_PUBLISH 9-4
END_PUBLISHING 9-4
GET_PREVIEW_LIST 9-5
GET_PREVIEW_LIST_EX 9-5
GET_PROJECT_ACTIONS 9-5
GET_PROJECT_INFO 9-6
GET_PROJECT_REGISTRATION 9-6
GET_REGISTEREDPROJECTS 9-7
PREVIEW_DOCUMENT 9-7
REGISTER_PROJECT 9-7
START_PUBLISHING 9-7
UNREGISTER_PROJECT 9-8

Chapter 10: Record Manager Services

Overview 10-1
About Record Manager Services 10-1

Record Manager Services 10-2
ACTIVATE_SERVICE 10-4
APPROVE_DELETE_SERVICE 10-5
BROWSE_CATEGORY_FORM 10-5
BROWSE_FOLDER_FORM 10-5
BROWSE_SERIES_FORM 10-5
CHECKIN_NEW_REVISION_SERVICE 10-5
CHECKIN_SIMILAR_FORM 10-6
CLEAR_FOLDER_CANCELLED_DATE 10-6
CLEAR_FOLDER_EXPIRATION_DATE 10-6
CLEAR_FOLDER_OSBOLETE_DATE 10-6
CLEAR_FOLDER_RESCINDED_DATE 10-6
CLEAR_FOLDER_REVIEW_DATE 10-7
CLEAR_RECORD_CANCELLED_DATE 10-7
CLEAR_RECORD_EXPIRATION_DATE 10-7
CLEAR_RECORD_OBSOLETE_DATE 10-7
CLOSE_FOLDER 10-7
CREATE_FOLDER 10-8
CREATE_FOLDER_FORM 10-8
DELETE_ALL_BUT_LAST_N_REVISIONS_SERVICE 10-8
DELETE_FOLDER 10-9
DELETE_REVISION_SERVICE 10-9
EDIT_FOLDER 10-9
EDIT_FOLDER_FORM 10-9
FREEZE_FOLDER 10-10
FREEZE_RECORD 10-10
INFO_CATEGORY_FORM 10-10
INFO_FOLDER_FORM 10-11
INFO_FOLDER_LIFECYCLE 10-11
INFO_FOLDER_METADATA_HISTORY 10-11
INFO_FOLDER_REVIEW_HISTORY 10-11
INFO_RECORD_LIFECYCLE 10-11
INFO_RECORD_METADATA_HISTORY 10-12
INFO_RECORD_REVIEW_HISTORY 10-12
INFO_SERIES_FORM 10-12
MARK_FOLDER_ACTIVATION_DATE 10-12
MARK_FOLDER_CANCELLED_DATE 10-12
MARK_FOLDER_EXPIRATION_DATE 10-13
MARK_FOLDER_OBSOLETE_DATE 10-13
MARK_FOLDER_RESCINDED_DATE 10-13
MARK_FOLDER_REVIEW_DATE 10-13
MARK_FOLDER_REVIEW_DATE_RECURSIVE 10-14

MARK_RECORD_CANCELLED_DATE 10-14
MARK_RECORD_EXPIRATION_DATE 10-14
MARK_RECORD_OBSOLETE_DATE 10-14
MARK_RECORD_RESCINDED_DATE 10-15
MARK_RECORD_REVIEW_DATE 10-15
MOVE_FOLDER 10-15
PREVIEW_RECORD_LIFECYCLE 10-15
RMA_CLOSE_SERVICE 10-16
RMA_CUTOFF_SERVICE 10-16
RMA_DESTROY_SERVICE 10-16
RMA_EXPORT_ARCHIVE_SERVICE 10-16
RMA_EXPORT_SERVICE 10-17
RMA_MARK_COMPLETED 10-17
RMA_NO_ACTION_SERVICE 10-17
RMA_OBSOLETE_SERVICE 10-17
RMA_SCRUB_SERVICE 10-18
RMA_SUPERSEDE_SERVICE 10-18
UNCLOSE_FOLDER 10-18
UNFREEZE_FOLDER 10-18
UNFREEZE_RECORD 10-19

Appendix A: Actions

Overview A-1
About Service Actions A-1
A A-2
B A-5
C A-5
D A-17
E A-25
F A-28
G A-28
H A-34
I A-34
L A-37
M A-41
N A-42
P A-43
Q A-47

R. A-58
S. A-63
T. A-68
U. A-69
V. A-74

Appendix B: Third Party Licenses

Overview B-1
Apache Software License B-1
W3C® Software Notice and License B-2
Zlib License B-3
General BSD License. B-4
General MIT License B-5
Unicode License. B-5
Miscellaneous Attributions B-7

Chapter

1

INTRODUCTION

This guide provides detailed information about Product_Name services, which are functions or procedures performed by the content server. This chapter contains these topics:

- About This Guide (page 1-1)
- Organization (page 1-2)
- Audience (page 1-2)
- Conventions (page 1-3)

Further details about these services and their use can be found in *The Definitive Guide to Stellent Content Server Development* by Brian Huff, © 2006, Apress, Berkeley, CA.

ABOUT THIS GUIDE

Calling a content server service (making a service request) is the only way to communicate with the content server or access the database. This guide describes service usage and syntax, and provides detailed descriptions and examples of commonly used and predefined Content Server services.

Note: The information contained in this guide is based on Content Server 10gR3. The information is subject to change as the product technology evolves and as hardware and operating systems are created and modified. Due to the technical nature of browsers, databases, web servers, and operating systems, Oracle, Inc. cannot warrant compatibility with all versions and features of third-party products.

Note: This reference guide is part of the Software Developer's Kit (SDK). For more information about customizing and integrating Content Server, see *Getting Started with the Software Developer's Kit (SDK)*.

ORGANIZATION

This guide is divided into the following chapters:

- Chapter 2 (*What's New)* lists changes to this documentation and to standard Content Server services.
- Chapter 3 (*Using Services)* explains how content server services work and provides examples of how content server services can be called externally from custom applications.
- Chapter 4 (*Customizing Services)* explains the structure of content server services, provides an example of how services, queries, and templates work together, and describes how to create a service resource for use with a custom component.
- Chapter 5 (*Core Content Server Services)* groups standard Content Server services by common usage and describes those services.
- Chapter 6 (*Workflow Services)* describes Content Server workflow service.
- Chapter 7 (*Archiving Services)* describes archiving services used in Content Server.
- Chapter 8 (*Folders Services)* describes the services used with the Folders component.
- Chapter 9 (*Content Publisher Services)* describes services describes services used with the Content Publisher.
- Chapter 10 (*Record Manager Services)* describes services used with the Record Manager optional component.
- Appendix A (*Actions*) describes the actions used by individual services.

An index is provided at the end of this guide.

AUDIENCE

This guide is intended for developers who need to use services to customize the software and for application developers who need to access Content Server functions.

CONVENTIONS

The following conventions are used throughout this guide:

- The notation <*Install_Dir*>/ is used to refer to the location on your system where the content server instance is installed.
- Forward slashes (/) are used to separate the directory levels in a path name. A forward slash will always appear after the end of a directory name.
- Notes, technical tips, important notices, and cautions use these conventions:

Symbols	Description
	This is a note. It is used to bring special attention to information.
	This is a tech tip. It is used to identify information that can be used to make your tasks easier.
	This is an important notice. It is used to identify a required step or required information.
	This is a caution. It is used to identify information that might cause loss of data or serious system problems.

Chapter

2

WHAT'S NEW

OVERVIEW

This chapter discusses changes to the documentation for this release. Highlights of these changes include the following:

- Revised Documentation (page 2-1)
- Changes to Services (page 2-1)

REVISED DOCUMENTATION

The documentation for services has been extensively changed from previous releases of Content Server. This document is now arranged according to Content Server functionality or product, making it easier for you to find information about specific services.

Throughout this documentation, services which are used extensively have an extended description. In addition, frequently-used services are marked with an asterisk (*) in introductory lists in each chapter.

CHANGES TO SERVICES

A new Service Class, DocProfiles, has been added. These services perform actions related to the document profile function.

NEW SERVICES INTERFACE

When adding a service via the Component Wizard, you can now preview details about services. Information for a service can be viewed by selecting the new **Preview** button in the upper right corner of the Resource Selection Dialog. When the Preview Information for 'SERVICE_NAME' dialog is displayed, the online help for that service is available by clicking the **Help** button.

See Creating a Custom Service using Component Wizard (page 4-32), step 9, for information about accessing this new functionality.

Chapter

3

USING SERVICES

OVERVIEW

This chapter discusses how to use services and provides an example of a custom service. It covers the following topics:

- Service Overview (page 3-1)
- Custom Application Example (page 3-10)
- Redirecting Template Page for Response Output (page 3-18)

SERVICE OVERVIEW

A service is a function or procedure that is performed by the content server. Calling a content server service (making a service request) is the only way a client can communicate with the content server or access the database.

This section covers the following topics:

- Service Requests and Responses (page 3-2)
- Page Retrieval (page 3-4)
- Content Server Search Services (page 3-5)
- Integration Methods (page 3-5)
- Calling Services Using Persistent URLs (page 3-6)
- Customizing Locale Parameters (page 3-8)

Service Requests and Responses

Any service can be called either externally (from outside the content server) or internally (within the content server itself). Typically, client services are called externally, while administrative services are called internally. When a service is requested, any applicable parameters are passed to the service. The service uses its attributes and actions to execute the request based on the specified parameters. The service then returns a response either externally or internally, as applicable. This section covers the following topics:

- Internal Service Requests (page 3-2)
- External Service Requests (page 3-2)
- Request Parameters (page 3-3)
- Case Sensitivity Considerations (page 3-4)

Internal Service Requests

Although any service can be called internally, typically only administrative services are called internally. Internal service requests are made from within the content server itself, and results are returned only to the content server. For example, you can use the START_SEARCH_INDEX service to update or rebuild the search index automatically in a background thread.

External Service Requests

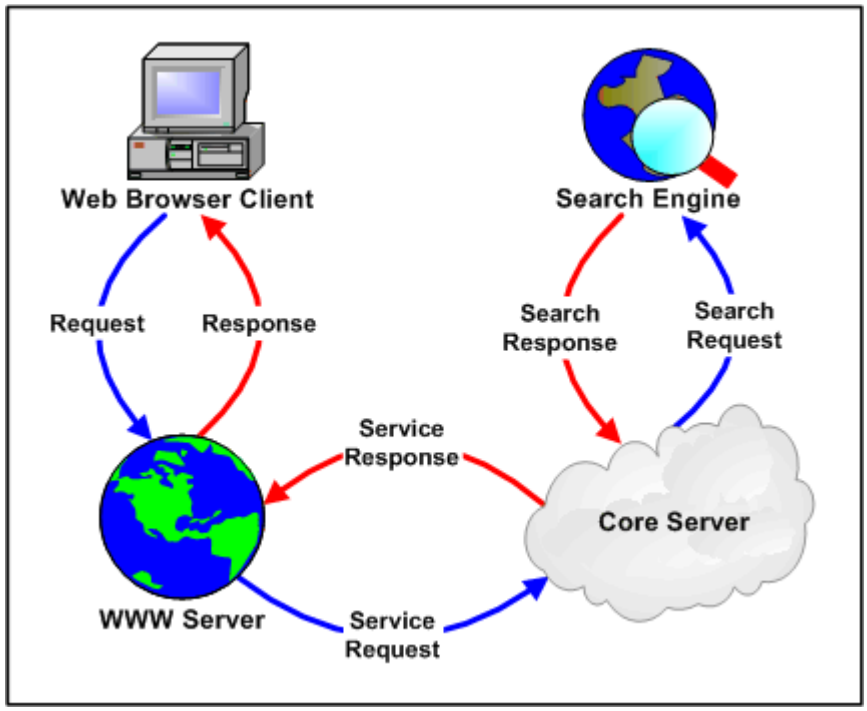
Any external program or HTML page can call any content server service to request information from the content server or perform a specified function, such as full-text and metadata searching, library services, workflow services, subscription notifications, and content conversion capabilities. Typically, only client services are called externally (administrative services are typically called internally). For example, when you click a **Search** link on a content server web page, the standard search page is delivered to your web browser by the GET_DOC_PAGE service using the following URL segment:

```
IdcService=GET_DOC_PAGE&Action=GetTemplatePage&Page=
STANDARD_QUERY_PAGE
```

External requests are sent from a web browser client to the web server using one of many protocols. The service call must include any parameters that the service requires. The web server routes the service request to the content server, along with any required and optional parameters. The content server then executes the service using the provided parameters. In the case of search services, this involves sending a search request to the

search engine. The content server then returns the results to the web server, and the web server returns the results to the web browser client.

Figure 3-1 External requests and responses between content management system elements.



Request Parameters

A service request must include any parameters that the service requires. For example, when calling the DOC_INFO service to obtain information about a content item, the service call must provide the dID (generated content item revision identifier) to the service. The following segment shows how this would be done through a persistent URL:

```
http://cs.company.com/intradoc-cgi/idc_cgi_isapi.dll?
IdcService=DOC_INFO&dID=194
```

Date and Time Formatting

Default date and time formatting are determined using the Localization tab on the System Properties. The general format for date and time is:

```
MM/DD/{yy} {hh:mm[:ss] {aa} [zzz]} !mAM,PM!zTimezoneCity
```

- The date/time format is a grouping of Y, D, M for Year, Day, and Month, and h, m, and s for hours, minutes, and seconds. The number of times the letter repeats designates the minimum number of digits used (for example, YY/MM/DD hh:mm could

designate 04/12/09 12:12 or MM/DD/YYYY hh:mm:ss would be 09/12/2004 04:12:33).

- The `a` designation represents either AM or PM if the `mAM,PM` designation is used to specify the meridian symbols.
- The `z` designation specifies the time zone, usually a city chosen from a list.
- Square brackets ([ ]) around an item indicates that it will be parsed if present but won't be in the output when used for formatting.
- Braces ({ }) indicate that the item is optional when in the input data, but will always appear in the output.
- The exclamation mark (!) is used to separate additional date format specifications.

See *Using Content Server in International Environments* for details about setting up system properties, including the default date and time designations.

Case Sensitivity Considerations

Case sensitivity is important when calling standard Content Server services.

- **Parameters**—Parameters are case sensitive. For example, when specifying the IdcService parameter you must use IdcService, not IDCSERVICE.
- **Parameter values**—Parameter values are typically case sensitive. The value for the IdcService parameter is always case sensitive, and the convention used for standard Content Server services is all capital letters. For example, when specifying the value for the IdcService parameter you must use DOC_INFO, not Doc_Info.
- **Databases**—The database you are using with Content Server might affect the case sensitivity of parameters and parameter values.

Page Retrieval

When a web page is requested from the content server, one of the following page types is returned:

- **static page—**The content of a static web page is pre-formatted, and does not change from one request to the next. In a typical content server website, the only static page is the guest home page (<*install_dir*>/weblayout/portal.htm).
- **dynamic page—**A dynamic web page is assembled at the time of the web server request, using content server services and templates to determine the content and formatting. For example, each user's portal design page is generated using a content

server service called GET_PORTAL_PAGE and a template called PNE_PORTAL_DESIGN_PAGE.

Content Server Search Services

A search request is a special kind of content server service. When the content server receives a search request, it sends the request on to the search engine using a search engine API. This allows different search engines to be used with the content server.

Integration Methods

Service requests can be made by any external program or HTML page using a wide variety of protocols. Content Server can be integrated with other enterprise applications using a wide variety of integration methods. One common integration method is to reference content that is managed within the content server by persistent URL. For more information, see Calling Services Using Persistent URLs (page 3-6).

The following are other possible integration methods:

- Java API (IdcCommand) integration using the IdcCommand Java Command Utility
- Component Object Model (COM) integration using the ActiveX utility or the IntradocClient OCX component
- Java Server Page (JSP) integration from a JSP running in Content Server, a JSP through the Content Server JavaBean, or a JSP through the Content Server Enterprise JavaBean (EJB) deployed on your J2EE application server
- Java 2 Enterprise Edition API (J2EE) integration by deploying the Content Server Enterprise JavaBean on your J2EE-compliant application server
- Common Object Request Broker Architecture (CORBA) integration by implementing Remote Method Invocation (RMI) over Internet Inter-ORB Protocol (IIOP) as the transport protocol and referencing the Enterprise JavaBean
- Open Document Management API (ODMA) integration using the ODMA-based plug-in
- Simple Object Access Protocol (SOAP) integration using the SOAP protocol
- Virtual Folders integration using the Folders component
- Web Distributed Authoring and Versioning (WebDAV) integration using the WebDAV component

Note: For more detailed information on available integration methods, see *Getting Started with the Software Developer's Kit (SDK)*.

Calling Services Using Persistent URLs

In this integration method, all of the necessary information for the service call is sent to the content server through the URL. The following is a typical URL; in this case, it is the URL for the Home page:

```
http://cs.company.com/intradoc-cgi/idc_cgi_isapi.dll?
IdcService=GET_DOC_PAGE&Action=GetTemplatePage&Page=HOME_PAGE
```

- **`http://cs.company.com/`** is the web address of the content server instance.
- **`intradoc-cgi/idc_cgi_isapi.dll`** is the path to the web server filter.
- **`IdcService=GET_DOC_PAGE`** tells the content server to execute the GET_DOC_PAGE service.
- **`Action=GetTemplatePage`** tells the content server to return the results using a specified template page.
- **`Page=HOME_PAGE`** tells the content server which template page to use.
- The question mark (**?**) indicates the end of the web server path and the beginning of content server instructions.
- Ampersands (**&**) are used as separators between content server instructions.
- You can include certain Idoc Script variables in a URL to affect page display at the time of the page request. This is useful for troubleshooting or for customizing your content server pages. For example:

  Troubleshooting Examples

  - IsJava=1
  - ScriptDebugTrace=1
  - ScriptErrorTrace=1

  Customization Examples

  - &StdPageWidth=1000
  - &dDocAuthor:isHidden
  - &dDocType=HRForm

Example 1

The following example describes the steps that occur when a persistent URL is used to request a dynamic page from the content server.

1. When a user clicks the Administration link in the navigation area, a request for the GET_ADMIN_PAGE service is sent to the web server. The URL of the Administration link contains the following commands:
   ```
   IdcService=GET_ADMIN_PAGE&Action=GetTemplatePage&Page=ADMIN_LINKS
   ```
2. The web server recognizes this request as a content server function, and sends the specific request to the content server.
3. When the content server has processed the request, it passes the result back to the web server. In the case of the Administration link, the GET_ADMIN_PAGE service:
   - Provides a login prompt if the user is not currently logged in.
   - Verifies that the user has *admin* permission.
   - Assembles the Administration page using the ADMIN_LINKS template.
   - Returns the assembled web page to the web server.
4. The web server delivers the results of the content server service to the originating web browser client.

Example 2

The following example describes the steps that occur when a persistent URL is used to perform a search request.

1. When a user clicks the Search button on the standard Search page, a request for the GET_SEARCH_RESULTS service is sent to the web server. The URL for the search request specifies the service to execute, the search criteria, and the result parameters:
   ```
   IdcService=GET_SEARCH_RESULTS&QueryText=stellent&ftx=1
   &AdvSearch=True&ResultCount=25&SortField=dInDate&SortOrder=Desc
   ```
2. The web server recognizes the request as a content server function, and sends the specific request to the content server.
3. The content server passes the request to the search engine.
4. The search engine returns the search results to the content server.
5. Based on the user login and security permissions, the content server assembles the search results page and returns it to the web server.
6. The web server delivers the results to the originating web browser client.

Customizing Locale Parameters

When using the content server in a client server operation mode, several parameters can be used to improve the handling of locale-sensitive data, helping to avoid date and encoding incompatibility problems.

In the following descriptions, it is assumed that an HDA formatted request is being made to the content server using an operation such as IdcClient, IdcCommandUX, IdcServerBean, or a custom server communication.

The following parameters are available:

- `UserDateFormat`: Specifies the date/time format for dates in the incoming request and any dates produced in the response to the request. If not specified, the response format always uses the login user's date locale. If that is unavailable, it uses the content server system locale date. Do not use the ODBC date/time format used internally by the content server for archiver batch load files; that can create time zone ambiguity errors.

  Example:

  ```
  UserDateFormat=M/D/YYYY hh:mm[:ss]{aa}!mAM,PM!zUTC
  ```

- `blDateFormat`: Specifies the format of dates in the body of the incoming request. It does not affect the format of the response. If set, this overrides values set with `UserDateFormat`. If not specified, the incoming request format uses the login user's locale date. If that is unavailable, it uses the content server system locale date. This parameter is not available except if one of the headers that precedes the request is a `REQUEST_METHOD` header with a `POST` value. `GET` style requests do not support this parameter.

- `ClientEncoding`: Specifies the character encoding to be used in the response. If no HDA header line exists with either the *charset* or *jcharset* specification, this parameter also dictates the encoding used to decode the body of the request.

  Example:

  ```
  ClientEncoding=cp1252
  ```

- `HEADER_ENCODING`: Used in the header that precedes the body of the HDA request. This allows the requesting agent to specify the encoding of headers. This functionality is useful for dictating the encoding of the `HTTP_INTERNETUSER` and `REMOTE_USER` header entities.

- `blFieldTypes`: Specifies a field type (date or message) during the translation of a response. A ‘message’ is an unlocalized string, usually an error message that has not been localized. This parameter can be used to ensure that ODBC formatted fields are

parsed and returned in the format specified by `blDateFormat` or `UserDateFormat` and will be applied even if the field is only in the response and not in the request.

The only fields that typically need such translation are ones created by customized extensions of content server behavior (such as those created using Idoc script, for example). This is only available for `POST` style requests (`REQUEST_METHOD` header with value `POST`). This is normally used to specify additional date fields and is best used in combination with `UserDateFormat` or `blDateFormat` to indicate that a field needs special handling as a date.

Example:

```
blFieldTypes=xNewDate date, xComment message
```

- `extraFieldTypes`: Specifies a field type during the translation of a response to HDA formatted requests, used in place of `blFieldTypes`. It is best used inside an Idoc script expression that is executed when fulfilling a content server request. It cannot be used in Idoc script when generating the format of the response because the response is in a data format (such as HDA). It can, however, be used in other places where Idoc script is evaluated when executing the logic of a request, such as the processing of an HCSP form, or determining the effects of a document profile.

  Example:

  ```
  extraFieldTypes=xNewDate date, xComment message
  ```

- `convertDatabaseDate`: Ensures that `blFieldTypes` or `extraFieldTypes` are used to convert the ODBC date format to the desired response date format. This variable should be used when using the `blFieldTypes` or `extraFieldTypes` variables.

  If this variable is not set, the ODBC date formats may not be converted to the desired response date format. ODBC dates can still be converted even if this variable is not set. This occurs if the content server determines that the response needs a full coercion from one date format to another. This typically happens only if the incoming date format is different from the outgoing date format.

  Example:

  ```
  convertDatabaseDate=1
  ```

- `SuppressResultLocalization`: Suppresses localization conversions performed before the response is sent back. Conversions done on incoming data can still be performed.

  This parameter is useful to prevent messages from being localized or dates being fully converted. An example of usage is when the response is to be forwarded to another

content server to be processed or when the data is to be persistently stored and replayed back to the original content server as needed.

All of the parameters that affect date and message processing can be used with this one, but the other parameters are only used when the data is replayed against another content server or the current content server.

Example:

```
SuppressResultLocalization=1
```

Note: The operation making the HDA request may set values for these parameters; therefore, users may find that the values they set do not change the behavior as expected.

Forcing Authentication Challenges

It is sometimes necessary that a user be re-authenticated for a content server service or for other activities. For example, during a workflow, it might be necessary to acquire a 'digital signature' for a user at a specific step in the workflow process. This can be done using the `isRepromptLogin` configuration variable. See the *Idoc Script Reference Manual* for details about its usage.

To force re-authentication of any service, perform the following steps:

1. Add `checkForRevalidateLogin` as a service definition function.
2. Add `revalidateLoginID` as a parameter to the service call being made, with a randomly generated value.

These two actions cause the content server to refuse to accept the currently supplied credentials until the *AllowedLoginID* cookie is set with the same value as the parameter to `revalidateLoginID`. The cookie is set during the redundant challenge to the current credentials.

Note that this clears the current credentials so the user will need to login again to access any content server functionality.

CUSTOM APPLICATION EXAMPLE

This example application calls five services and defines six private functions:

Services Called

These services are called and a serialized HDA string is built for each:

- CHECKOUT_BY_NAME
- CHECKIN_UNIVERSAL
- DOC_INFO
- GET_FILE

These parameters for CHECKIN_UNIVERSAL are defined:

- doFileCopy
- dDocName
- dDocTitle
- dDocType
- dSecurityGroup
- dDocAuthor
- dDocAccount
- primaryFile

The GET_TABLE service is called and this parameter is defined:

- tableName

Private Functions

These private functions are defined:

- getNativeFilePath
- checkOutByName
- checkinUniversal
- getDocInfo
- getFile
- parseResultSet

Sample Code

```
' Defines a private function.
Private Function getNativeFilePath() As String
Dim idccmd As IdcCommandX
Dim str As String
Dim res
Dim dID As String, dExtension As String, dDocType As String, dDocAccount As String

dID = "2"
dExtension = "pdf"
dDocType = "acc"
dDocAccount = ""

' Builds a serialized HDA string.
str = "@Properties LocalData" + vbCrLf
str = str + "dID=" + dID + vbCrLf
str = str + "dExtension=" + dExtension + vbCrLf
str = str + "dDocType=" + dDocType + vbCrLf
str = str + "dDocAccount=" + dDocAccount + vbCrLf
```

```
str = str + "@end" + vbCrLf

Set idccmd = New IdcCommandX
res = idccmd.init("sysadmin", "c:\stellent\bin")
res = idccmd.computeNativeFilePath(str)

Open "c:\newdoc.txt" For Binary Access Write As #1
Put #1, , str
Put #1, , res
Close #1
MsgBox (res)
End Function

' Defines a private function.
Private Function checkOutByName(ByVal dDocName As String) As String
Dim idccmd As IdcCommandX
Dim idcService As String, str As String
Dim res

' Calls a service and builds a serialized HDA string.
idcService = "CHECKOUT_BY_NAME"
str = "@Properties LocalData" + vbCrLf
str = str + "IdcService=" + idcService + vbCrLf
str = str + "dDocName=" + dDocName + vbCrLf
str = str + "@end" + vbCrLf

' In an actual application the return codes need to be handled. For this example,
the service is called while there is no content with that specific dDocName.
Set idccmd = New IdcCommandX
res = idccmd.init("sysadmin", "c:\stellent\bin")
res = idccmd.executeCommand(str)
End Function

' Defines a private function.
Private Function checkinUniversal(ByVal doFileCopy As String, ByVal dDocName As
String, ByVal dDocTitle As String, ByVal dDocType As String, ByVal dSecurityGroup
As String, ByVal dDocAuthor As String, ByVal dDocAccount As String, ByVal
primaryFile As String) As String

' Builds a serialized HDA string.
Dim idccmd As IdcCommandX
Dim idcService, res, str As String

' Calls a service and builds a serialized HDA string.
idcService = "CHECKIN_UNIVERSAL"
str = "@Properties LocalData" + vbCrLf
str = str + "IdcService=" + idcService + vbCrLf
str = str + "doFileCopy=" + doFileCopy + vbCrLf
str = str + "dDocName=" + dDocName + vbCrLf
```

```
str = str + "dDocTitle=" + dDocTitle + vbCrLf
str = str + "dDocType=" + dDocType + vbCrLf
str = str + "dSecurityGroup=" + dSecurityGroup + vbCrLf
str = str + "dDocAuthor=" + dDocAuthor + vbCrLf
str = str + "dDocAccount=" + dDocAccount + vbCrLf
str = str + "primaryFile=" + primaryFile + vbCrLf
str = str + "@end" + vbCrLf

' exec hda...
Set idccmd = New IdcCommandX
res = idccmd.init("sysadmin", "c:\stellent\bin")
res = idccmd.executeCommand(str)
MsgBox (CStr(res))
End Function

' Defines a private function.
Private Function getDocInfo(ByVal dID As String) As String
Dim idccmd As IdcCommandX
Dim idcService As String, str As String
Dim res

' Calls a service and builds a serialized HDA string.
idcService = "DOC_INFO"
str = "@Properties LocalData" + vbCrLf
str = str + "IdcService=" + idcService + vbCrLf
str = str + "dID=" + dID + vbCrLf
str = str + "@end" + vbCrLf

' exec hda....
Set idccmd = New IdcCommandX
res = idccmd.init("sysadmin", "c:\stellent\bin")
res = idccmd.executeCommand(str)
MsgBox (res)
End Function

' Defines a private function.
Private Function getFile(ByVal dID As String, ByVal dDocName As String, ByVal
RevisionSelectionMethod As String, ByVal Rendition As String)
Dim idccmd As IdcCommandX
Dim idcService, str As String

Dim res As Variant
Dim fileName As String
Dim fileSize As Long
Dim indexStop As Integer

' Calls a service and builds a serialized HDA string.
idcService = "GET_FILE"
str = "@Properties LocalData" + vbCrLf
```

```
str = str + "IdcService=" + idcService + vbCrLf
str = str + "dDocName=" + dDocName + vbCrLf
str = str + "dID=" + dID + vbCrLf
If (RevisionSelectionMethod = "Specific" Or RevisionSelectionMethod = "Latest" Or
RevisionSelectionMethod = "LatestReleased") Then

' Ignore dDocName and use dID instead.
str = str + "RevisionSelectionMethod=" + RevisionSelectionMethod + vbCrLf
End If
If (Revision = "Primary" Or Revision = "Web" Or Revision = "Alternate") Then
str = str + "Revision=" + Revision + vbCrLf
End If
str = str + "@end" + vbCrLf

' exec hda...
Set idccmd = New IdcCommandX
res = idccmd.init("sysadmin", "c:\stellent\bin")
res = idccmd.executeCommand(str)

Open "c:\newdoc.txt" For Binary Access Write As #1
Put #1, , res
Close #1
MsgBox (Len(res))

' chop at filename= and store fileName
indexStop = InStr(res, "filename=")
tmpStr = (Mid(res, indexStop))
indexStop = InStr(tmpStr, Chr(13))
fileName = Mid(tmpStr, 10, indexStop - 10)
MsgBox (fileName)
' MsgBox (CStr(Asc(Mid(tmpStr, 2, 1))))

' chop at Content-length: and store fileSize
tmpStr = Mid(tmpStr, indexStop)
indexStop = InStr(tmpStr, "Content-Length: ")
tmpStr = (Mid(tmpStr, indexStop))
indexStop = InStr(tmpStr, Chr(10))
fileSize = CLng(Mid(tmpStr, 17, indexStop - 17))
MsgBox (CStr(fileSize))
MsgBox (Len(res))
End Function

Private Sub cmdAddUser_Click()
frmAddUser.Show
End Sub

Private Sub cmdCheckin_Click()
Dim idcService As String, doFileCopy As String, dDocName As String
Dim dDocTitle As String, dDocType As String, dSecurityGroup As String
```

```
Dim dDocAuthor As String, dDocAccount As String, primayFile As String

' Calls a service and defined parameters.
idcService = "CHECKIN_UNIVERSAL"
doFileCopy = "1"
dDocName = "myDocNameNewh"
dDocTitle = "myDocTitleb"
dDocType = "ADACCT"
dSecurityGroup = "Public"
dDocAuthor = "Jennifer"
dDocAccount = ""
primaryFile = "c:/junk_b.doc"

' In an actual application check for errors.
' Lock the file in order to upload a new revision.
Call checkOutByName(CStr(dDocName))
' If the dDocName is not in the system, it gets added as first revision
' by the CHECKIN_UNIVERSAL call.
Call checkinUniversal(doFileCopy, dDocName, dDocTitle, dDocType, dSecurityGroup,
dDocAuthor, dDocAccount, primaryFile)
End Sub

Private Sub cmdDocInfo_Click()
Call getDocInfo("269")
End Sub

Private Sub cmdDownload_Click()
Call ba
End Sub

Private Sub cmdGetFile_Click()
Call getFile("14", "", "", "")
End Sub

Private Sub cmdGetNativeFile_Click()
Call getNativeFilePath
End Sub

Private Sub Command1_Click()
Dim idccmd As IdcCommandX
Dim res, str
Open "c:\adduser.txt" For Append As #1
Set idccmd = New IdcCommandX
res = idccmd.init("sysadmin", "c:\stellent\bin")
str = "@Properties LocalData" + vbCrLf + "IdcService=ADD_USER" + vbCrLf +
"dName=Jennifer" + vbCrLf + "dFullName=Jennifer Smith" + vbCrLf +
"dPassword=password" + vbCrLf + "dEmail=email@email.com" + vbCrLf +
"dUserAuthType=LOCAL" + vbCrLf + "@end" + vbCrLf + "@ResultSet UserAttribInfo" +
```

```
vbCrLf + "2" + vbCrLf + "dUserName" + vbCrLf + "AttributeInfo" + vbCrLf +
"Jennifer" + vbCrLf + "role,admin,15" + vbCrLf + "@end" + vbCrLf
res = idccmd.executeCommand(str)
Print #1, res
Close #1
End Sub

Private Sub Command2_Click()
Dim idccmd As IdcCommandX
Dim res, str

Dim myRS As String
Dim idcService, tableName As String

idcService = "GET_TABLE"
tableName = "Accounts"

Open "c:\a_getsec.txt" For Append As #1
Set idccmd = New IdcCommandX
res = idccmd.init("sysadmin", "c:\stellent\bin")
str = "@Properties LocalData" + vbCrLf + "IdcService=" + idcService + vbCrLf +
"tableName=" + tableName + vbCrLf + "@end" + vbCrLf
myRS = idccmd.executeCommand(str)

' Parse out the results set.
Call parseResultSet(myRS, tableName)
Print #1, res
Close #1
End Sub

' Calls a private function.
Private Function parseResultSet(strResultsSet As String, strSearchString As
String) As String
Dim indexStop As Integer
Dim tmpStr As String

Dim numberOfRows As Integer
Dim numberOfElementsInSet As Integer
Dim resultElement()
MsgBox (strResultsSet)

' Start of results set.
indexStop = InStr(strResultsSet, "@ResultSet " & strSearchString)

' Check for error (0 index) before moving on.
tmpStr = (Mid(strResultsSet, indexStop))
MsgBox (tmpStr)

' Determine how many data lines are in the HTA file.
```

```
indexStop = InStr(tmpStr, "@end")
For i = 1 To indexStop
If (Mid(tmpStr, i, 1) = Chr(10)) Then
numberOfRows = numberOfRows + 1
End If
Next i
numberOfRows = numberOfRows - 2 ' Remove the first line of data.

' Find first line that identifies the ResultsSet
indexStop = InStr(tmpStr, Chr(10))
tmpStr = (Mid(tmpStr, indexStop + 1))

' Get number of elements in record set, chop the line off the record set...
indexStop = InStr(tmpStr, Chr(10))
numberOfElementsInSet = CInt((Mid(tmpStr, 1, indexStop)))
tmpStr = (Mid(tmpStr, indexStop + 1))

' Set storage array.
ReDim resultElement((numberOfRows / numberOfElementsInSet), numberOfElementsInSet)
Dim junk As String
' Populate array from HTA dataset
For i = 1 To (numberOfRows / numberOfElementsInSet)
For j = 1 To numberOfElementsInSet
indexStop = InStr(tmpStr, Chr(10))
resultElement(i, j) = Mid(tmpStr, 1, indexStop - 1)
tmpStr = (Mid(tmpStr, indexStop + 1))
junk = junk + resultElement(i, j)
Next j
Next i
MsgBox (junk)
parseResultSet = "je"
End Function

' Set storage array.
Sub ba()
Dim b() As Byte 'This byte array will capture the file
Dim strURL As String ' URL string
Dim strDest As String ' Destination File

' Set the strURL to a valid address.
'strURL = "http://localhost/intradoc-
cgi/idc_cgi_isapi.dll?IdcService=GET_FILE&dID=14"
'strDest = "C:\myjunk.html"
strURL = "localhost"
b() = Inet1.OpenURL(strURL, icByteArray)
'Open strDest For Binary Access Write As #1
'Put #1, , b()
'Close #1
End Sub
```

REDIRECTING TEMPLATE PAGE FOR RESPONSE OUTPUT

Sometimes it is desirable to display a page other than the default (change the delivery mechanism for the response template) after executing a CGI request (either a GET or a POST). For example, you might want to redirect the page after a login or after executing a search. One way to do this is by modifying the service call via component architecture, and specifying a different template page. Another more flexible way to do this on the fly is to specify the *urlTemplate* parameter, to redirect the response page to a HCSP or HCST, and reformat the results in any way you wish.

This section covers the following topics:

- Basic Concepts (page 3-18)
- Creating a HCST Page (page 3-18)
- Reformatting the Search Results Page (page 3-19)
- Additional Options (page 3-19)

Basic Concepts

You should be somewhat familiar with IdocScript, and Dynamic Server Pages (HCST, HCSP, HCSF) before attempting this exercise. You should also be somewhat familiar with Component Architecture. It would also be helpful to be familiar with HTML FORM objects.

Creating a HCST Page

As an example, we will create a HCST page that can be used as a URL template to reformat the search results. We could also create a HCSP, but for simplicity, we will use a HCST. We will name the file *test_result.hcst*, and have it contain the following text:

```
<html>
<table width=300>
<tr bgcolor="#000000" style="color: #ffffff;">
    <td><b>Name</b></td>
    <td><b>Title (Author)</b></td>
</tr>
<$loop SearchResults$>
<tr <$if doShade$>bgcolor="#E5E7D4"<$endif$>>
    <td><a href="<$URL$>"><$dDocName$></a></td>
```

```
    <td><$dDocTitle$> (<$dDocAuthor$>)</td>
</tr>
    <$if doShade$><$doShade=""$><$else$><$doShade="1"$><$endif$>
<$endloop$>
</table>
</html>
```

Next, we should check this file into the Content Server. For simplicity, we will check it into the Public security group, with a content type of ADACCT, and the Content ID *test_result*. Its URL will then be something like this:

http://myhost/stellent/groups/public/documents/adacct/test_result.hcst

Reformatting the Search Results Page

To test our new template, we will start by going to a search page. Enter in any search criteria that you wish, and click **Search**. You should see the standard search page with your results contained in it.

Now, add this text to the end of the URL that brought you to the search results page:

&urlTemplate=/stellent/groups/public/documents/adacct/test_result.hcst

You should now see the same search results formatted in a minimalistic HTML page. Note how the full URL is not used, but just the URL relative to your host computer.

If you would like the default search page to always format pages with this template, you can change the HTML FORM object on the search page to also have this field:

```
<input type=hidden name="urlTemplate"
    value="/stellent/groups/public/documents/adacct/test_result.hcst">
```

This can be done by creating a component that modifies the include *query_results_options* to contain the above HTML. Alternatively, the value for *urlTemplate* can be calculated dynamically on the search page with JavaScript, to redirect to different pages based on the metadata entered by the user.

Additional Options

In addition to *urlTemplate*, you can also use the parameters *docTemplateName*, *docTemplateID*, or *RedirectUrl* to change the result page. These all have different behavior, as follows:

- **urlTemplate**—set to the full relative URL of the hcst page you want to use. For example:

```
IdcService=DOC_INFO&urlTemplate=/idc1/groups/public/documents/adacct/test_resu
lt.hcst
```

Because *RedirectURL* doesn't work with all service calls, and pre-6.0 versions of Content Server have minor data pollution bugs with *docTemplateName* and *docTemplateID*, it's usually safest to use *urlTemplate*. However, if you change the Content Type or the Security Group of your template, then the URL will no longer be valid and will need to be updated. Also, this parameter is not recommended for overriding the template used for a "POST" service.

- **docTemplateName**—set to a dDocName of a template (for example, *test_result*). Like *urlTemplate*, but finds the location of the latest released web-viewable for a document with dDocName of *docTemplateName*.
- **docTemplateID**—set to a dID of a specific revision of a template (for example, *100*). Like *docTemplateName*, but finds the web-viewable of a specific dID revision.
- **RedirectUrl**—set to the last part of a CGI URL back into the content server (for example, *IdcService=DOC_INFO&dID=<dID>*). This is only for the few dozen "POST" services that execute the action *prepareRedirect*, such as CHECKIN_NEW and SUBMIT_HTML_FORM.

  By using a redirect after each HTML "POST", the response page can be safely refreshed by the end user without reissuing the post. In the definition of each of these services there is a `3:prepareRedirect:….:0:null` line. The RedirectUrl overrides the results of the prepareRedirect method and allows a different URL to be used as the location for redirects. The RedirectUrl can have Idoc script in it that will be executed just before the redirect is issued. This can create complex Idoc script nesting, because the RediretUrl assignment will typically occur in a resource includes. For example:

```
… Standard Idoc form beginning …
<input type=edit name=myparam value="">
<input type=hidden name=RedirectUrl
value='<$HttpCgiPath$>?<$xml("IdcService=MY_RESPONSE_TEMPLATE&dID=<$dID$>&
myparam=<$myparam$>")$>'>
… Standard Idoc form closure …
```

Much of the Idoc script is nested inside an Idoc literal string. This delays the execution of the script until the redirect URL is being computed. That allows the RedirectUrl to pick up the value of 'myparam' even though the user has still to select its value.

A single quote is used on the outside and a double quote on the inside. This reduces confusion and because both HTML and Idoc script support both single and double quotes for quoting, it is sometimes a good idea to switch between the two for nesting constructs.

Note the usage of the ‘xml’ function. This guarantees that the input field’s value is a well-formed HTML literal string construct. In this particular case, it is not needed. But for more complex constructs it can be helpful.

Chapter

4

CUSTOMIZING SERVICES

OVERVIEW

This chapter discusses the basic structure of services and covers the following topics:

- Service Structure Overview (page 4-1)
- Service Example (page 4-18)
- Creating a Service Resource (page 4-29)

SERVICE STRUCTURE OVERVIEW

This section describes how standard services are implemented in Product_Name. For information about calling services from other programs, see the *IdcCommand Reference Guide*.

A service resource is defined in an HTM file using a ResultSet table with the following three columns:

- Name (page 4-2)
- Attributes (page 4-3)
- Actions (page 4-12)

The standard content server services are defined in the *StandardServices* table in the <*install_dir*>/shared/config/resources/std_services.htm file. You can also find special-purpose services in the *workflow.htm* file in the same directory.

Services depend on other resource definitions to perform their functions.

- Any service that returns HTML requires a template to be specified. A common exception is the PING_SERVER service, which does not return a page to the browser.
- Most services use a query. A common exception is the SEARCH service, which sends a request directly to the search collection.

Merge rules are not required for a service resource. However, the service resource must be listed as a *table* in the ResourceDefinition result set.

The following table row is an example of a service definition:

Figure 4-2 Example of a service definition

<@table StandardServices@>

Scripts For Standard Services

Name	Attributes	Actions
DELETE_DOC	DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)	5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists 3:checkParametersAgainstResultSet:DOC_INFO,dRevClassID,!csRevClassIDMismatch, dDocName,!csDocNameMismatch:0:!csUnableToDeleteItem(dDocName) 3:checkSecurity:DOC_INFO:0:null 3:checkWorkflow:WF_INFO,isNotActiveBasic, DOC_INFO:0:!csUnableToDeleteItem(dDocName)!csItemIsInWorkflow(dWfName) 3:docHistoryInfo:Delete Document,IdocHistory:1:null 5:QrevisionsByClass:REVISIONS:0:!csDeleteUnableToAccessRevList(dDocName) 3:markDocDeleted::0:!csUnableToDeleteItem(dDocName) 3:doWorkflowAction:deleteCriteriaDoc:0:null 3:deleteDoc:REVISIONS:0:null 3:deleteDocumentSubscription:deleteDoc,REVISIONS:8:null 3:setStatusMessage:delete_doc:0:null

<@end@>

NAME

The *Name* column defines the name for each service. For client-side service requests, this is the name called in Calling Services Using Persistent URLs (page 3-6). For standard web requests, this is almost always the URL to the Content Server web page.

Figure 4-3 The *Name* column of the DELETE_DOC service definition

<@table StandardServices@>

Scripts For Standard Services

Name	Attributes	Actions
DELETE_DOC	DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)	5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists 3:checkParametersAgainstResultSet:DOC_INFO,dRevClassID,!csRevClassIDMismatch, dDocName,!csDocNameMismatch:0:!csUnableToDeleteItem(dDocName) 3:checkSecurity:DOC_INFO:0:null 3:checkWorkflow:WF_INFO,isNotActiveBasic, DOC_INFO:0:!csUnableToDeleteItem(dDocName)!csItemIsInWorkflow(dWfName) 3:docHistoryInfo:Delete Document,IdocHistory:1:null 5:QrevisionsByClass:REVISIONS:0:!csDeleteUnableToAccessRevList(dDocName) 3:markDocDeleted::0:!csUnableToDeleteItem(dDocName) 3:doWorkflowAction:deleteCriteriaDoc:0:null 3:deleteDoc:REVISIONS:0:null 3:deleteDocumentSubscription:deleteDoc,REVISIONS:8:null 3:setStatusMessage:delete_doc:0:null

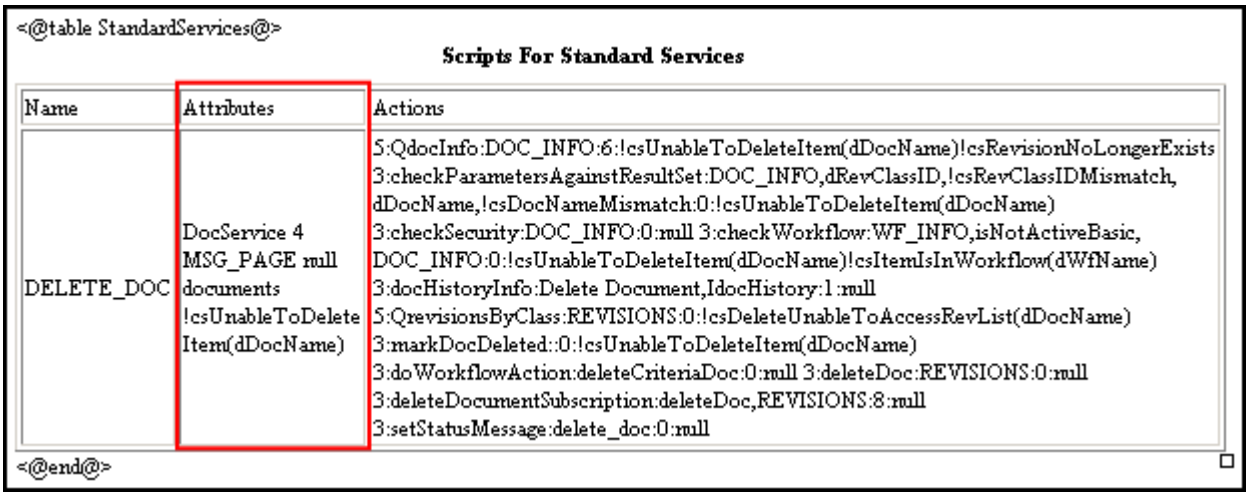
<@end@>

ATTRIBUTES

The *Attributes* column defines aspects of each service, discussed in the following sections:

- Service Class (page 4-4)
- Access Level (page 4-6)
- Template Page (page 4-8)
- Service Type (page 4-9)
- Subjects Notified (page 4-9)
- Error Message (page 4-11)

Figure 4-4 The *Attributes* column of the DELETE_DOC service definition



<@table StandardServices@>

Scripts For Standard Services

Name	Attributes	Actions
DELETE_DOC	DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)	5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists 3:checkParametersAgainstResultSet:DOC_INFO,dRevClassID,!csRevClassIDMismatch, dDocName,!csDocNameMismatch:0:!csUnableToDeleteItem(dDocName) 3:checkSecurity:DOC_INFO:0:null 3:checkWorkflow:WF_INFO,isNotActiveBasic, DOC_INFO:0:!csUnableToDeleteItem(dDocName)!csItemIsInWorkflow(dWfName) 3:docHistoryInfo:Delete Document,IdocHistory:1:null 5:QrevisionsByClass:REVISIONS:0:!csDeleteUnableToAccessRevList(dDocName) 3:markDocDeleted::0:!csUnableToDeleteItem(dDocName) 3:doWorkflowAction:deleteCriteriaDoc:0:null 3:deleteDoc:REVISIONS:0:null 3:deleteDocumentSubscription:deleteDoc,REVISIONS:8:null 3:setStatusMessage:delete_doc:0:null

<@end@>

Service Class

The *service class* attribute specifies the Java class object that the service has access to. The classpath prefix `intradoc.service` is assumed unless a full path is given. The service class determines, in part, what actions can be performed by the service. The possible service classes are:

Service Class	Description
ArchiveService	Performs functions related to archiving.
BatchService	Performs functions related to batch loading.
ChunkedService	Performs functions related to HTTP file chunking for the upload and download of applets.
DocService	Performs actions on documents. Examples are checkin, checkout, document information, and subscription services.
DocProfileService	Performs actions on document profiles, such as adding, editing, and deleting profiles.
FileService	Retrieves files from the content server.

Service Class	Description
IndexerService	Performs functions related to indexing for search engine maintenance.
ListBoxService	Downloads lists from the content server. For example, lists of users, dependent choice lists, and so forth.
LocaleService	Performs functions specific to a user's location or environment (for example, used in internationalization to identify a user's location and provide string files in the appropriate language).
MetaService	Manages metadata fields.
PageHandlerService	Manages Library web pages (created by Web Layout Editor).
PageRequestService	Retrieves an HTML page.
ProjectService	Manages Publisher projects.
ProviderManagerService	Manages providers (an Application Programming Interface, or API, that establishes connection to outside entities).
SchemaService	Manages the server-side publishing of JavaScript files of database tables, such as option lists.
SearchService	Performs functions related to searching.
Service	Performs a general service.
UserService	Manages users.
WorkflowService	Manages workflows.
WorkflowTemplateService	Manages workflow templates.

Service Class	Description
intradoc.admin.AdminService	Performs functions through the Admin Server. Generally called internally by the content server itself. These services are very complicated, and failing to call them correctly can result in the loss or corruption of content server data. Therefore, it is strongly recommends that you ***do not*** use or modify these services.

Example

In the example of the DELETE_DOC service, the service class is *DocService*:

```
DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)
```

Access Level

The service security model is similar to the document security model used throughout Content Server. The *access level* attribute assigns permission levels to the service. Any user attempting to execute the service must have at least this permission.

Security access is stored as bit-flags. The access level number is the sum of the following possible bit flags:

Bit Flag	Permission	Description
1	READ_PRIVILEGE	Read permission is required for the security group referenced in the service.
2	WRITE_PRIVILEGE	Write permission is required for the security group referenced in the service.
4	DELETE_PRIVILEGE	Delete permission is required for the security group referenced in the service.
8	ADMIN_PRIVILEGE	Admin permission is required for the security group referenced in the service.

Bit Flag	Permission	Description
16	GLOBAL_PRIVILEGE	The service calls the global security check to determine if the current user has permission to execute the service. The check validates if the *admin* role is required or if the user only needs a given permission (Read, Write, or Delete) on at least one security group.
32	SCRIPTABLE_SERVICE	Scriptable services don't require parameter input, so they can be called with the *executeService* function on dynamic server pages.

If a service is acting on a document, the user must have READ, WRITE, DELETE, or ADMIN permission (in that order) for that document's security group to execute the service. For example, to subscribe to a document the user only needs READ permission for that document's security group. However, to check in a new document the user would also need WRITE permission for that document's security group.

If the service does not act on a specific document (such as GET_USER_INFO, CHECKIN_NEW_FORM, and so forth), the GLOBAL_PRIVILEGE bit flag should be set along with at least one more permission bit flag. This means that the user must have that level of permission in at least one security group in order to execute the service.

Note: A service should never just specify the GLOBAL_PRIVILEGE bit flag alone. At least one more permission bit flag should be specified.

SCRIPTABLE_SERVICE permission means that the service can be executed via the *executeService* IdocScript function. This should be restricted to read-only services, such as GET_SEARCH_RESULTS, GET_USER_INFO, and so forth.

The following is a complete list of all access levels and their meanings:

- 0: no access allowed
- 1: Read permission required
- 2: Write permission required
- 3: Read/write permission required
- 4: Delete permission required

- 8: Admin permission required
- 16: Global permission required
- 17: Global and read permission required
- 18: Global and write permission required
- 19: Global and read/write permission required
- 23: Global, read/write/delete permission required
- 24: Global, admin permission required
- 32: Scriptable permission required
- 33: Scriptable and read permission required
- 34: Scriptable and write permission required
- 40: Scriptable and admin permission required
- 49: Scriptable and global, read permission required
- 50: Scriptable, global, write permission required
- 51: Scriptable, global, read/write permission required
- 56: Scriptable, global, admin permission required

Examples

In the example of the DELETE_DOC service, the access level is 4, meaning that the user must have DELETE_PRIVILEGE to execute the service:

```
DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)
```

As another example, the access level for the ADD_ALIAS service is 24, meaning that the user must have ADMIN_PRIVILEGE and GLOBAL_PRIVILEGE to execute the service:

```
ADD_ALIAS UserService 24 null null aliases !csUnableToAddAlias
```

Template Page

The *template page* attribute specifies the template that displays the results of the service. If the results of the service do not require presentation (such as the *PageHandlerService* type), this attribute is *null*.

Templates are a combination of HTML and IdocScript. The IdocScript is used to format the HTML and display the data in the response. The template page name is mapped to an HTM file in the <*install_dir*>/custom/Folders/resources/templates/templates.hda file:

- Most template pages are mapped in the *IntradocTemplates* ResultSet.
- Search template pages are mapped in the *SearchResultTemplates* ResultSet.

Example

In the example of the DELETE_DOC service, the template page that presents the results of the service is *MSG_PAGE*, which is mapped to the *msg_page.htm* file:

```
DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)
```

Service Type

The *service type* attribute specifies if the service is to be executed as a *subservice* inside another service. You cannot call a second service from a main service unless the second service is a *SubService*.

Service Type	Description
SubService	The service is a subservice that is executed only inside another service.
null	The service is not a subservice.

For example, the UPDATE_DOCINFO service executes the UPDATE_DOCINFO_SUB, which has the service type of *SubService*.

Example

In the example of the DELETE_DOC service, the service is not a subservice, so the service type is *null*:

```
DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)
```

Subjects Notified

The *subjects notified* attribute specifies the subjects (subsystems) to be notified by the service. If a service changes one or more subjects, it must notify the remote sources (such as database tables) that cached information has been updated.

For example, if you do an *IsJava=1* call for any service, you will always see *changedSubjects* and *refreshSubjects* in the response. These subjects are used to notify the client when the state of the content server has changed. For example, they notify the client

when a new user is added to the system, when a new document has been checked in, when custom metadata has been changed, and so on. This allows external applications to refresh their data when the content server state changes. It is also the underlying mechanism behind keeping the content server administration applets up-to-date with the number of users, document types, and documents in the system. For example, if you launch the Repository Manager and then check in a new document, you will (in a few seconds) see that item appear in the applet.

The subjects notified string is a comma-separated list of changed subjects. (If no subjects are notified, this attribute is *null*.) For example, the value of the subjects notified attribute for the EDIT_METADEF service is *metadata,dynamicqueries*. This service modifies a metadata field, and subsequently informs the system that the *metadata* and *dynamicqueries* subjects have changed.

Possible subjects are:

Subject	**Must be notified of changes to:**
accounts	Predefined accounts
aliases	User aliases
collaborations	Collaboration Server projects
collections	Archiver collections
config	Global configuration information
docformats	File formats
doctypes	Content Types
documents	New content items, revised content items, or updated content item metadata
dynamicqueries	Dynamic queries that retrieve the list of metadata fields from the database
indexerstatus	Indexer status
indexerwork	Content items; specifies that an indexing update cycle is required
metadata	Metadata fields
metaoptlists	Metadata field option lists

Subject	Must be notified of changes to:
pagelist	Library (Web Layout Editor) pages
projects	Registered Content Publisher projects
renditions	Additional renditions (from XML Converter, Thumbnails, and so forth)
reports	Report data sources
schema	Schema definitions
searchapi	Connection to the indexing search engine
subscriptiontypes	Subscription types
templates	Search result templates (which are configured from Web Layout Editor)
userlist	Users; specifies the list of users for Collaboration Server
usermetaoptlists	User information field option lists
users	User information
wfscripts	Workflow event scripts
wftemplates	Workflow templates
workflows	Workflows

Example

In the example of the DELETE_DOC service, the *documents* subject is the only subject notified:

```
DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)
```

Error Message

The *error message* attribute defines the error message that is returned by the service if no action error message overrides it. This can be either an actual text string or a reference to a locale-sensitive string. For more information, see *Getting Started with the Software Developer's Kit (SDK)*.

Example

In the example of the DELETE_DOC service, the error message is a localized string:

```
DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)
```

ACTIONS

The *Actions* column defines one or more steps taken to process the service. An *action* is an operation to be performed as part of a service script. Actions can execute SQL statements, perform a query, run code, cache the results of a query, or load an option list. The data returned by one action can alter the behavior of later actions.

An action is defined as a list of colon-separated segments, using the following format:

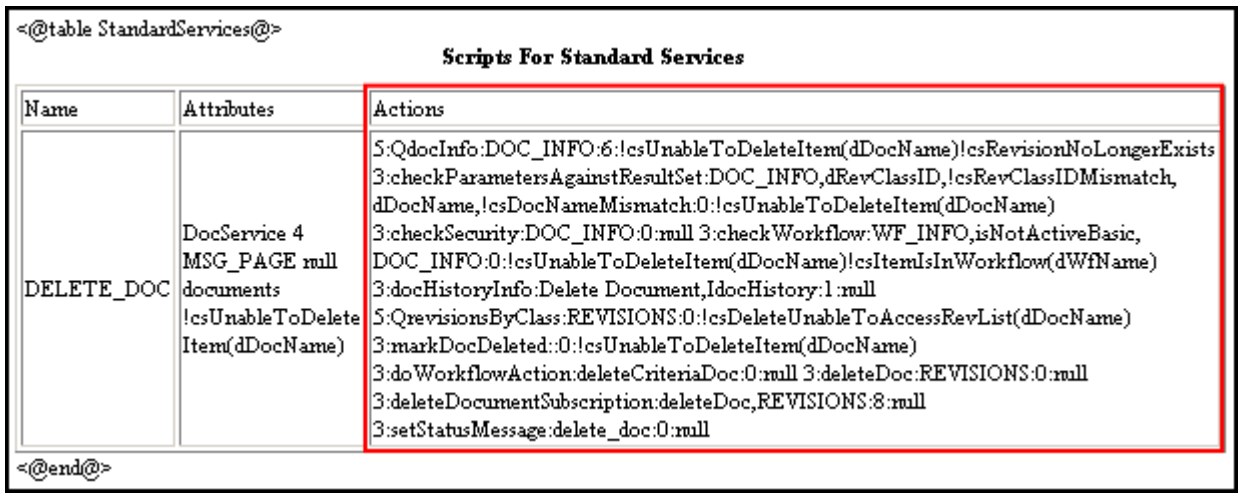
```
type:name:parameters:control mask:error message
```

See the following sections for more information:

- Action Type (page 4-13)
- Action Name (page 4-14)
- Action Parameters (page 4-15)
- Action Control Mask (page 4-15)
- Action Error Message (page 4-18)

Note: In an HTM resource file that defines services, the `<br>` tags in the *Actions* column are for browser display purposes only, so they are optional. However, the `</td>` tag must occur immediately after the list of actions, without a line break in between.

Figure 4-5 The *Actions* column of the DELETE_DOC service definition



<@table StandardServices@>

Scripts For Standard Services

Name	Attributes	Actions
DELETE_DOC	DocService 4 MSG_PAGE null documents !csUnableToDeleteItem(dDocName)	5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists 3:checkParametersAgainstResultSet:DOC_INFO,dRevClassID,!csRevClassIDMismatch, dDocName,!csDocNameMismatch:0:!csUnableToDeleteItem(dDocName) 3:checkSecurity:DOC_INFO:0:null 3:checkWorkflow:WF_INFO,isNotActiveBasic, DOC_INFO:0:!csUnableToDeleteItem(dDocName)!csItemIsInWorkflow(dWfName) 3:docHistoryInfo:Delete Document,IdocHistory:1:null 5:QrevisionsByClass:REVISIONS:0:!csDeleteUnableToAccessRevList(dDocName) 3:markDocDeleted::0:!csUnableToDeleteItem(dDocName) 3:doWorkflowAction:deleteCriteriaDoc:0:null 3:deleteDoc:REVISIONS:0:null 3:deleteDocumentSubscription:deleteDoc,REVISIONS:8:null 3:setStatusMessage:delete_doc:0:null

<@end@>

Action Type

The first segment of an action statement defines the type of action.

- **Action Type #**—used to identify the action type in the service resource file.
- **Component Wizard Identifier**—used to identify the action type in the Component Wizard. In the service resource file, the action type is represented as a number.
- **Java Constant**—used to identify the action type in Content Server Java code.

The possible action types are:

Action Type #	Component Wizard Identifier	Java Constant	Description
1	Select query	QUERY_TYPE	Executes a predefined SQL database query to retrieve information (read-only action) and then immediately discards the results. For example, a select query might be used to see if a specific dDocName (Content ID) already exists in the database. The query is specified by the Action Name (page 4-14) of the action.
2	Execute query	EXECUTE_TYPE	Executes a predefined SQL database query to delete, add, or update information in the database.

Action Type #	Component Wizard Identifier	Java Constant	Description
3	Java method	CODE_TYPE	Specifies a code module that is a part of the Java class implementing the service.
4	Load option list	OPTION_TYPE	Loads an option list stored in the system. This is a deprecated action type.
5	Select cache query	CACHE_RESULT_TYPE	Executes an SQL database query to retrieve information (read-only action) and then stores the results for later use. For example, a select cache query might be used to find all users subscribed to a content item and save the list for display to consumers. The query is specified by the Action Name (page 4-14) of the action.

Example

In the example of the first action of the DELETE_DOC service, the action is a *Select cache query* type (*5*):

```
5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists
```

Action Name

The second segment of an action statement defines the name of the action.

- For the *Java method* action type, the action name is the Java method.
- For the *Load option list* action type, the action name is the option list.
- For the *Select query*, *Execute query*, and *Select cache query* action types, the action name is the query name. For standard content server services, Oracle typically uses a prefix to identify the action performed on the database. The possible prefixes for queries are:

Query Prefix	Description
Q	Query (retrieve) information from the database (read-only action).
D	Delete information from the database.

Query Prefix	Description
I	Insert (add) information in the database.
U	Update information in the database.

Example

In the example of the first action of the DELETE_DOC service, the name of the action is *QdocInfo*. This specifies that a read-only query (*Q*) will be performed on the database:

```
5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists
```

Action Parameters

The third segment of an action statement specifies the parameters that the action requires. If no parameters are required, this segment is left empty (two colons appear in place of the parameters).

- If the action requires parameters, enter the parameters as a comma-separated list.
- For the *Select cache query* action type, the first parameter is the name that the action assigns to the ResultSet returned from the query. This ResultSet can then be referenced in the template page.
- For the *Load option list* action type, the parameters are optional. However, if parameters are given, the first parameter is the key under which the option list is loaded, and the second parameter is the selected value for display on an HTML page.

Example

In the example of the first action of the DELETE_DOC service, the ResultSet that is returned from the service query will be named *DOC_INFO*:

```
5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists
```

Action Control Mask

The fourth segment of an action statement is an optional bit flag that controls the results of queries to the content server database.

- **Control Mask Bit Flag**—used to identify the control mask in the service resource file. The control mask number used in the service resource file represents the sum of

the bit flags for all controls being applied. For standard content server services, the control mask bit flag is typically used instead of the control mask string value.

- **Control Mask String Value**—used to identify the control mask in the service resource file. For multiple control masks, the string values are placed in a comma-separated list. For custom services, the string value is used instead of the control mask bit flag.
- **Component Wizard Identifier**—used to identify the control mask in the Component Wizard. In the service resource file, the control mask is represented by either the sum of its bit flags (standard content server services), or a comma-separated list of bit flag string values (custom services).
- **Java Constant**—used to identify the control mask in the Content Server Java code.

The possible control masks are:

Control Mask Bit Flag	Control Mask String Value	Component Wizard Identifier	Java Constant	Description
0	-	-	-	No control is applied.
1	ignoreError	Ignore error	CONTROL_IGNORE_ERROR	Do not abort the service on error.
2	mustExist	Check result non-empty	CONTROL_MUST_EXIST	At least one record must be returned by the query, or the action fails. Used only for the *Select query* and *Select cache query* action types.
4	beginTran	Begin transaction	CONTROL_BEGIN_TRAN	Starts a database transaction.
8	commitTran	Commit transaction	CONTROL_COMMIT_TRAN	Concludes a database transaction.
16	mustNotExist	Check result empty	CONTROL_MUST_NOT_EXIST	Query must not return any rows, or the action fails. Used only for the *Select query* and *Select cache query* action types.

Control Mask Bit Flag	Control Mask String Value	Component Wizard Identifier	Java Constant	Description
32	retryQuery	Retry query	CONTROL_RETRY QUERY	When the action type is set to *Execute query* and the query fails, the error will be logged. After three seconds, the query will be run again. If the query fails three times, the service will fail. Used mostly for checking in content. No longer used.
64	doNotLog	Do not log	CONTROL_DO_NOT_LOG	When the action type is set to *Select query* and the query fails, the service will fail. However, the error will NOT be logged in the system logs. Used internally by Oracle when developing components as a debug flag for testing.

Note: The *Check result non-empty* and *Check result empty* control masks are used only for the *Select query* and *Select cache query* action types. See Action Type (page 4-13).

Examples

In the example of the first action of the DELETE_DOC service, the control mask value is *6*, which means that at least one record must be returned by the query (*2*), and the action starts a database transaction (*4*):

```
5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists
```

If this was a custom service created using the Component Wizard, the sum of the control mask bit flags would be replaced by a comma-separated list of the bit flag string values:

```
5:QdocInfo:DOC_INFO:mustExist, beginTran:!csUnableToDeleteItem(dDocName)
!csRevisionNoLongerExists
```

Action Error Message

The fifth segment of an action statement defines the error message to be displayed if this action fails. This can be either an actual text string or a reference to a locale-sensitive string. For more information, see *Getting Started with the Software Developer's Kit (SDK)*.

- An action error message overrides the error message provided as an attribute of the service.
- If the error message for an action is not null, it becomes the error message for the remainder of the actions in the service.
- If the error message for an action is null, the error message remains unchanged from the previous action.
- String references are preceded by an exclamation point.

Example

In the example of the first action of the DELETE_DOC service, the error message is a combination of two localized strings:

```
5:QdocInfo:DOC_INFO:6:!csUnableToDeleteItem(dDocName)!csRevisionNoLongerExists
```

SERVICE EXAMPLE

The DOC_INFO service provides a good example of how services, queries, and templates work together. This section covers the following topics:

- DOC_INFO Service Definition (page 4-19)
- DOC_INFO Attributes (page 4-20)
- DOC_INFO Actions (page 4-21)
- DOC_INFO Template (page 4-26)

DOC_INFO Service Definition

The following figures show the DOC_INFO service definition from the <*install_dir*>/config/resources/std_services.htm file:

Figure 4-6 Example of the DOC_INFO service definition, displayed in a text editor and web browser.

```
<@table StandardServices@>
<table border=1><caption><strong>Scripts For Standard Services</strong></caption>
<tr>
    <td>Name</td><td>Attributes</td><td>Actions</td>
</tr>
<tr>
    <td>DOC_INFO</td>
    <td>DocService
        33
        DOC_INFO
        null
        null<br>
        !csUnableToGetRevInfo</td>
    <td>5:QdocInfo:DOC_INFO:2:!csItemNoLongerExists2
        3:mapNamedResultSetValues:DOC_INFO,dStatus,dStatus,dDocTitle,dDocTitle:0:null
        3:checkSecurity:DOC_INFO:0:!csUnableToGetRevInfo2(dDocName)
        3:getDocFormats:QdocFormats:0:null
        3:getURLAbsolute::0:null
        3:getUserMailAddress:dDocAuthor,AuthorAddress:0:null
        3:getUserMailAddress:dCheckoutUser,CheckoutUserAddress:0:null
        3:getWorkflowInfo:WF_INFO:0:null
        3:getDocSubscriptionInfo:QisSubscribed:0:null
        5:QrevHistory:REVISION_HISTORY:0:!csUnableToGetRevHistory(dDocName)</td>
</tr>
</table>
<@end@>
```

<@table StandardServices@>

Scripts For Standard Services

Name	Attributes	Actions
DOC_INFO	DocService 33 DOC_INFO null null !csUnableToGetRevInfo	5:QdocInfo:DOC_INFO:2:!csItemNoLongerExists2 3:mapNamedResultSetValues:DOC_INFO,dStatus,dStatus,dDocTitle,dDocTitle:0:null 3:checkSecurity:DOC_INFO:0:!csUnableToGetRevInfo2(dDocName) 3:getDocFormats:QdocFormats:0:null 3:getURLAbsolute::0:null 3:getUserMailAddress:dDocAuthor,AuthorAddress:0:null 3:getUserMailAddress:dCheckoutUser,CheckoutUserAddress:0:null 3:getWorkflowInfo:WF_INFO:0:null 3:getDocSubscriptionInfo:QisSubscribed:0:null 5:QrevHistory:REVISION_HISTORY:0:!csUnableToGetRevHistory(dDocName)

<@end@>

DOC_INFO Attributes

The following table describes the attributes of the DOC_INFO service:

Attribute	Value	Description
Service class	DocService	This service is providing information about a content item.
Access level	33	1 = The user requesting the service must have Read permission on the content item. 32 = This service can be executed with the *executeService* Idoc Script function.
Template page	DOC_INFO	This service uses the DOC_INFO template (*doc_info.htm* file). The results from the actions will be merged with this template and presented to the user.
Service type	null	This service is not a subservice.
Subjects notified	null	No subjects are affected by this service.
Error Message	!csUnableToGetRevInfo	If this service fails on an English content server system, it returns the error message string: *Unable to retrieve information about the revision*

DOC_INFO Actions

The DOC_INFO service executes the following ten actions:

Action Definition	Description
Action 1 `5:QdocInfo:DOC_INFO:2:!csItemNoLongerExists2`	
5	Select cache query action that retrieves information from the database using a query.
QDocInfo	This action retrieves content item information using the *QDocInfo* query in the *query.htm* file.
DOC_INFO	The result of the query is assigned to a ResultSet called DOC_INFO and stored for later use.
2	The Check result non-empty control mask specifies that the query must return a record, or the action fails.
!csItemNoLongerExists2	If this action fails on an English content server system, it returns the error message string: *This content item no longer exists*
Action 2 `3:mapNamedResultSetValues:DOC_INFO,dStatus,dStatus,dDocTitle,dDocTitle:0:null`	
3	Java method action specifying a module that is a part of the Java class implementing the service.
mapNamedResultSetValues	This action retrieves the values of *dStatus* and *dDocTitle* from the first row of the DOC_INFO ResultSet and stores them in the local data. (This increases speed and ensures that the correct values are used.)
DOC_INFO,dStatus,dStatus, dDocTitle,dDocTitle	Parameters required for the *mapNamedResultSetValues* action.
0	No control mask is specified.
null	No error message is specified.

Action Definition	Description
Action 3 `3:checkSecurity:DOC_INFO:0:!csUnableToGetRevInfo2(dDocName)`	
3	Java method action specifying a module that is a part of the Java class implementing the service.
checkSecurity	This action retrieves the data assigned to the DOC_INFO ResultSet and evaluates the assigned security level to verify that the user is authorized to perform this action.
DOC_INFO	ResultSet that contains the security information to be evaluated by the *checkSecurity* action.
0	No control mask is specified.
!csUnableToGetRevInfo2(dDocName)	If this action fails on an English content server system, it returns the error message string: *Unable to retrieve information for "{dDocName}."*
Action 4 `3:getDocFormats:QdocFormats:0:null`	
3	Java method action specifying a module that is a part of the Java class implementing the service.
getDocFormats	This action retrieves the file formats for the content item using the *QdocFormats* query in the *query.htm* file. A comma-separated list of the file formats is stored in the local data as *dDocFormats*.
QdocFormats	Specifies the query used to retrieve the file formats.
0	No control mask is specified.
null	No error message is specified.

Action Definition	Description
Action 5 `3:getURLAbsolute::0:null`	
3	Java method action specifying a module that is a part of the Java class implementing the service.
getURLAbsolute	This action resolves the URL of the content item and stores it in the local data as *DocUrl*.
blank	This action takes no parameters.
0	No control mask is specified.
null	No error message is specified.
Action 6 `3:getUserMailAddress:dDocAuthor,AuthorAddress:0:null`	
3	Java method action specifying a module that is a part of the Java class implementing the service.
getUserMailAddress	This action resolves the e-mail address of the content item author.
dDocAuthor,AuthorAddress	This action passes *dDocAuthor* and *AuthorAddress* as parameters.
0	No control mask is specified.
null	No error message is specified.

Action Definition	Description
Action 7 `3:getUserMailAddress:dCheckoutUser,CheckoutUserAddress:0:null`	
3	Java method action specifying a module that is a part of the Java class implementing the service.
getUserMailAddress	This action resolves the e-mail address of the user who has the content item checked out.
dCheckoutUser, CheckoutUserAddress	This action passes *dCheckoutUser* and *CheckoutUserAddress* as parameters.
0	No control mask is specified.
null	No error message is specified.
Action 8 `3:getWorkflowInfo:WF_INFO:0:null`	
3	Java method action specifying a module that is a part of the Java class implementing the service.
getWorkflowInfo	This action evaluates whether the content item is part of a workflow. If the WF_INFO ResultSet exists, workflow information is merged into the DOC_INFO template.
WF_INFO	This action passes WF_INFO as a parameter.
0	No control mask is specified.
null	No error message is specified.

Action Definition	Description
Action 9 `3:getDocSubscriptionInfo:QisSubscribed:0:null`	
3	Java method action specifying a module that is a part of the Java class implementing the service.
getDocSubscriptionInfo	This action evaluates if the current user has subscribed to the content item: • If the user is subscribed, an **Unsubscribe** button is displayed. • If the user is not subscribed, a **Subscribe** button is displayed.
QisSubscribed	Specifies the query used to retrieve the subscription information.
0	No control mask is specified.
null	No error message is specified.
Action 10 `5:QrevHistory:REVISION_HISTORY:0:!csUnableToGetRevHistory(dDocName)`	
5	Select cache query action that retrieves information from the database using a query.
QrevHistory	This action retrieves revision history information using the *QrevHistory* query in the *query.htm* file.
REVISION_HISTORY	The result the query is assigned to a ResultSet called REVISION_HISTORY. The DOC_INFO template loops on this ResultSet to present information about each revision.
0	No control mask is specified.
!csUnableToGetRevHistory (dDocName)	If this action fails on an English content server system, it returns the error message string: *Unable to retrieve revision history for "{dDocName}."*

DOC_INFO Template

The template page for the DOC_INFO service is the DOC_INFO template. It is important to know what is happening between the files so that you can understand the interactions between the template page and the actions performed in a service.

The definition for the content that the doc_info.htm template contains is located in the <*install_dir*>/custom/Folders/resources/std_page.htm file. Code from both files appear in the following markup section:

Markup from the <*install_dir*>/shared/config/templates/doc_info.htm file:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
<html>
<head>
    <$include std_info_html_head_declarations$>
</head>
<$include info_body_def$>
<$include info_page_content$>
</body>
</html>
```

Markup from the <*install_dir*>/custom/Folders/resources/std_page.htm file that defines what will appear in the doc_info.htm template:

```
<@dynamichtml info_page_content@>
<$include std_page_begin$>
<$include std_header$>
…
<!-- Do a loop on DOC_INFO so that all substitution tags will use DOC_INFO as
their first place to find their values.  Otherwise their is confusion between this
result set and the REVISION_HISTORY table that comes later.  For example 'dStatus'
is a value in both tables-->
<$loop DOC_INFO$>
<$if AllowPrimaryMetaFile and isTrue(AllowPrimaryMetaFile) and
isTrue(dFormat like "*idcmeta*")$>
<$showPrimaryMetaFileFields = "1"$>
<$endif$>
<$include doc_info_notify_data$>

<table border=0 cellpadding=2 cellspacing=0 width=<$docInfoWidth-30$>>
<caption align=top><h4 class=pageTitle><$pageTitle$></caption>
<$include special_checkin_fields1$>
<$include std_revision_label_field$>
<$include std_document_type_field$>
<$include std_document_title_field$>
<$include author_checkin_field$>
<$include std_meta_fields$>
<$include security_checkin_fields$>
```

```
<$include checkout_author_info_field$>
<$if IsStagingDoc$>
<$include doc_date_fields$>
<$endif$>
<$fieldName = "dStatus", fieldCaption = "Status"$><$include
std_displayonly_field$>
<$if HasOriginal$>
<$fieldName = "dDocFormats", fieldCaption = "Formats"$><$include
std_display_field$>
<$endif$>
<$include workflow_list_for_doc$>
<$if HasUrl$>
<$include doc_url_field$>
<$endif$>
<$if HasOriginal and not ClientControlled and not showPrimaryMetaFileFields$>
<$fieldName = "dOriginalName", fieldCaption = "Get Native File"$>
<$if DownloadApplet$>
<$valueStyle="xxsmall", fieldValue =
strTrimWs(inc("download_file_by_applet_form_content"))$>
<$else$>
<$fieldValue = strTrimWs(inc("doc_file_get_copy"))$>
<$endif$>
<$if DownloadApplet$><form name=downloadForm><$endif$>
<$include std_displayonly_field$>
<$if DownloadApplet$></form><$endif$>
<$endif$>
<$if IsFailedConversion or IsFailedIndex or IsDocRefinePassthru$>
<$if IsFailedConversion$><$include std_namevalue_separator$><$endif$>
<tr>
<td align=right><span class=errorHighlight>
<$if IsFailedIndex$>Index Error:
<$else$>Conversion Error:
<$endif$></span></td>
<td>
<table>
<tr>
<td><span class=tableEntry>
<$dMessage$>
<$if IsFailedIndex$>
<br>Content has been indexed with Info only.
Resubmit should only be performed if the problem has been resolved.
<$elseif IsDocRefinePassthru$>
<br>Content Refinery failed to convert the content item but released it to the
web by copying the native file.
<$endif$></span></td>
<td><form action="<$HttpCgiPath$>" method="POST">
<input type=hidden name=dID value="<$dID$>">
<input type=hidden name=dDocName value="<$dDocName$>">
<input type=hidden name=IdcService value="RESUBMIT_FOR_CONVERSION">
```

```
<input type=submit value= "Resubmit ">
<$if ClientControlled$>
<input type=hidden name=ClientControlled value="DocMan">
<$endif$>
</form></td>
</tr>
</table>
</td>
</tr>
<$if IsFailedConversion$><$include std_namevalue_separator$><$endif$>
<$endif$>
</table>
<$if IsNotSyncRev$>
<table width="100%">
<tr>
<td align=center><span class=errorHighlight>The local copy of this content item
has
not been updated to the latest revision.  Use <i>Get Native File</i> or <i>Check
out</i>
to update your local copy of <i><$dDocName$></i>.</span></td>
</tr>
</table>
<$endif$>

<$if IsStagingDoc$>
<br>
<table width="90%">
<tr>
<td width="20%" align=center><$include doc_problem_reports$></td>
<td width="20%" align=center><$include project_problem_reports$></td>
</tr>
</table>
<$include doc_provider_info$>
<$else$>
<table width="90%">
<tr>
<$if ClientControlled$>
<td width="20%" align=center><$include doc_select_actions$></td>
<$else$>
<td width="20%" align=center><$include doc_file_undo_checkout$></td>
<td width="20%" align=center><$include doc_file_checkout$></td>
<td width="20%" align=center><$if showPrimaryMetaFileFields$><$include
meta_file_update$>
<$else$><$include doc_file_update$><$endif$></td>
<$endif$>
<td width="20%" align=left><$include doc_subscription_unsubscription$></td>
<$if ClientControlled$>
<td width="20%"></td>
<td width="20%"></td>
```

```
<$endif$>
</tr>
</table>
<$endif$>
<$if HasOriginal and DownloadApplet$>
<$include download_native_applet$>
<$endif$>

<!-- end loop on DOC_INFO-->
<$endloop$>
<$if IsStagingDoc$>
<!-- present a problem report form -->
<$include doc_add_problem_report$>
<$else$>
<!-- Table holding information about all revisions of this document-->
<$include doc_rev_table$>
<$endif$>
</td>
</tr>
</table>
<$include std_page_end$>
<@end@>
```

CREATING A SERVICE RESOURCE

There are two ways to create a service resource for use with a custom component:

- Creating a Custom Service Manually (page 4-29)
- Creating a Custom Service using Component Wizard (page 4-32)

Note: For more information about custom components, see the *Working with Components* guide.

Creating a Custom Service Manually

To create a custom service resource manually:

1. Define the service in an HTM file (page 4-30).
2. Load the service in the custom component HDA file (page 4-32).

Define the service in an HTM file

The HTM file must include a table that is identical in structure to the *StandardServices* table. See Service Structure Overview (page 4-1).

a. Make a copy of the *std_services.htm* file, place it in your custom component's *resources* directory, and rename the file to avoid confusion. For example:

```
/custom/my_component/resources/my_services.htm
```

3. Change the name of the *StandardServices* table to a new name. For example:

```
<@table MyServices@>
```

4. Delete all of the rows in the table except for a service that is similar to the one you want to create.
5. Edit the entries in the Name, Attributes, and Actions columns.
6. Save and close the file.

For example, the following HTM file shows two custom services named ADD_REPORT and REPORTS_LIST:

Figure 4-7 Example of custom services HTM file, displayed in a text editor.

```
<HTML>
<HEAD>
<META HTTP-EQUIV='Content-Type' content='text/html; charset=iso-8859-1'>
<TITLE>Custom Scripted Services</TITLE>
</HEAD>
<BODY>
<@table MyServices@>
<table border=1><caption><strong>Scripts For Custom Services
</strong></caption>
<tr>
<td>Name</td><td>Attributes</td><td>Actions</td>
</tr>
<tr>
<td>ADD_REPORT</td>
<td>Service
18
ADD_REPORT_FORM
null
null<br>
Unable to add report.</td>
<td>2:Ireport::0:null</td>
</tr>
<tr>
<td>REPORTS_LIST</td>
<td>Service
17
REPORT_LIST_FORM
null
null<br>
Unable to retrieve reports.</td>
<td>5:Qreports:REPORT_LIST:0:null</td>
</tr>
</table>
<@end@>
<br><br>
</BODY>
</HTML>
```

Figure 4-8 Example of custom services HTM file, displayed in a web browser

<@table MyServices@>

Scripts For Custom Services

Name	Attributes	Actions
ADD_REPORT	Service 18 ADD_REPORT_FORM null null Unable to add report.	2:Ireport::0:null
REPORTS_LIST	Service 17 REPORT_LIST_FORM null null Unable to retrieve reports.	5:Qreports:REPORT_LIST:0:null

<@end@>

Load the service in the custom component HDA file

1. Open the component definition (glue) file of the custom component in a text editor. For example, <*install_dir*>/custom/my_component/my_component.hda.
2. Add the new HTM file to the *ResourceDefinition* ResultSet. For example:

```
@ResultSet ResourceDefinition
4
type
filename
tables
loadOrder
service
resources/my_services.htm
MyServices
1
@end
```

3. Save the file.

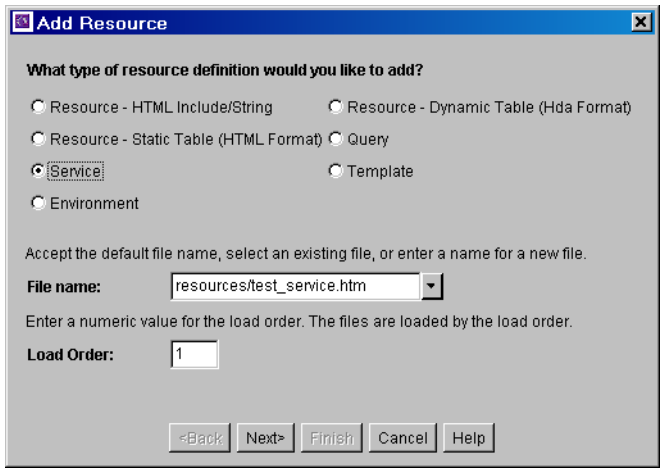
Creating a Custom Service using Component Wizard

To create a service resource using the Component Wizard:

1. In the Component Wizard, open the component the resource will be created for.
2. On the Resource Definition tab, click **Add**.

   The Add Resource screen is displayed.

Figure 4-9 Defining a new custom service



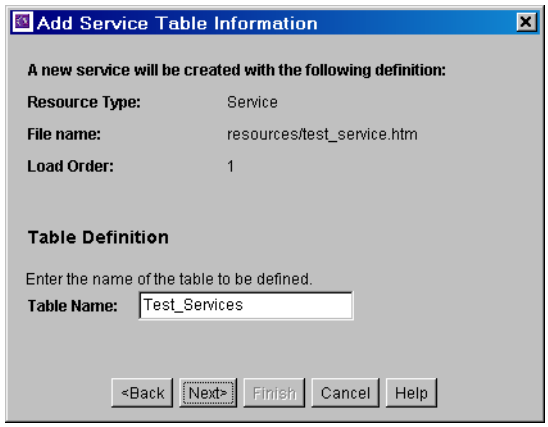
3. Select the **Service** option.
4. Enter the file name for the resource file. The default file name is *resources/componentname_service.htm*.
   - ❖ If a resource file has been created for services, you can append the new service table to the existing file by selecting the file name. Any changes you make to the load order at this time will apply to the entire resource file.
   - ❖ To create a new resource file with a different file name, enter the file name. For example, *my_services.htm*.
5. If you want the new resource file to be loaded in a particular order, enter the number in the **Load Order** field.

Note: Unless you have a particular reason for the resource file to be loaded after other resources, you should leave the load order set to 1.

6. Click **Next**.

   The Add Service Table Information screen is displayed.

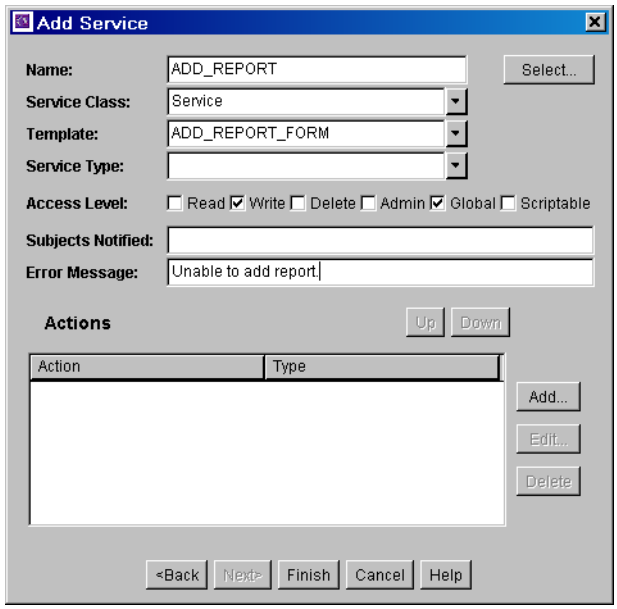
Figure 4-10 Defining a new service table



7. Enter a name for the service table.
   - ❖ It is a good idea to leave the name of the component as a prefix. For example, *My_Component_Services*.
   - ❖ Each service table in a component must have a unique name, even if the tables are in different resource files.
8. Click **Next**.

   The Add Service screen is displayed.

Figure 4-11 Defining service attributes.



9. Enter the service attributes directly, or start with an existing service definition as follows:
   a. Click **Select**.

      A list of commonly used services is displayed.

   b. Select the **Show All** check box to show the entire list of predefined services.

   c. Select a service from the list. To view details about a service, highlight the service name and click **Preview**.

Tech Tip: To view the online help for the selected service, click the **Help** button on the Preview Information for Service <*SERVICE_NAME*> dialog.

   d. Click **OK**.

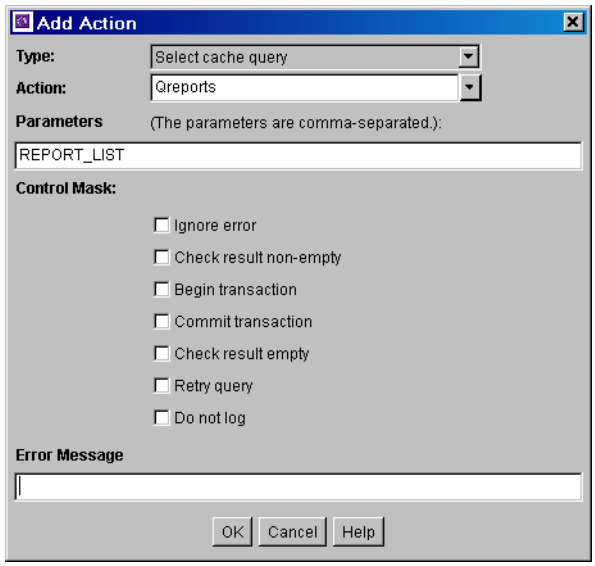
      The service attributes and actions are filled in.

Note: If you do not change the name of the service and this component is loaded last, the custom service will override the standard service and any other custom services with the same name.

   e. Edit the service attributes as necessary.

10. Enter the actions as necessary.
    - ❖ Actions must appear in the Actions list in order of execution. Use the **Up** and **Down** buttons to move the selected action.
    - ❖ To add an action, click **Add**. Enter the action definition and click **OK**.
    - ❖ To edit an action, select the action and click **Edit**. Modify the action definition and click **OK**.
    - ❖ To remove an action, select the action and click **Delete**.

Figure 4-12 Defining an action



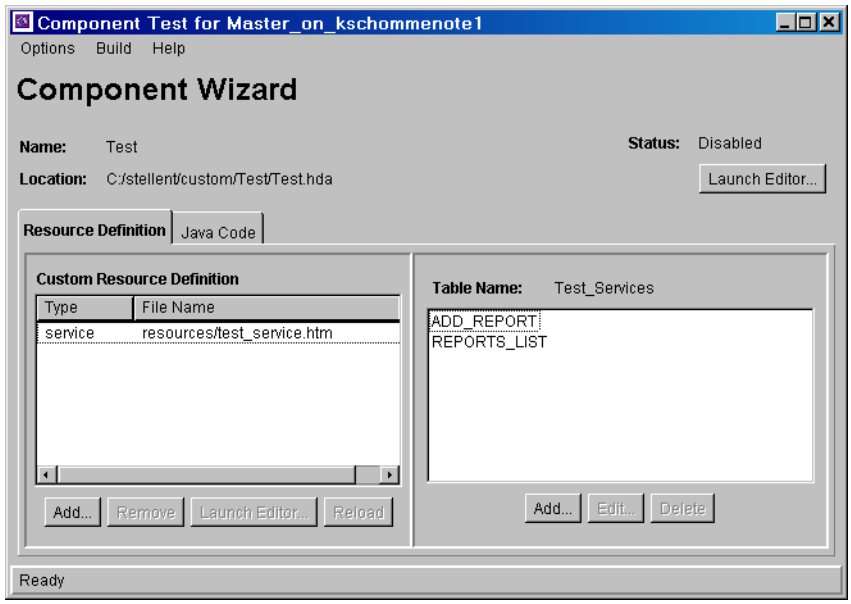
11. Click **Finish**.

    A dialog box asks if you want to launch the text editor to continue editing.

12. Click **Yes** to open the resource file in the text editor. Click **No** to return to the Component Wizard.

    The service resource file now appears in the Custom Resource Definition list, and the service table appears in the Table Name list in the right pane.

Figure 4-13 Custom service resource defined in the Component Wizard



Chapter

5

CORE CONTENT SERVER SERVICES

OVERVIEW

This chapter describes the services available for the Content Server. The services in this chapter are divided according to Service Class.

This chapter has the following sections:

- General Services (Core Content Server) (page 5-3)
- Doc Services (Core Content Server) (page 5-21)
- Doc Profile Services (Core Content Server) (page 5-113)
- File Services (Core Content Server) (page 5-117)
- Indexer Services (Core Content Server) (page 5-124)
- Internal Services (Core Content Server) (page 5-133)
- Meta Services (Core Content Server) (page 5-135)
- Miscellaneous Services (Core Content Server) (page 5-143)
- Page Handler/Page Request Services (Core Content Server) (page 5-144)
- Provider Manager Services (Core Content Server) (page 5-157)
- Schema Services (Core Content Server) (page 5-162)
- Search Services (Core Content Server) (page 5-169)
- User Services (Core Content Server) (page 5-178)

ABOUT CORE CONTENT SERVER SERVICES

Note: The most commonly used services have more extensive descriptions.

A typical service entry is similar to the following:

```
REMOVE_METAFILE_SUB
DocService 0 null SubService null !csDocumentNotDeleted(dDocName)
3:deleteRevFiles::0:!csUpdateFileDeleteFailed(dDocName)
2:Ddocument::0:null
```

In this example:

- The first line indicates the service name: `REMOVE_METAFILE_SUB`
- The second line lists the service's attributes:
  - Service class: `DocService`
  - Access level: `0`
  - Template: `null` (none used)
  - Service type: `SubService`
  - Subjects Notified: `null`
  - Error message: `Document Not Deleted`
- The remainder of the entry lists the service's actions, each line designating one action:
  - action type: for example, `3`, indicating a Java method.
  - action name: for example, `deleteRevFiles`
  - action parameters: if no parameters are specified, double-colons (`::`) are included.
  - action control mask: for example, `0`, indicating no control mask. The double-colon indicator (`::`) can also be included.
  - action error message, or null if no message is available.

Each element of the service's action is separated by a colon (`:`).

Important: All services have at least one required parameter. The IdcService parameter takes the name of the service as its argument. If other parameters are required, they are noted in the description of the service.

See Chapter 4 (*Customizing Services)* for details about a service's attributes and actions.

GENERAL SERVICES (CORE CONTENT SERVER)

General services are used throughout the Content Server and are often not tied to a specific functionality. Frequently used services are marked with an asterisk (*) in the following list.

This section describes these services:

- ADD_DOC_ACCOUNT (page 5-4)
- ADD_DOCEXTENSION (page 5-4)
- ADD_DOCFORMAT (page 5-5)
- ADD_DOCTYPE (page 5-6)
- APPLET_DOCINFO (page 5-6)
- CONFIG_INFO (page 5-7)
- DELETE_DOC_ACCOUNT (page 5-7)
- DELETE_DOCEXTENSION (page 5-7)
- DELETE_DOCFORMAT (page 5-8)
- DELETE_DOCTYPE (page 5-8)
- EDIT_DOCEXTENSION (page 5-8)
- EDIT_DOCFORMAT (page 5-9)
- EDIT_DOCTYPE (page 5-10)
- EDIT_TRACE_OPTIONS (page 5-10)
- GET_DATARESULTSET (page 5-10)
- GET_DOCEXTENSIONS (page 5-11)
- GET_DOCFORMATS (page 5-11)
- GET_DOCTYPES (page 5-11)
- GET_FILELIST (page 5-12)
- GET_METADEFS (page 5-12)
- GET_RESULT_OPTIONS (page 5-12)
- GET_SYSTEM_AUDIT_INFO (page 5-13)
- GET_TABLE (page 5-13)

- GET_USER_METADEFS (page 5-14)
- JAVA_PROPERTIES (page 5-14)
- LM_BUILD_WEB_STRING_FILES (page 5-14)
- LM_LOAD_LAYOUTS (page 5-14)
- LM_LOAD_LAYOUTS_SUB (page 5-14)
- LOAD_DOC_ENVIRONMENT (page 5-14)
- *LOGIN (page 5-15)
- MERGE_TABLE (page 5-16)
- *PING_SERVER (page 5-19)
- QUERY_DOC_ACCOUNTS (page 5-20)
- SOAP_FAULT (page 5-20)

ADD_DOC_ACCOUNT

Service used to create a new account. The most likely error is an account name that is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocAccount: The security account for the content item.

Example

```
IdcService=ADD_DOC_COLLECTION
dDocAccount=newaccount
```

ADD_DOCEXTENSION

Service that adds a file extension to an existing file. The most likely error is when a matching file with that extension already exists in the system.

Additional Required Service Parameters

- dExtension: The file extension such as *hcsf*, *doc*, *txt*.

- dFormat: The formatting process used to create the web-viewable version of the content. For example, *application/hcsf* or *application/doc*.

Example

```
IdcService=ADD_DOCEXTENSION
dExtension=doc
dFormat=application/doc
```

ADD_DOCFORMAT

Service that creates a new file format. The most likely error is when the file format already exists in the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dFormat: The formatting process used to create the web-viewable version of the content. For example, *application/hcsf* or *application/doc*.
- dConversion: The conversion algorithm is determined by the parameter *dConversion*.

  If the conversion type is not *Passthru* or *NoConversion* (default value) and the content is not web-viewable or has a publish state, the content gets added to the queue to get converted. Otherwise, the server looks for the web-viewable (if applicable) and the vault file and updates the processing state to say that the file has already been converted.

  If the file mime-type is *application/FDF*, the server sets the conversion to *exchange-fdf*.
- dDescription: The file format description.

Example

```
IdcService=ADD_DOCFORMAT
dFormat=application/doc
dDescription=msword
dConversion=PASSTHRU
```

ADD_DOCTYPE

Used to create a new content item type. The most likely error is when the content item type name is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocType: The new content item Type.
- dDescription: Description of the new type.
- dGif: The file name of the GIF image that is displayed as an icon to represent the new doc type. Include the *.gif* extension.

Example

```
IdcService=ADD_DOCTYPE
dDocType=MYTEST
dDescription=My Description.
dGif=adeng.gif
```

APPLET_DOCINFO

Service that retrieves content item information. The most likely error is when the content item no longer exists in the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

```
IdcService=APPLET_DOCINFO
dID=47
```

CONFIG_INFO

Service that retrieves configuration information from the Admin Server. The service retrieves configuration information for the currently active component.

Location: <*install_dir*>/shared/config/resources/std_services.htm

DELETE_DOC_ACCOUNT

Service that deletes an existing account.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocAccount: The account name.

Example

```
IdcService=DELETE_DOC
dDocAccount=mainaccount
```

DELETE_DOCEXTENSION

Service that deletes an existing file extension.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dExtension: The file extension, such as *hcsf*, *doc*, *txt*.

Example

```
IdcService=DELETE_DOCEXTENSION
dExtension=hcsf
```

DELETE_DOCFORMAT

Service that deletes an existing document format.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dFormat: The name of the MIME format. For example, *application/hcsf* or *application/doc*.

Example

```
IdcService=DELETE_DOCFORMAT
dFormat=application/hcsf
```

DELETE_DOCTYPE

Service that deletes an existing content item type. The most likely errors are when the specified file type does not exist or when a file of that type still exists in the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocType: The content item type.

Example

```
IdcService=DELETE_DOCTYPE
dDocType=TEST
```

EDIT_DOCEXTENSION

Service that modifies an existing file extension.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dExtension: The file extension such as *doc*, *txt*, or *pdf*.

- dFormat: The formatting process used to create the web-viewable version of the content. For example, *application/hcsf* or *application/doc*.

Example

```
IdcService=EDIT_DOCEXTENSION
dExtension=hcsf
dFormat=application/hcsf
```

EDIT_DOCFORMAT

Service that modifies an existing content item format. This service is called from the Configuration Manager applet.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dFormat: The formatting process used to create the web-viewable version of the content. For example, *application/hcsf* or *application/doc*.
- dDescription: The format description.
- dConversion: The conversion algorithm is determined by this parameter.

  If the conversion type is not *Passthru* or *NoConversion* (default value) and the content is not web-viewable or has a publish state, the content gets added to the queue to get converted. Otherwise, the server looks for the web-viewable (if applicable) and the vault file and updates the processing state to say that the file has already been converted.

  If the file mime-type is *application/FDF*, the server sets the conversion to *exchange-fdf*.

Example

```
IdcService=EDIT_DOCFORMAT
dFormat=application/hcsf
dDescription=hypercontent
dConversion=PASSTHRU
```

See Also

– *EDIT_GROUP (page 5-194)*

– *EDIT_DOCEXTENSION (page 5-8)*
– *EDIT_DOC_FORMATS (page 5-75)*

EDIT_DOCTYPE

Service that modifies an existing content item type.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocType: The content item type.
- dDescription: The type description.
- dGif: The file name of the GIF image that is displayed as an icon to represent the type. Include the *.gif* extension.

Example

```
IdcService=EDIT_DOCTYPE
dDocType=MY_TEST
dDescription=edit testing
dGif=admkt.gif
```

EDIT_TRACE_OPTIONS

Service that retrieves trace options on the System Audit Info page.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_DATARESULTSET

This service executes a Select query against the database. The query is built from the dataSource parameter. The service returns the following information:

- The resultset containing the results of the query.
- An indication if the results were truncated. If the dataSource was defined so it cannot exceed the maximum number of rows and if the query returns more than the maximum allowed, the copyAborted key is set to 1 (*true*). This indicates that the returned resultset only contains a subset of the query.

Any query that tries to select against certain core content server tables have a security clause applied. In particular, Documents, Revisions, and Users tables have extra security clauses applied.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dataSource: A Select query with a potential WHERE clause and ORDER BY clause that is provided by the caller. The dataSource is a Content Server resource, defined in the DataSources table (see the resource.htm file for the standard list of dataSources.)

Optional Service Parameters

- whereClause: The WHERE clause to the Select query.
- orderClause: If set to *true*, orders the query by clause.
- resultName: specifies the name to use for the resultset of the query.

GET_DOCEXTENSIONS

Service that returns a list of all content item extensions and the file format each extension is mapped to. Returns the properties information and ExtensionFormatMap ResultSet.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_DOCFORMATS

Service that returns a list of all content item formats and their associated conversion methods and descriptions. Returns the properties information and DocFormats ResultSet.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *GET_DOC_PAGE (page 5-146)*

GET_DOCTYPES

Service that returns a list of all content item types, their descriptions, and their associated GIF images. Returns the properties information and DocTypes ResultSet.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_FILELIST

An administrative service that retrieves a file listing in a directory matching a specified filter. The directory is specified by an ID, not by the user. The service then maps the ID to a directory.

In practice, this service is only used to retrieve the content type GIF list for the Configuration Manager applet. This list is located in the /weblayout/images/docgifs directory. The results are returned in the value specified in the fileListName parameter as an option list.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- directoryID: The directory identifier. Possible values include *docgifs, images, templates, resources.*
- fileFilter: Wild card filter to use to select the files.
- fileListName: Option list which holds the values returned.

GET_METADEFS

Service that returns a list of all custom metadata fields and their attributes. Returns the properties information and MetaFieldInfo ResultSet.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_RESULT_OPTIONS

Service that returns information from the search results templates.

Note: As of version 3.5.3, the content server no longer uses this service. It remains in the *std_services.htm* file as legacy code for reverse compatibility.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_SYSTEM_AUDIT_INFO

Service that retrieves system audit information for the content server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *APPEND_FILE_CACHING_INFO (page 5-118)*

– *APPEND_DATABASE_AUDIT_INFO (page 5-158)*

– *APPEND_SEARCH_AUDIT_INFO (page 5-170)*

GET_TABLE

Service that exports a database table to a ResultSet in an HDA file.

If the specified table is not found, the service fails. It is up to the calling program receiving the HDA data to store this ResultSet for later usage.

The most likely error is a table name that does not exist.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ tableName: The name of table to export.

Example

❖ IdcCommand command file format:

```
IdcService=GET_TABLE
tableName=Users
```

❖ HDA format:

```
@Properties LocalData
IdcService=GET_TABLE
tableName=Users
@end
```

GET_USER_METADEFS

Service that returns a list of all user information fields and their attributes. Returns the properties information and ResultSet.

Location: <*install_dir*>/shared/config/resources/std_services.htm

JAVA_PROPERTIES

Service that returns information about the Java Resource Environment in use.

Location: <*install_dir*>/shared/config/resources/std_services.htm

LM_BUILD_WEB_STRING_FILES

Service that publishes strings to a static `.js` file.

Location: <*install_dir*>/shared/config/resources/std_services.htm

LM_LOAD_LAYOUTS

Service that loads user interface layouts from the <*install_dir*>/weblayout/common/layouts directory. Calls LM_LOAD_LAYOUTS_SUB.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *LM_LOAD_LAYOUTS_SUB (page 5-14)*

LM_LOAD_LAYOUTS_SUB

Subservice used to call layouts from the <*install_dir*>/weblayout/common/layouts directory.

Location: <*install_dir*>/shared/config/resources/std_services.htm

LOAD_DOC_ENVIRONMENT

This subservice loads the template page configuration information for the content item environment.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *GET_DOC_PAGE (page 5-146)*

– *GET_SECURE_PAGE (page 5-151)*

LOGIN

Service that forces a login and executes an HTML page request. This is one of many ways to authenticate the user with the web server.

Access Level: N/A (0)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- Action: The action to execute. Usually GetTemplatePage.
- Page: The name of the page template.
- Auth: Required only when logging in from a browser interface. For example:

  `Auth=Internet`

  The other possible value is `Intranet`, which is used for NTLM logins.

Results

- Local Data:
  - dUser
  - RedirectParams
  - StatusMessage
- Response Template: null (default redirect to HOME_PAGE)

Used By

- Resource Includes:
  - pne_nav_userprofile_links
  - subscription_action_script
  - home_page_static_content

- Templates:
  - QUERY_NOTIFICATION (query_notification_mail.htm)
  - SELF_REGISTER_PROMPT_LOGIN (self_register_prompt_login.htm)
  - (std_home_page.htm)
- Standard Navigation: commonNav.js

Example

Displays the home page when you login to the content server:

```
IdcService=LOGIN
Action=prepareRedirect
Page=HOME_PAGE
```

MERGE_TABLE

Service that merges a ResultSet with an existing database table. All errors are logged to the content server log.

Important: By default, this service is not error tolerant and does not execute the merge in a transaction. Consequently, if the *isErrorTolerant* or *isTransactional* optional parameters are not set to *true* and if the service encounters an error, it aborts where the error occurred and does not roll back the already updated and inserted rows. In this situation, if you do not specify constraints and rerun the merge table, the merge fails in Microsoft SQL Server and Oracle due to constraint violations.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- tableName: The name of the database table to merge into. The ResultSet must have the same name as this table.

Important: The ResultSet definition for the table containing the rows to insert or update must also be specified.

Optional Service Parameters

- constraintKeys: Specifies a comma-separated list of column names used as unique identifiers. The service initially does a look up with these constraints. If the row is found, the service does an update. Otherwise, a new row is inserted.

  If this parameter is not defined, the service attempts to insert the rows.

- isDeleteTable: If *true*, the service deletes the table before attempting the merge. The delete is performed in a transaction with the merge, if *isTransactional* is *true*. By default, this value is *false*.

- isErrorTolerant: If *true*, does not abort when errors are encountered during the merge. By default, this value is *false*.

- isTransactional: If *true*, the merge is performed in a transaction. If an error occurs during the merge and *isErrorTolerant* is *false*, the merge aborts and rolls back all the changes. By default, this value is *false*.

Example

- IdcCommand command file format:

```
# Merge new entries into the Alias table
IdcService=MERGE_TABLE
tableName=Alias
constraintKeys=dAlias

# Rows to be updated or inserted
@ResultSet Alias
2
dAlias
dAliasDescription
MyAlias
Just a test alias
@end
<<EOD>>

# Delete the alias table and insert new rows, do a rollback if something
fails.
IdcService=MERGE_TABLE
tableName=Alias
```

```
isTransactional=true
isDeleteTable=true

# Rows to be updated or inserted
@ResultSet Alias
2
dAlias
dAliasDescription
MyAlias
Just a test alias
@end
<<EOD>>
```

- HDA format (Example 1):

```
@Properties LocalData
IdcService=MERGE_TABLE
tableName=Alias
constraintKeys=dAlias
@end
@ResultSet Alias
2
dAlias
dAliasDescription
MyAlias
Just a test alias
@end
```

- HDA format defining a table name and ResultSet and inserting a new row into the database (inserts new row into Users table):

```
@Properties LocalData
IdcService=MERGE_TABLE
tableName=Users
@end
@ResultSet Users
2
dName
dUserAuthType
```

```
user15
LOCAL
@end
```

PING_SERVER

Service that evaluates if a connection to the content server exists and returns status information. It also forces users to log in if they are not already logged in.

Tech Tip: Execute a PING_SERVER request before calling other services to ensure that there is a connection to the content server and that you are logged in as a user authorized to execute commands.

Access Level: N/A (0)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results

- Local Data:
  - dUser
  - StatusMesage
- Response Template: null

Used By

- Applets:
  - Batch Loader
  - Configuration Manager
  - Page Builder
- Templates:
  - SUBSCRIBE_FORM (subscribe_form.htm)
  - SUBSCRIPTION_LIST (subscription_list.htm)
  - UNSUBSCRIBE_FORM (unsubscribe_form.htm)

Example

HDA format:

```
@Properties LocalData
IdcService=PING_SERVER
IsJava=1
Auth=Internet
@end
```

Sample return information:

```
Content-type: text/html
Content-Length: 421

<?hda version="6.0 SNAP-020207" jcharset=Cp1252 encoding=iso-8859-1?>

@Properties LocalData
dUser=sysadmin
blFieldTypes=StatusMessage message
refreshSubMonikers=
StatusMessage=You are logged in as 'sysadmin'.
loadedUserAttributes=1
blDateFormat=M/d{/yy} {h:mm[:ss] {aa}[zzz]}!tAmerica/Chicago!mAM,PM
changedSubjects=
refreshSubjects=
Auth=Internet
refreshMonikers=
changedMonikers=
IdcService=PING_SERVER
IsJava=1
@end
```

QUERY_DOC_ACCOUNTS

Service that returns a list of all accounts in the content server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

SOAP_FAULT

Service used to process failed SOAP requests.

Location: <*install_dir*>/shared/config/resources/std_services.htm

DOC SERVICES (CORE CONTENT SERVER)

Doc services perform actions on documents, such as checkin, checkout, subscription actions, and accessing document information. Frequently used services are marked with an asterick (*) in the following list.

This section describes these services:

- ASSIGN_DOCINFO_FORM (page 5-24)
- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_ARCHIVE_NO_NOTIFY (page 5-25)
- CHECKIN_BYNAME (page 5-26)
- *CHECKIN_CONFIRM_FORM (page 5-30)
- *CHECKIN_LIST (page 5-30)
- *CHECKIN_NEW (page 5-31)
- *CHECKIN_NEW_FORM (page 5-37)
- *CHECKIN_NEW_SUB (page 5-38)
- *CHECKIN_SEL (page 5-40)
- *CHECKIN_SEL_FORM (page 5-45)
- CHECKIN_SEL_SUB (page 5-47)
- *CHECKIN_SIMILAR_FORM (page 5-47)
- *CHECKIN_UNIVERSAL (page 5-48)
- *CHECKOUT (page 5-52)
- CHECKOUT_BY_NAME (page 5-55)
- *CHECKOUT_OK (page 5-56)
- *CHECKOUT_SUB (page 5-57)
- CONTINUE_CHECKIN (page 5-59)
- CONTINUE_SUBMIT_HTML_FORM (page 5-59)

- CREATE_SUBSCRIPTION_TYPE (page 5-60)
- DELETE_BYCLASS (page 5-60)
- DELETE_BYNAME (page 5-60)
- DELETE_BYREV (page 5-61)
- DELETE_BYREV_REVISION (page 5-61)
- DELETE_CHECKIN_CACHE (page 5-61)
- DELETE_DOC (page 5-61)
- *DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)
- DELETE_SUBSCRIPTION_TYPE (page 5-64)
- DOC_FORMATS_WIZARD (page 5-64)
- *DOC_INFO (page 5-64)
- *DOC_INFO_BY_NAME (page 5-68)
- *DOC_INFO_LATESTRELEASE (page 5-70)
- *DOC_INFO_SIMPLE (page 5-71)
- *DOC_INFO_SIMPLE_BYREV (page 5-72)
- *DOC_SUBS_LIST (page 5-73)
- EDIT_DOC_FORMATS (page 5-75)
- FORM_PROCESS (page 5-76)
- FORM_SUBMIT (page 5-76)
- GET_CACHED_CHECKIN_INFO (page 5-77)
- GET_DOC_CONFIG_INFO (page 5-77)
- GET_DOC_SUBSCRIBERS (page 5-78)
- *GET_EXPIRED (page 5-78)
- GET_PACKAGE_ENVIRONMENT_PAGE (page 5-79)
- *GET_UPDATE_FORM (page 5-80)
- NOTIFY_INDEXER (page 5-81)
- ODMA_DOC_INFO_SIMPLE (page 5-81)

- PACKAGE_ENVIRONMENT (page 5-81)
- REMOVE_METAFILE_SUB (page 5-81)
- REPLACE_METAFILE_SUB (page 5-82)
- *RESUBMIT_FOR_CONVERSION (page 5-82)
- *REV_HISTORY (page 5-84)
- SELECTDOC (page 5-84)
- *SUBMIT_HTML_FORM (page 5-85)
- *SUBSCRIBE (page 5-86)
- SUBSCRIBE_DOC_USER (page 5-88)
- SUBSCRIBE_EX (page 5-89)
- *SUBSCRIBE_FORM (page 5-90)
- *SUBSCRIPTION_LIST (page 5-91)
- *UNDO_CHECKOUT (page 5-92)
- UNDO_CHECKOUT_BY_NAME (page 5-95)
- *UNSUBSCRIBE (page 5-95)
- *UNSUBSCRIBE_FORM (page 5-97)
- *UNSUBSCRIBE_FROM_LIST (page 5-98)
- UNSUBSCRIBE_FROM_LIST_EX (page 5-100)
- UPDATE_BYREV (page 5-101)
- *UPDATE_DOCINFO (page 5-102)
- *UPDATE_DOCINFO_BYFORM (page 5-105)
- UPDATE_DOCINFO_BYREV (page 5-109)
- UPDATE_DOCINFO_METAFILE_BYREV (page 5-109)
- UPDATE_DOCINFO_STATUS (page 5-109)
- UPDATE_DOCINFO_SUB (page 5-110)
- UPDATE_SUBSCRIPTION_NOTIFY (page 5-110)
- UPDATE_SUBSCRIPTION_TYPE (page 5-110)
- UPDATE_SUBSCRIPTION_USED (page 5-111)

- UPDATE_METADATA (page 5-112)
- VALIDATE_DOCINFO (page 5-112)
- *WORK_IN_PROGRESS (page 5-112)

ASSIGN_DOCINFO_FORM

Service that retrieves the DOCINFO_FORM and assigns it to content.

Location: <*install_dir*>/shared/config/resources/std_services.htm

CACHE_CHECKIN_NEW

Service used with preview. When a user performs a preview during a checkin the information is cached on the server side. This service is modeled after the non-cache version and uses the same parameters as CHECKIN_NEW.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_NEW (page 5-31)*

CACHE_CHECKIN_SEL

Service used with preview. When a user performs a preview during a checkin the information is cached on the server side. This service is modeled after the non-cache version and uses the same parameters as CHECKIN_SEL.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_SEL (page 5-40)*

CACHE_SUBMIT_HTML_FORM

Service used with preview. When a user performs a preview during a checkin the information is cached on the server side. This service is modeled after the non-cache version and uses the same parameters as SUBMIT_HTML_FORM.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *SUBMIT_HTML_FORM (page 5-85)*

CACHE_WORKFLOW_CHECKIN

Service used with preview. When a user performs a preview during a checkin the information is cached on the server side. This service is modeled after the non-cache version and uses the same parameters as WORKFLOW_CHECKIN.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *WORKFLOW_CHECKIN (page 6-45)*

CHECKIN_ARCHIVE_NO_NOTIFY

A general checkin service used by client applications. It is generally used when the client application knows it will be checking in a large number of files and it does not want the indexer to be initiated by the check-in activity.

NO_NOTIFY indicates that this service does not notify the Content Server subjects that are normally notified during a checkin. This service suppresses the Released Documents subject, which starts the indexer.

The user of this service should notify the indexer subject to proceed to do work by calling the NOTIFY_INDEXER service.

This service is not used by the archiver.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

See CHECKIN_ARCHIVE (page 7-6) for additional parameters.

Optional Service Parameters

See CHECKIN_ARCHIVE (page 7-6) for additional parameters.

Example

```
IdcService=CHECKIN_ARCHIVE_NO_NOTIFY
```

```
Action=insert
dDocAuthor=user1
dDocName=test
dDocTitle=new content
dSecurityGroup=Public
primaryFile=c:/test.txt
doFileCopy=true
```

See Also

– *NOTIFY_INDEXER (page 5-81)*

CHECKIN_BYNAME

Service that checks in a content item revision based on the content item name or Content ID.

Location: <*install_dir*>/shared/config/resources/std_services.htm

A user must have Admin permission to the content item's security settings to change the security group (*dSecurityGroup*), account (*dDocAccount*), or author (*dDocAuthor*).

- If the content item uses entity security, the user must have at least Write permission granted by the entities, unless the user has Admin rights to the security group being assigned to the document.
- If the content item is in a workflow, the user must be a reviewer/contributor for the current step.
- If metafile generation is enabled (by setting the *AllowPrimaryMetaFile* and/or *AllowAlternateMetaFile* environment variables) and the metafile generation variables (*createPrimaryMetaFile* and *createAlternateMetaFile*) are *true*, a real file cannot be associated with the current content item. Only one metafile can be associated with a content item. If *createPrimaryMetaFile* is *true*, *createAlternateMetaFile* must be FALSE and vice-versa.
- The most likely error is a revision failing to insert or when the refinery was not successfully initiated.

Important: Either the content item name or the content item revision ID must be specified.

Additional Required Service Parameters

- dDocName: The Content ID for the content item
  - If Content ID auto generation is enabled, this parameter is not required. If *dDocName* is defined, it overrides the auto generated Content ID.
  - The Content ID cannot contain spaces or invalid characters ;/ \?:@&=+"#%<>*~| [ ].
- dDocAuthor: The content item author (contributor).
- dDocTitle: The content item title.
- dDocType: The content item type.
- dSecurityGroup: The security group such as *Public* or *Secure*.
- dDocAccount: The account for the content item. Required only if accounts are enabled.
- primaryFile: The absolute path to the location of the file as seen from the server. Use the forward slash as the file separator.

  A primary file must be specified unless checking in metadata only. If an alternate file is specified with the primary file, the content refinery converts the alternate file. Otherwise, the primary file is converted.
  - If a primary file is not specified, a metafile can be used in its place. Only one metafile can exist for each content item (a primary AND alternate meta file cannot co-exist).
  - If both a primary and alternate file is specified, their extensions must be different.
- doFileCopy: 1 (*true*)—The file is not deleted from the hard drive after checkin.

  0 (*false*)—The file is removed from the hard drive after checkin.
- Required custom fields: Custom metadata fields that are required must also be specified.

Optional Service Parameters

- dID: The generated content item revision ID.
- alternateFile: The alternate file for conversion.
  - Only one metafile can exist for each content item (a primary AND alternate meta file cannot co-exist.)

- If an alternate file is specified with the primary file, the content refinery converts the alternate file. Otherwise, the primary file is converted.

❖ AutoNumberPrefix: This is a configuration entry but can be specified as a parameter. The auto prefix script is evaluated after the revision class ID is generated and before metadata validation.

❖ createAlternateMetaFile: To enable alternate metafile generation, the *AllowAlternateMetaFile* environment value must be *true* and the parameter *createAlternateMetaFile* must be *true*.

❖ createPrimaryMetaFile: To enable primary metafile generation, the *AllowPrimaryMetaFile* environment value must be *true* and the parameter *createPrimaryMetaFile* must be *true*.

❖ dConversion: The conversion algorithm is determined by this parameter.

If the conversion type is not *Passthru* or *NoConversion* (default value) and the content is not web-viewable or has a publish state, the content gets added to the queue to get converted. Otherwise, the server looks for the web-viewable (if applicable) and the vault file and updates the processing state to say that the file has already been converted.

If the file mime-type is *application/FDF*, the server sets the conversion to *exchange-fdf*.

❖ dCreateDate: The date the content item was created. By default, this is the current date.

❖ dExtension: The file extension such as *hcsf*, *doc*, *txt*.

❖ dFormat: The formatting process used to create the web-viewable version of the content. For example, *application/hcsf* or *application/doc*.

❖ dInDate: The content release date. The date the content item is to be released to the web. By default, this is the current date.

If the content release date (*dInDate*) is not specified, the creation date (*dCreateDate*) is used. This value is auto generated if it is not supplied.

❖ dOutDate: The content expiration date. By default, this is blank and does not specify an expiration date.

If the content expiration date (*dOutDate*) is not entered, the value remains empty. This is a valid state.

❖ dPublishState: The publish state. If the content item exists in the system *dPublishState* must be empty.

- dReleaseState: The release state (used to indicate the web state of the revision).
- dRevClassID: The revision class ID.
- dRevLabel: The revision label for the content item. If set, the label is used to locate the specified revision.
- dWfType: The workflow type such as *Basic* or *Criteria*.
- IsAutoNumber: This is a configuration entry but can be specified as a parameter. If the configuration variable *IsAutoNumber* is set to *true*, the Content ID (*dDocName*) is generated by concatenating the auto prefix with the revision class ID (*dRevClassID*, the counter of uniquely differentiated content items), pre-filled with leading zeroes to be six digits long. Auto numbering is not performed if the Content ID (*dDocName*) is already specified.
- IsEditRev: If set to *true*, this entry checks if the content revision is in a workflow and enables editing.
- IsWorkflowInfo: This is a configuration entry but can be specified as a parameter. If set to *true*, this entry checks for workflow information and checks allowable actions for the workflow steps.
- webViewableFile: If a content item has a web-viewable file associated with it, the conversion format is the format of the web-viewable file (the parameter *webViewableFile:format*) and the extension is the web-viewable file's extension (the parameter *dWebExtension*). Otherwise, the extension and file format are determined by the parameters *dExtension* and *dFormat*, respectively. The user can override the file format and extension by setting these additional parameters.
- Optional custom fields: Custom metadata fields that are not required can also be specified.

Example

```
IdcService=CHECKIN_BYNAME
dDocName=test1000
dSecurityGroup=public
dDocAuthor=sysadmin
dDocType=ADENG
dDocTitle=another test
doFileCopy=true
primaryFile=c:/test.txt
```

CHECKIN_CONFIRM_FORM

Service that returns confirmation upon successful checkin through a browser.

Access Level: Write, Global, Scriptable (50)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The content ID.
- dDocAuthor: The document author.
- dDocName: The document name.
- dDocTitle: The document title.

Results

- Local Data:
  - dDocAuthor
  - dDocName
  - dDocTitle
  - dID
- Response Template: CHECKIN_CONFIRM (checkin_confirm.htm)

Used By

- Redirect service for: CHECKIN_NEW, CHECKIN_SEL, CONTINUE_CHECKIN

CHECKIN_LIST

Service that returns a list of checked-out items (those that are not checked in or deleted). The most likely error is when the checkout list cannot be retrieved from the database.

Access Level: Write, Global, Scriptable (50)

Queries Executed: QcheckinCachesForUser

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Optional Service Parameters

- userOnly: When 1 (*true*), returns only items checked out to the current user.
- MaxQueryRows: Controls the number of items returned in the list.

Results

- Result Sets:
  - CHECKIN_LIST (All Fields from Revisions and DocMeta for checked-out revisions.)
  - CHECKIN_CACHES (All Fields from DatedCaches.)
- Local Data: copyAborted, DataSource
- Response Template: CHECKIN_LIST (checkin_list.htm)

Used By

- Resource Includes:
  - pne_nav_management_links
  - std_doc_man_pages
  - checkin_multiuploadapplet_processing_functions
- Standard Navigation:
  - commonNav.js
  - commonBundle.js
- Other:
  - SoapCustom:Wsdl:CheckIn:Services
  - Redirect service for: DELETE_CHECKIN_CACHE

CHECKIN_NEW

Service that checks in a new content item. This service the CHECKIN_NEW_SUB subservice.

Access Level: Write (2)

Location: <*install_dir*>/shared/config/resources/std_services.htm

- If you attempt to set the author (*dDocAuthor*) to someone other than the currently logged in user, you must have Admin permission to the security attributes being assigned to the document.
- If the content uses entity security, the user must have at least Write permission granted by the entities, unless the user has Admin rights to the security group being assigned to the document.
- If metafile generation is enabled (by setting the *AllowPrimaryMetaFile* and/or *AllowAlternateMetaFile* environment variables) and the metafile generation variables (*createPrimaryMetaFile* and *createAlternateMetaFile*) are true, a real file cannot be associated with the current content item. Only one metafile can be associated with a content item—if *createPrimaryMetaFile* is true, *createAlternateMetaFile* must be false and vice-versa.
- The most likely error is when the content item was not properly defined.
- This service executes the subservice CHECKIN_NEW_SUB. This subservice checks in a new content item revision.
- A primary file is required. If you do not want to check in a primary file and want to check in only metadata, an additional parameter must be included and a configuration entry added in the content server.

  Required additional parameter (metadata checkin):

  ```
  createPrimaryMetaFile=true
  ```

  Required content server configuration entry (metadata checkin):

  ```
  AllowPrimaryMetaFile=true
  ```

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
  - If Content ID auto generation is enabled, this parameter is not required. If *dDocName* is defined, it overrides the auto generated Content ID.
  - The Content ID cannot contain spaces or invalid characters ;/ \?:@&=+"#%<>*~| [ ].
- dDocAuthor: The content item author (contributor).
- dDocTitle: The content item title.
- dDocType: The content item Type.
- dSecurityGroup: The security group such as *Public* or *Secure*.

- dDocAccount: The account for the content item. Required only if accounts are enabled.
- primaryFile: The absolute path to the location of the file as seen from the server. Use the forward slash as the file separator.

  A primary file must be specified unless checking in metadata only. If an alternate file is specified with the primary file, the content refinery converts the alternate file. Otherwise, the primary file is converted.

  - If a primary file is not specified, a metafile can be used in its place. Only one metafile can exist for each content item (a primary AND alternate meta file cannot co-exist).
  - If both a primary and alternate file is specified, their extensions must be different.
- Required custom fields: Custom metadata fields that are required must also be specified.

Optional Service Parameters

- alternateFile: The alternate file for conversion.
  - Only one metafile can exist for each content item (a primary AND alternate meta file cannot co-exist.)
  - If an alternate file is specified with the primary file, the content refinery converts the alternate file. Otherwise, the primary file is converted.
- createAlternateMetaFile: To enable alternate metafile generation, the *AllowAlternateMetaFile* environment value must be set to *true* and the parameter *createAlternateMetaFile* must be set to *true*.
- createPrimaryMetaFile: To enable primary metafile generation, the *AllowPrimaryMetaFile* environment value must be *true* and the parameter *createPrimaryMetaFile* must be set to *true*.
- dConversion: The conversion algorithm is determined by this parameter.

  If the conversion type is not *Passthru* or *NoConversion* (default value) and the content is not web-viewable or has a publish state, the content gets added to the queue to get converted. Otherwise, the server looks for the web-viewable (if applicable) and the vault file and updates the processing state to say that the file has already been converted.

  If the file mime-type is *application/FDF*, the server sets the conversion to *exchange-fdf*.

- dCreateDate: The date the content item was created. By default, this is the current date.
- dFormat: The formatting process used to create the web-viewable version of the content item. For example, *application/hcsf* or *application/doc*.
- dInDate: The release date. By default, this is the current date. If the content release date (*dInDate*) is not specified, the creation date (*dCreateDate*) is used. This value is auto generated if it is not supplied.
- dOutDate: The expiration date. By default, this is blank and does not specify an expiration date. If the content expiration date (*dOutDate*) is not entered, the value remains empty. This is a valid state.
- dProjectID: The Content Publisher project ID generated by the content server.
- dPublishType: The publish type (used with content created by Content Publisher). These values can be assigned:

  H - Home

  N - Navigation

  S - Query results pages

  P - Pages

  G - Gallery Graphics

  C - Contributor embedded graphics

  O - All others
- dRendition: Used to specify an additional rendition of the content item.
- dRevClassID: The revision class ID.
- dRevisionLabel: The revision label for the content item. If set, the label is used to locate the specified revision.
- AutoNumberPrefix: This is a configuration entry but can be specified as a parameter. The auto prefix script is evaluated after the revision class ID is generated and before metadata validation.
- IsAutoNumber: This is a configuration entry but can be specified as a parameter. If the configuration variable *IsAutoNumber* is set to *true*, the Content ID (*dDocName*) is generated by concatenating the auto prefix with the revision class ID (*dRevClassID*, the counter of uniquely differentiated content items), pre-filled with leading zeroes to be six digits long. Auto numbering is not performed if the Content ID (*dDocName*) is already specified.

- webViewableFile: If a content item has a web-viewable file associated with it, the conversion format is the format of the web-viewable file (the parameter *webViewableFile:format*) and the extension is the web-viewable file's extension (the parameter *dWebExtension*). Otherwise, the extension and file format are determined by the parameters *dExtension* and *dFormat*, respectively. The user can override the file format and extension by setting these additional parameters.
- Optional custom fields: Custom metadata fields that are not required can also be specified.

Results

- Local Data:
  - dAction
  - dActionDate
  - dClbraName
  - dConversion: If refinery-processes, contains information about what conversion occurred.
  - dCreateDate
  - dDocAccount
  - dDocAuthor
  - dDocID
  - dDocName
  - dDocTitle
  - dDocType
  - dExtension
  - dFileSize
  - dFormat
  - dID: Internal reference to ID for the new content item.
  - dInDate
  - dIsWebFormat
  - dIsPrimary
  - dLocation
  - dOriginalName
  - dOutDate

- dpAction
- dpEvent
- dProcessingState
- dPublishState
- dPublishType
- dRawDocID
- dReleaseState
- dRevClassID
- dRevisionID
- dRevLabel
- dRevRank
- dSecurityGroup
- dStatus
- dUser
- dWorkflowState
- isDocProfileUsed
- isEditMode
- isNew
- isStatusChanged
- prevReleaseState
- primaryFile
- StatusCode
- StatusMessage
- VaultfilePath
- WebfilePath
- xClbraAliasList
- xClbraUserList
- Plus any custom metadata related to the item checked in

❖ Response Templates:

- REDIRECT_TEMPLATE (redirect_template.htm)
- Default redirect service: CHECKIN_CONFIRM_FORM

Used By

❖ Resource Includes: std_doc_page_definitions

Example

```
IdcService=CHECKIN_NEW
dDocName=test1111
dDocTitle=test information
dDocAuthor=john
dSecurityGroup=public
primaryFile=c:/test.txt
doFileCopy=1
```

See Also

– *CACHE_CHECKIN_NEW (page 5-24)*

– *CHECKIN_NEW_SUB (page 5-38)*

CHECKIN_NEW_FORM

Service that returns the check-in form for a new content item in a browser. The most likely error is when the content item was not properly defined.

Access Level: Write, Global, Scriptable (50)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results

❖ Result Sets:

- DocFormats (dFormat, dConversion, dDescription)
- DocTypes (dDocType, dDescription, dGif)

❖ Response Template: CHECKIN_NEW_FORM (checkin_new.htm)

Used By

❖ Applets:

- Repository Manager
- Workflow Admin

- User Admin
- Configuration Manager

❖ Resource Includes:

- pne_nav_management_links
- pne_nav_bookmark_links
- calculate_doc_profile_urls
- std_doc_man_pages

❖ Standard Navigation:

- commonNav.js
- commonBundle.js

CHECKIN_NEW_SUB

Subservice that checks in a new content item revision. Depending on which service calls this subservice, other validation is done, such as determining if this content item belongs in a criteria workflow. Also includes computing of derived fields, adding of renditions, and initiation of refinery processing. A security check against group and account may need to be performed before executing this subservice.

Access Level: Subservice (N/A)

Inputs: All standard required fields for content checkin must be provided, plus any optional content metadata.

Queries Executed:

- QnextRevID
- UnextRevID
- Irevision
- Imeta
- Idocument
- IdocHistory

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results

❖ Local Data:

- dAction
- dActionDate
- dClbraName
- dConversion
- dCreateDate
- dDocAccount
- dDocAuthor
- dDocID
- dDocName
- dDocTitle
- dDocType
- dExtension
- dFileSize
- dFormat
- dID
- dInDate
- dIsPrimary
- dIsWebFormat
- dLocation
- dOriginalName
- dOutDate
- dpAction
- dpEvent
- dProcessingState
- dPublishState
- dPublishType
- dRawDocID
- dReleaseState
- dRevClassID
- dRevisionID
- dRevLabel

- dRevRank
- dSecurityGroup
- dStatus
- dUser
- dWorkflowState
- isDocProfileUsed
- isEditMode
- isNew
- isStatusChanged
- prevReleaseState
- primaryFile
- StatusCode
- StatusMessage
- VaultfilePath
- xClbraAliasList
- xClbraUserList
- WebfilePath
- Plus any custom metadata related to the item checked in

Used By

❖ Services: CHECKIN_NEW

See Also

– *CHECKIN_NEW (page 5-31)*

– *CHECKIN_UNIVERSAL (page 5-48)*

– *CHECKIN_ARCHIVE (page 7-6)*

CHECKIN_SEL

Service that checks in a revision to an existing content item. This calls the CHECKIN_SEL_SUB subservice.

Access Level: Write (2)

Location: <*install_dir*>/shared/config/resources/std_services.htm

- The content item must be checked out for this service to execute.
- If the content item is in a workflow, the user must be a reviewer/contributor for the current step.
- A user must have Admin permission to the content item's security settings to change the security group (*dSecurityGroup*), account (*dDocAccount*), or author (*dDocAuthor*).
- Any metadata values not specified in the service call are inherited from the previous revision.
- If metafile generation is enabled (by setting the *AllowPrimaryMetaFile* and/or *AllowAlternateMetaFile* environment variables) and the metafile generation variables (*createPrimaryMetaFile* and *createAlternateMetaFile*) are *true*, a real file cannot be associated with the current content item. Only one metafile can be associated with a content item—if *createPrimaryMetaFile* is *true*, *createAlternateMetaFile* must be false and vice-versa.
- The content item cannot be in the published state.
- The current user must be the Author of the content item or have Admin permission to check in a revision.
- The most likely error is when the content item is no longer in the system.
- This service executes the subservice CHECKIN_SEL_SUB. This subservice checks in a revision to a content item.

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
  - If Content ID auto generation is enabled, this parameter is not required. If *dDocName* is defined, it overrides the auto generated Content ID.
  - The Content ID cannot contain spaces or invalid characters ;/ \?:@&=+"#%<>*~| [ ].
- dDocAuthor: The content item author (contributor).
- dDocTitle: The content item title.
- dDocType: The content item Type.
- dID: The generated content item revision ID.
- dRevLabel: The content item revision label.

- dSecurityGroup: The security group such as *Public* or *Secure*.
- dDocAccount: The account for the content item. Required only if accounts are enabled.
- primaryFile: The absolute path to the location of the file as seen from the server. Use the forward slash as the file separator. A primary file must be specified unless checking in metadata only. If an alternate file is specified with the primary file, the content refinery converts the alternate file. Otherwise, the primary file is converted.
  - If a primary file is not specified, a metafile can be used in its place. Only one metafile can exist for each content item (a primary AND alternate meta file cannot co-exist).
  - If both a primary and alternate file is specified, their extensions must be different.
- doFileCopy: 1 (*true*)—The file is not deleted from the hard drive after checkin. 0 (*false*)—The file is removed from the hard drive after checkin.
- Required custom fields: Custom metadata fields that are required must also be specified.

Optional Service Parameters

- alternateFile: The alternate file for conversion.
  - Only one metafile can exist for each content item (a primary AND alternate meta file cannot co-exist.)
  - If an alternate file is specified with the primary file, the content refinery converts the alternate file. Otherwise, the primary file is converted.
- createAlternateMetaFile: To enable alternate metafile generation, the *AllowAlternateMetaFile* environment value must be set to *true* and the parameter *createAlternateMetaFile* must be set to *true*.
- createPrimaryMetaFile: To enable primary metafile generation, the *AllowPrimaryMetaFile* environment value must be set to *true* and the parameter *createPrimaryMetaFile* must be set to *true*.
- dConversion: The conversion algorithm is determined by the parameter dConversion.

  If the conversion type is not *Passthru* or *NoConversion* (default value) and the content is not web-viewable or has a publish state, the content gets added to the queue to get converted. Otherwise, the server looks for the web-viewable (if applicable) and the vault file and updates the processing state to say that the file has already been converted.

- If the file mime-type is *application/FDF*, the server sets the conversion to *exchange-fdf*.
- dCreateDate: The date the content item was created. By default, this is the current date.
- dFormat: The formatting process used to create the web-viewable version of the content. For example, *application/hcsf* or *application/doc*.
- dInDate: The content release date. The date the content item is to be released to the web. By default, this is the current date. If the content release date (*dInDate*) is not specified, the creation date (*dCreateDate*) is used. This value is auto generated if it is not supplied.
- dOutDate: The content expiration date. By default, this is blank and does not specify an expiration date. If the content expiration date (*dOutDate*) is not entered, the value remains empty. This is a valid state.
- dProjectID: The Content Publisher project ID generated by the content server.
- dPublishState: The publish state. If the content item exists in the system *dPublishState* must be empty.
- dPublishType: The publish type (used with content created by Content Publisher). These values can be assigned:

  H - Home

  N - Navigation

  S - Query results pages

  P - Pages

  G - Gallery Graphics

  C - Contributor embedded graphics

  O - All others
- dRevClassID: The revision class ID.
- AutoNumberPrefix: This is a configuration entry but can be specified as a parameter. The auto prefix script is evaluated after the revision class ID is generated and before metadata validation.
- IsAutoNumber: This is a configuration entry but can be specified as a parameter. If the configuration variable IsAutoNumber is set to *true*, the Content ID (*dDocName*) is generated by concatenating the auto prefix with the revision class ID (*dRevClassID*, the counter of uniquely differentiated content items), pre-filled with leading zeroes to

be six digits long. Auto numbering is not performed if the Content ID (*dDocName*) is already specified.

- IsWorkflowInfo: This is a configuration entry but can be specified as a parameter. If set to *true*, this entry checks for workflow information and checks allowable actions for the workflow steps.
- webViewableFile: If a content item has a web-viewable file associated with it, the conversion format is the format of the web-viewable file (the parameter *webViewableFile:format*) and the extension is the web-viewable file's extension (the parameter *dWebExtension*). Otherwise, the extension and file format are determined by the parameters *dExtension* and *dFormat*, respectively. The user can override the file format and extension by setting these additional parameters.
- RedirectUrl: Used to display another page after a topic file has been changed. If omitted, the user is redirected to the Content Server home page.
- Optional custom fields: Custom metadata fields that are not required can also be specified.

Results

- Local Data:
  - dConversion: If refinery-processed, this contains information about what conversion occurred.
  - dID: Internal reference ID for the new content item.
  - IsWorkflow: Returns 1 (*true*) if item entered workflow upon checkin.
- Response Templates:
  - REDIRECT_TEMPLATE (redirect_template.htm)
  - Default redirect service: CHECKIN_CONFIRM_FORM

Used By

- Resource Includes: std_doc_page_definitions

Example

```
IdcService=CHECKIN_SEL
dDocName=test_000036
dDocTitle=my test
dDocAuthor=sysadmin
```

```
dDocType=ADENG
dSecurityGroup=Public
dID=49
dRevLabel=1
doFileCopy=1
```

See Also

– *CACHE_CHECKIN_SEL (page 5-24)*

– *CHECKIN_SEL_SUB (page 5-47)*

CHECKIN_SEL_FORM

Service that returns the check-in form for a content item revision in a browser. It loads the current content information for the item. The form page is displayed with old metadata information.

The most likely error is when the content item is no longer in the system, information about the content item can not be found, or when the system is unable to check revision properties.

Access Level: Write, Scriptable (34)

Queries Executed: QdocID, Qrevisions, QdocInfo

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dID: The generated content item revision ID.

Optional Service Parameters

❖ dDocName

❖ dWfName

Results

❖ Result Sets:

- DOC_INFO (All fields from Revisions and DocMeta for the specified revision.)
- DocFormats (dFormat, dConversion, dDescription)
- DocTypes (dDocType, dDescription, dGif)

- Local Data:
  - CurRevCheckoutUser
  - CurRevID
  - CurRevIsCheckedOut
  - dDocAccount
  - dDocAuthor
  - dDocName
  - defaultAccount
  - dID
  - DocUrl
  - dPublishState
  - dReleaseState
  - dRevClassID
  - dRevLabel
  - dSecurityGroup
  - dStatus
  - dUser
  - dWorkflowState
  - isCurRevEmpty
  - IsNotLatestRev
  - IsWorkflow
  - latestID
  - Plus all DocMeta fields
- Response Template: CHECKIN_SEL_FORM (checkin_sel.htm)

Used By

- Resource Includes:
  - classic_info_page_content
  - workflow_action_popup
  - checkin_list_action_popup
  - legacy_checked_out_content_table

- workflow_doc_action_links
- docinfo_checkin_similar_option
- setup_checked_out_content_action_popups
- wf_reviewer_doc_action_links

❖ Templates:

- CHECKOUT_OK (chkook.htm)
- WORKFLOW_REVIEW_FORM (workflow_review_form.htm)

Example

```
IdcService=CHECKIN_SEL_FORM
dID=55
```

CHECKIN_SEL_SUB

Subservice used if a content item being checked in already exists, is checked out, or if it is not in a workflow.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_SEL (page 5-40)*

– *CHECKIN_UNIVERSAL (page 5-48)*

CHECKIN_SIMILAR_FORM

Service that returns the check-in form for a new content item in a browser, with metadata from a similar content item filled in. The most likely error is when the content item was not properly defined.

Access Level: Write, Global, Scriptable (50)

Queries Executed: QdocInfoSimilarCheckin

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results

❖ Result Sets:

- DOC_INFO_SIMILAR (dDocTitle, dDocType, dSecurityGroup, dDocAccount for the specified revision)
- DocFormats (dFormat, dConversion, dDescription)
- DocTypes (dDocType, dDescription, dGif)

❖ Local Data:

- dDocAccount
- dDocType
- dSecurityGroup

❖ Response Template: CHECKIN_NEW_FORM (checkin_new.htm)

Used By

❖ Resource Includes:

- search_results_action_popup
- docinfo_checkin_similar_option
- doc_file_checkin_similar
- setup_search_results_action_popups

❖ Templates: CHECKIN_CONFIRM (checkin_confirm.htm)

CHECKIN_UNIVERSAL

Service that performs a Content Server-controlled checkin.

Access Level: Write (2)

Location: <*install_dir*>/shared/config/resources/std_services.htm

❖ This service sends the check-in request to one of the following subservices, which are the same subservices called during checkin through the browser or Repository Manager application. (These subservices are not called during a Batch Loader or Archive import.)

- CHECKIN_NEW_SUB - If the content item does not exist in the server. This subservice validates the check-in data and determines if this content item belongs to a criteria workflow.
- CHECKIN_SEL_SUB - If the content item exists on the system, the content item is checked out, and the content item is not in a workflow.
- WORKFLOW_CHECKIN_SUB - If the content item exists and is in a workflow.

- This service checks security to determine if the user has sufficient permission to check in the content item.
- Determines if the content item is new or already exists in the system by querying the database using the Content ID (*dDocName*) as the key.
- If the content item exists in the system, the publish state (*dPublishState*) must be empty.
- If the item exists and is checked out, a new revision is checked in.
- The most likely errors are mismatched parameters or when the content item was not successfully checked in.

Note: All paths use the forward slash (/) as the file separator. This is because the back slash (\) is an escape character. For example, *primaryFile=d:/temp/myfile.txt*.

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
  - If Content ID auto generation is enabled, this parameter is not required.
  - If *dDocName* is specified and exists in the content server, a revision is checked in. If *dDocName* is specified and does not exist in the content server, a new content item is checked in.
  - The Content ID cannot contain spaces or invalid characters ;/ \?:@&=+"#%<>*~| [ ].
- dDocAuthor: The content item author (contributor).
- dDocTitle: The content item title.
- dDocType: The content item type.
- dSecurityGroup: The security group such as *Public* or *Secure*.
- dDocAccount: The account for the content item. Required only if accounts are enabled.
- primaryFile: The absolute path to the location of the file as seen from the server. Use the forward slash as the file separator. A primary file must be specified unless checking in metadata only. If an alternate file is specified with the primary file, the content refinery converts the alternate file. Otherwise, the primary file is converted.

- If a primary file is not specified, a metafile can be used in its place. Only one metafile can exist for each content item (a primary AND alternate meta file cannot co-exist).
- If both a primary and alternate file is specified, their extensions must be different.

❖ doFileCopy: 1 (*true*)—The file is not deleted from the hard drive after checkin. 0 (*false*)—The file is removed from the hard drive after checkin.

❖ Required custom fields: Custom metadata fields that are required must also be specified.

Optional Service Parameters

❖ alternateFile: The alternate file for conversion.

- Only one metafile can exist for each content item (a primary AND alternate meta file cannot co-exist.)
- If an alternate file is specified with the primary file, the content refinery converts the alternate file. Otherwise, the primary file is converted.

❖ dCreateDate: The date the content item was created. By default, this is the current date.

❖ dInDate: The content release date. The date the content item is to be released to the web. By default, this is the current date. If the content release date (dInDate) is not specified, the creation date (dCreateDate) is used. This value is auto generated if it is not supplied.

❖ dOutDate: The content expiration date. By default, this is blank and does not specify an expiration date. If the content expiration date (dOutDate) is not entered, the value remains empty. This is a valid state.

❖ dRevLabel: The revision label for the content item. If set, the label is used to locate the specified revision. If a revision label (*dRevLabel*) is specified, this service checks if the content revision exists in the system; an exception is thrown if it exists.

❖ isFinished: Set to *true* (1) if this is a workflow checkin and you have finished editing it.

See WORKFLOW_CHECKIN (page 6-45) for additional information.

Note: Do not confuse the Content ID (dDocName) with the internal content item revision identifier (dID). The *dID* is a generated reference to a specific rendition of a content item.

Results

- Local Data:
  - dConversion: If refinery-processed, contains information about what conversion occurred.
  - dID: Internal reference ID for the new content item.
  - IsWorkflow: Returns 1 (*true*) if item entered workflow upon checkin.

Used By

- Resource Includes: std_doc_page_definitions
- Other:
  - SoapCustom:Esdl:CheckIn:Services
  - SoapCustom:Wsdl:Workflow:Services

Example

- IdcCommand command file format:

```
IdcService=CHECKIN_UNIVERSAL
dDocName=adsales
dDocType=ADACCT
dDocTitle=Advertising Sales
dSecurityGroup=Secure
dDocAuthor=user1
dRevLabel=1
dDocType=ADACCT
primaryFile=c:/temp/docs/mydoc.txt
doFileCopy=1

# If this is a workflow check-in and you have finished editing it, mark it as
finished.
#isFinished=true

#Required depending on configuration
dDocAccount=mainaccount

#Optional fields:
#dCreateDate=
```

```
#dInDate=
#dOutDate=
#alternateFile=

#Custom metadata fields:
xComments=
xLocation=
xProjects=
```

❖ HDA format (check in the content item *myDocName*):

```
@Properties LocalData
IdcService=CHECKIN_UNIVERSAL
doFileCopy=1
dDocName=myDocName
dDocTitle=My document title
dDocType=ADACCT
dSecurityGroup=Secure
dDocAuthor=user1
primaryFile=c:/temp/docs/mydoc.txt
dDocAccount=mainaccount
xComments=
xLocation=
xProjects=
@end
```

See Also

– *CHECKIN_NEW_SUB (page 5-38)*
– *CHECKIN_SEL_SUB (page 5-47)*
– *WORKFLOW_CHECKIN_SUB (page 6-11)*

CHECKOUT

Service that checks out the latest revision of a content item from a browser.

❖ The service fails if the content item does not exist in the system, if the content item is already checked out, or if the user does not have sufficient permission to check out the content item.

- ❖ The most likely error is when the content no longer exists in the system or when the system is unable to retrieve revision information.
- ❖ This service executes the subservice CHECKOUT_SUB. This subservice checks out a content item revision.

Access Level: Write, Scriptable (34)

Calls Subservice: CHECKOUT_SUB

Queries Executed:

- QdocID
- Qrevisions
- QdocInfo

Location: <*install_dir*>/shared/config/resources/std_services.htm

Note: This service only marks the content item as locked. It does not perform a download.

Additional Required Service Parameters

- ❖ dID: The generated content item revision ID.

Optional Service Parameters

- ❖ dDocTitle: The content item title.
- ❖ RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.

Results

- ❖ Local Data:
  - CurRevCheckoutUser
  - CurRevIsCheckedOut
  - CurRevID
  - dAction
  - dActionDate
  - dActionMillis
  - dCheckoutUser

- dClbraName
- dDocName
- dIsCheckedOut
- dPublishState
- dReleaseState
- dRevClassID
- dRevLabel
- dSecurityGroup
- dStatus
- dWorkflowState
- isCurRevEmpty
- isFinished
- IsNotLatestRev
- IsWorkflow
- latestID
- prevReleaseState
- RedirectParams
- wfAction

❖ Response Templates:

- REDIRECT_TEMPLATE (redirect_template.htm)
- Default redirect service: CHECKOUT_OK

Used By

❖ Applets:

- Repository Manager
- Workflow Administrator

❖ Resource Includes

- classic_search_result_item_checkout
- clbra_active_doc_display
- doc_file_checkout
- docinfo_checkout_form

- searchapi_result_table_content_end
- search_results_action_popup
- setup_search_results_action_popups
- wf_reviewer_doc_action_links
- wips_list_action_popup
- workflow_action_popup
- workflow_doc_action_links
- workflow_docs_table
- workflow_in_queue_table
- work_in_progress_table

❖ Template: WORKFLOW_REVIEW_FORM (workflow_review_form.htm)

❖ Other: SoapCustom:Wsdl:CheckIn:Services

Example

❖ IdcCommand command file format:

```
IdcService=CHECKOUT
dID=55
```

❖ HDA format with optional parameter:

```
@Properties LocalData
IdcService=CHECKOUT
dID=55
dDocTitle=Sample Title
@end
```

See Also

– *CHECKOUT_SUB (page 5-57)*

CHECKOUT_BY_NAME

Service that checks out a content item from an applet or application.

Location: <*install_dir*>/shared/config/resources/std_services.htm

- The service fails if the content item does not exist in the system, if the content item is already checked out, or if the user does not have sufficient permission to check out the content item.
- The most likely error is a content item name not in the system.
- This service executes the subservice CHECKOUT_SUB. This subservice checks out a content item revision.

Note: This service only marks the content item as locked. It does not perform a download.

Additional Required Service Parameters

- dDocName: The Content ID of the content item.

Optional Service Parameters

- dDocTitle: The content item title.

Example

- IdcCommand command file format:

```
IdcService=CHECKOUT_BY_NAME
dDocName=myDocument
```

- HDA format with optional parameter:

```
@Properties LocalData
IdcService=CHECKOUT_BY_NAME
dDocName=myDocument
dDocTitle=Just a title
@end
```

CHECKOUT_OK

Called as a redirect service by the service(s) that check out a content item from a browser and display a checkout confirmation page.

Access Level: Write, Scriptable (34)

Queries Executed: QdocInfo

Location: <*install_dir*>/shared/config/resources/std_services.htm

- The most likely error is when the revision no longer exists or when the system is unable to retrieve revision information.
- This service executes the subservice CHECKOUT_SUB. This subservice checks out a content item revision.

Additional Required Service Parameters

- dID: The generated content item revision ID.
- CurRevCheckoutUser: The user who checked out the item.

Optional Service Parameters

- CurRevID: The ID of the current revision, usually the same as dID.

Results

- Result Sets: DOC_INFO (All Fields from Revisions and DocMeta for the specified version.)
- Response Template: CHECKOUT_OK (chkook.htm)

Used By

- Other: Redirect service for CHECKOUT, CHECKOUT_BY_NAME

Example

```
IdcService=CHECKOUT_OK
dID=48
```

See Also

– *CHECKOUT_SUB (page 5-57)*

CHECKOUT_SUB

Subservice that checks out a content item revision. Requires that the binder contain a result set named DOC_INFO that contains the results of *QdocInfo* (or equivalent) for the most recent version of a content item.

Access Level: Subservice (N/A)

Queries Executed:

- QworkflowDocInfo
- IdocHistory
- UcheckoutRevision

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results

❖ Local Data:

- CurRevCheckoutUser
- dAction
- dActionDate
- dActionMillis
- dCheckoutUser
- dClbraName
- dIsCheckedOut
- hasUserAccessChanged
- isCurRevEmpty
- isFinished
- IsNotLatestRev
- prevReleaseState
- wfAction

Used By

❖ Services:

- CHECKOUT
- CHECKOUT_BY_NAME
- FORM_PROCESS (indirectly via processForm method)

See Also

– *CHECKOUT (page 5-52)*

– *CHECKOUT_OK (page 5-56)*

CONTINUE_CHECKIN

Service that completes a checkin that was started with the CACHE_CHECKIN_NEW or CACHE_CHECKIN_SEL service.

After a preview has been performed and the check-in data is cached the user can decide to finish or continue the checkin using this service.

The cached data must be provided to this service to perform a successful checkin. Filters are provided to customize the cache and check-in behavior.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

See Also

- *CACHE_CHECKIN_NEW (page 5-24)*
- *CACHE_CHECKIN_SEL (page 5-24)*

CONTINUE_SUBMIT_HTML_FORM

Service that completes a checkin that was started with the CACHE_SUBMIT_HTML_FORM service.

After a preview has been performed and the check-in data is cached the user can decide to finish or continue the checkin using this service.

The cached data must be provided to this service to perform a successful checkin. Filters are provided to customize the cache and check-in behavior.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

See Also

- *CACHE_SUBMIT_HTML_FORM (page 5-24)*

CREATE_SUBSCRIPTION_TYPE

Service that creates a new subscription type. The most likely error is when the subscription type name is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- scpType: The subscription type.
- scpDescription: The subscription description.
- scpEnabled: 1 (*true*)—Subscription notifications are enabled. 0 (*false*)—Subscription notifications are disabled.
- scpFields: The metadata fields that define the subscription criteria. For example, *dDocAuthor, dDocType*.

Example

```
IdcService=CREATE_SUBSCRIPTION_TYPE
scpType=subscription_test
scpFields=dDocType
scpDescription=testing the subscription
scpEnabled=1
```

DELETE_BYCLASS

Subservice used to delete content items based on specific parameters.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_ARCHIVE (page 7-6)*

DELETE_BYNAME

Subservice used to delete content items based on specific parameters.

Location: <*install_dir*>/shared/config/resources/std_services.htm

DELETE_BYREV

Subservice used to delete content items based on specific parameters.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_ARCHIVE (page 7-6)*

DELETE_BYREV_REVISION

Location: *install_dir*>/shared/config/resources/std_services.htm

DELETE_CHECKIN_CACHE

Subservice used to delete content items based on specific parameters.

Location: <*install_dir*>/shared/config/resources/std_services.htm

DELETE_DOC

Service that deletes an existing content item revision. The user must have permission to delete the specified revision. The most likely error is when the content item no longer exists in the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dID: The generated content item revision ID.

Example

```
IdcService=DELETE_DOC
dID=48
```

DELETE_REV

Service that deletes an existing content item revision from a browser. The most likely errors are mismatched parameters, when the content item no longer exists in the system, or when the content item is part of a workflow.

Access Level: Read, Write (3)

Calls Subservice: DELETE_BYREV_REVISION

Queries Executed: QdocInfo

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Results

- Result Sets: DOC_INFO (All fields from Revisions and DocMeta for the specified revision.)
- Local Data:
  - Content Server:
    - currentReleaseState
    - dAction (returns 'Delete Revision')
    - dReleaseState
    - isAllRevisionsDeleted (returns true when the last revision of an item is deleted)
    - isCurRevEmpty
    - newReleaseState
    - prevID
  - If the item being deleted is currently in a workflow, the operation fails and the following workflow info is returned:
    - dWfCurrentStepID
    - dWfDocState
    - dWfID
    - dWfName

- dWfStatus
- dWfStepDescription
- dWfStepID
- dWfStepIsAll
- dWfStepName
- dWfStepType
- dWfStepWeight
- dWfType
- dWorkflowState
- IsWorkflow
- wfStepCheckinType

- Response Templates:
  - REDIRECT_TEMPLATE (redirect_template.htm)
  - Default redirect service: DOC_INFO

Used By

- Resource Includes:
  - delete_rev_form
  - delete_revision_form

Example

```
IdcService=DELETE_REV
dID=51
```

DELETE_REV_EX

Service that deletes an existing content item revision from an applet or application. The most likely errors are mismatched parameters, when the content item no longer exists n the system, or when the content item is part of a workflow.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

```
IdcService=DELETE_REV_EX
dID=56
```

DELETE_SUBSCRIPTION_TYPE

Service that deletes an existing subscription type. The most likely error is a subscription type that is not in the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- scpType: The subscription type.

Example

```
IdcService=DELETE_SUBSCRIPTION_TYPE
scpType=subscription_test
```

DOC_FORMATS_WIZARD

Service that retrieves content item format extensions. Used to load the content item configuration information and set the default content item format extension.

Location: <*install_dir*>/shared/config/resources/std_services.htm

DOC_INFO

Service that retrieves content item information for a specific revision of a content item. The most likely errors are when the content item no longer exists in the system or when the user does not have the security level to perform this action.

Limited information is provided for other revisions via the REVISION_HISTORY result set.

This service returns subscription information and workflow information if the item is in workflow.

Access Level: Read, Scriptable (33)

Queries Executed:

- QdocInfo
- QdocFormats
- QisSubscribed
- QrevHistory

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dID: The generated content item revision ID.

Results

❖ Result Sets:
  - DOC_INFO (all fields from Revisions and DocMeta for the specified revision)
  - REVISION_HISTORY (dFormat, dInDate, dOutDate, dStatus, dProcessingState, dRevLabel, dID, dDocName, dRevisionID for all non-deleted revisions)
  - If item is in workflow: WF_INFO (all fields from Workflows and WorkflowDocuments for the specified revision)

❖ Local Data:
  - Content Server:
    ❖ AuthorAddress
    ❖ dDocFormats
    ❖ dDocTitle
    ❖ dID
    ❖ DocUrl
    ❖ dStatus
    ❖ dSubscriptionAlias
    ❖ dSubscriptionID
    ❖ dSubscriptionType
    ❖ dUser
  - If item is in workflow:
    ❖ dWfCurrentStepID
    ❖ dWfDocState

- dWfID
- dWfName
- dWfStatus
- wfStepCheckinType
- dWfStepDescription
- dWfStepID
- dWfStepIsAll
- dWfStepName
- dWfStepType
- dWfStepWeight
- dWfType,
- IsWorkflow

- Response Template: DOC_INFO (doc_info.htm)

Used By

- Resource Includes:
  - checkin_list_action_popup
  - checkin_multiuploadapplet_processing_functions
  - classic_doc_rev_info
  - clbra_active_doc_display
  - clbra_wf_doc_list
  - dam_result_table_content_row
  - doc_info_action_image
  - docinfo_page_title
  - doc_rev_info
  - doc_revisions_table
  - email_docinfo_body_by_id
  - email_docinfo_body_by_name
  - legacy_basic_subscriptions_table
  - legacy_checked_out_content_table
  - legacy_expired_content_table

- legacy_work_in_progress_table
- my_view_result_table_cells
- searchapi_result_doc_href_start
- searchapi_result_table_content_end
- search_result_info_button
- search_results_action_popup
- setup_checked_out_content_action_popups
- setup_search_results_action_popups
- setup_subscription_action_popups
- setup_work_in_progress_action_popups
- slim_result_table_content_row
- subscription_action_popup
- trays_search_result_section
- wf_in_queue_display
- wf_reviewer_mail_link
- wips_list_action_popup

❖ Templates:

- CHECKIN_CONFIRM (checkin_confirm.htm)
- DOCUMENT_PROBLEMREPORTS (doc_problemreports.htm)
- DOC_SUB_LIST (doc_sub_list.htm)
- PR_CONTRIBUTOR_MAIL (pr_contributor_mail.htm)
- PROBLEMREPORT_INFO (pr_info.htm)
- QUERY_NOTIFICATION (query_notification_mail.htm)
- SUBSCRIPTION_MAIL (subscription_mail.htm)
- WORKFLOW_REVIEW_FORM (workflow_review_form.htm)
- WORKFLOW_REVIEW_FRAMES (workflow_review_frames.htm)

❖ Other:

- Link in LC Variable: wwCommentOnRevision
- SoapCustom:Wsdl:DocInfo:Services
- SoapCustom:Wsdl:DocInfo:Service:DocInfoByID:ResponseParams
- SoapCustom:Wsdl:DocInfo:Service:DocInfoByName:ResponseParams

- SoapCustom:Wsdl:GetFile:Service:GetFileByID:ResponseParams
- Redirect service for:
  - DELETE_REV
  - RESUBMIT_FOR_CONVERSION
  - SUBSCRIBE
  - UNDO_CHECKOUT
  - UNSUBSCRIBE
  - UPDATE_DOCINFO_BYFORM

Example

```
IdcService=DOC_INFO
dID=54321
```

DOC_INFO_BY_NAME

Service that retrieves information about the latest revision of a content item based on the content ID (the dDocName) as a parameter rather than the revision-specific dID parameter, which is required by the DOC_INFO service. The *RevisionSelectionMethod* parameter can be set to *specific* to return information about a specific revision.

By default, this service returns information for the Latest revision, whether it is released or not. An item is not released until it is out of workflow. Results returned for a given revision are identical to those of the DOC_INFO service.

Access Level: Read, Scriptable (33)

Queries Executed:

- QdocInfo
- QdocFormats
- QisSubscribed
- QrevHistory
- QlatestIDByName (if RevisionSelectionMethod is empty or 'Latest')
- QlatestReleasedIDByName (if RevisionSelectionMethod is 'LatestReleased')

Location: <*install_dir*>/shared/config/resources/std_services.htm

Important: Either the content item name (*dDocName*) must be specified or a Content ID (*dDocName*) with the *RevisionSelectionMethod* parameter must be specified.

Additional Required Service Parameters

- dDocName: The content item name.

Optional Service Parameters

- RevisionSelectionMethod: Can be set to *Latest* to retrieve the most recent version, or *LatestReleased* to retrieve the most recently released version, or *Specific* (if set to *Specific*, a dID must be provided). If set to *Specific*, dID can be used instead of dDocName to point to a specific revision.

Results

- Result Sets:
  - DOC_INFO (all fields from Revisions and DocMeta for the specified revision)
  - REVISION_HISTORY (dFormat, dInDate, dOutDate, dStatus, dProcessingState, dRevLabel, dID, dDocName, dRevisionID for all non-deleted revisions)
  - If item is in workflow: WF_INFO (all fields from Workflows and WorkflowDocuments for the specified revision)
- Local Data:
  - Content Server:
    - AuthorAddress
    - dDocFormats
    - dDocTitle
    - dID
    - DocUrl, dStatus
    - dSubscriptionAlias
    - dSubscriptionID
    - dSubscriptionType
    - dUser
  - If item is in workflow:
    - dWfStatus

- dWfCurrentStepID
- dWfDocState
- dWfID
- dWfName
- dWfStepDescription
- dWfStepID
- dWfStepIsAll
- dWfStepName
- dWfStepType
- dWfStepWeight
- dWfType
- IsWorkflow
- wfStepCheckinType

- Response Template: DOC_INFO (doc_info.htm)

Used By

- Resource Includes:
  - email_docinfo_body_by_id
  - email_docinfo_body_by_name
  - setup_workflow_action_popups
  - workflow_action_popup
- Template: PR_PUBLISHER_MAIL (pr_publisher_mail.htm)
- Other: SoapCustom:Wsdl:DocInfo:Services

DOC_INFO_LATESTRELEASE

Service that retrieves information only for the latest revision of a released content item. If the content item has no released revision, the service returns a StatusCode of -1, with a corresponding StatusMessage value. It only returns the DOC_INFO result set (no revision, workflow, or subscription information). Because it does not have a template it returns only raw data.

The most likely errors are when the content item no longer exists in the system or when the user does not have the security level to perform this action.

Access Level: Read, Scriptable (33)

Queries Executed: QlatestReleasedIDByName, QdocInfo

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The document name for the content item.

Results

- Result Sets: DOC_INFO (All fields from Revisions and DocMeta for the latest released revision.)
- Local Data:
  - AuthorAddress
  - dDocName
  - dID
  - dUser

Example

```
IdcService=DOC_INFO_LATESTRELEASE
dDocName=PublicDoc1_ia3c488971
```

DOC_INFO_SIMPLE

Service that returns information about a specific content item without workflow or subscription information. It returns revision information as a simple listing of rows from the Revisions table rather than a specific selection of fields from the Documents and Revisions table. These differences make this service a less expensive operation than the standard DOC_INFO service.

Because it does not have a template it returns only raw data.

Access Level: Read, Scriptable (33)

Queries Executed: QdocInfo, QdocName

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dID: The Content ID for the content item.

Results

❖ Result Sets:

- DOC_INFO (all fields from Revisions and DocMeta for specific revision)
- DocRevisions (all fields from Revisions for all non-deleted revisions)

❖ Local Data:

- AuthorAddress
- dID
- dUser

DOC_INFO_SIMPLE_BYREV

Service that returns information about a content item based on a revision number. This service differs from others by requiring a combination of Content ID (dDocName) and revision number (dRevLabel) as required parameters. All other revision-specific doc_info services require dID.

This service returns the DOC_INFO result set. DOC_INFO_SIMPLE also returns revision information.

Access Level: Read, Scriptable (33)

Queries Executed: QdocRev, QdocInfo

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dDocName: The document name for the content item.

❖ dRevLabel: The revision number for the content item.

Results

❖ Result Sets:

- DOC_INFO (all fields from Revisions and DocMeta for specific revision)

❖ Local Data:

- dUser
- dRevLabel
- dID
- AuthorAddress
- dDocName

DOC_SUBS_LIST

Service that returns a list of content items in a specific subscription.

For criteria-based subscriptions, this returns items matching the criteria. For basic subscriptions, it returns the item with a name that matches the subscription ID.

Access Level: Read, Global, Scriptable (49)

Queries Executed: QuserSubscription

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dSubscriptionID: The subscription ID. For a Basic subscription, this is the Content ID. For a Criteria subscription, this is a comma-separated list of the values of the criteria fields.

For example, if the criteria fields are Author and Security Group, the value for a Public content item authored by user1 would be *Public,user1*.

❖ dSubscriptionType: The subscription type, either *Basic* or a custom name. Must be set to *Basic* for a basic subscription.

Optional Service Parameters

❖ dSubscriptionAlias: This can be used to pass an alias or user name. If set to *alias*, a valid alias name must be used. If set to *user* a valid user name should be used. If an invalid alias or user name is used, an error occurs. If left blank, the value defaults to *user* and is set to the current user (dUser).

❖ dSubscriptionAliasType: Determines what the service expects as a value for *dSubscriptionAlias*. Valid values are *alias* or *user*.

❖ unsubscribeService: For response pages (such as the default template for this service) that provide a link that allows the user to unsubscribe, a value must be passed as a

parameter. If omitted an error does not appear but the unsubscribe link on the response page produces an error if clicked (the standard value should be UNSUBSCRIBE).

- subscribeService: Same as *unubscribeService* except a link to subscribe is provided if the user is not already subscribed to the subscription (the standard value should be SUBSCRIBE).
- MaxQueryRow: Truncates the number of rows returned for the DOCUMENT_LIST result set.

Results

- Result Sets:
  - DOCUMENT_LIST: All fields from Revisions and DocMeta for items matching the subscription criteria. This data is returned regardless of whether the user is subscribed to the specified subscription.
  - USER_SUBSCRIPTION: All fields from Subscriptions for the row that assigns the specified subscription to the user/alias. If the user/alias isn't subscribed to the specified subscription, the result set is empty.
- Local Data:
  - dataSource
  - dSubscriptionAlias
  - dSubscriptionAliasType
  - dSubscriptionID
  - dSubscriptionType
  - dUser
  - MaxQueryRows
  - resultName
  - scpDescription
  - scpEnabled
  - whereClause
- Response Template: DOC_SUB_LIST (doc_sub_list.htm)

Used By

- Applets: Repository Manager
- Resource Includes:

- criteria_subscription_info_image
- legacy_criteria_subscriptions_table
- setup_subscription_action_popups
- subscription_action_popup

❖ Templates:

- SUBSCRIBE_FORM (subscribe_form.htm)
- UNSUBSCRIBE_FORM (unsubscribe_form.htm)

Example

```
IdcService=DOC_SUBS_LIST
dSubscriptionType=my_subscription
dSubscriptionID=ADENG
```

EDIT_DOC_FORMATS

Service that modifies existing file formats. The service updates content item formats and file extension information. This service is executed in a browser interface by the Inbound Refinery.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dFormat: The formatting process used to create the web-viewable version of the content. For example, *application/hcsf* or *application/doc*.

❖ dDescription: The format description.

❖ dConversion: The conversion algorithm is determined by this parameter.

If the conversion type is not *Passthru* or *NoConversion* (default value) and the content is not web-viewable or has a publish state, the content gets added to the queue to get converted. Otherwise, the server looks for the web-viewable (if applicable) and the vault file and updates the processing state to say that the file has already been converted.

If the file mime-type is *application/FDF*, the server sets the conversion to *exchange-fdf*.

❖ extensions: The file extensions such as *doc*, *txt*, or *pdf*.

Example

```
IdcService=EDIT_DOC_FORMATS
dFormat=application/msword
extensions=doc
dConversion=PASSTHRU
dDescription=adding the description
```

See Also

– *EDIT_DOCFORMAT (page 5-9)*

FORM_PROCESS

Service that processes a PDF form for submission. This service is called by FORM_SUBMIT. It uses the auto-generated *ClientId* parameter.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Note: This service cannot be executed by IdcCommand.

See Also

– *FORM_SUBMIT (page 5-76).*

FORM_SUBMIT

Service that submits a PDF form. This service calls the FORM_PROCESS (page 5-76) service to process the PDF form before submission.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Note: This service cannot be executed by IdcCommand.

See Also

– *FORM_PROCESS (page 5-76).*

GET_CACHED_CHECKIN_INFO

Service used to display the cached data for a cache checkin such as CACHE_CHECKIN_NEW. It is available from the checked-out content listing.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dDocName: The Content ID for the content item.

See Also

– *CACHE_CHECKIN_NEW (page 5-24)*

– *CACHE_CHECKIN_SEL (page 5-24)*

– *CACHE_SUBMIT_HTML_FORM (page 5-24)*

GET_DOC_CONFIG_INFO

Service that returns content server configuration information for Content Publisher purposes. Gets a dump of the server environment form external applications.

Returns the properties information, the security group option list, and these ResultSets:

- DOC_DEFAULT_INFO
- DocMetaDefinition
- DocTypes
- DocFormats

Location: <*install_dir*>/shared/config/resources/std_services.htm

Example

❖ IdcCommand command file format:

```
# Retrieves content item configuration information
IdcService=GET_DOC_CONFIG_INFO
```

❖ HDA format:

```
@Properties LocalData
IdcService=GET_DOC_CONFIG_INFO
@end
```

GET_DOC_SUBSCRIBERS

Service that returns a list of subscribers to a specific content item.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

Example

```
IdcService=GET_DOC_SUBSCRIBERS
dDocName=test_000036
```

GET_EXPIRED

Service used to retrieve a list of expiring and expired items. The optional *isExpiredQuery* parameter, when set to *true*, causes the service to list only items that have already expired.

Without this parameter, the list also contains items that are scheduled to expire within the range of dates specified by endDate and startDate. This may include items that have already expired if the start date is earlier than the current system time.

Access Level: Read, Write, Global (18)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Optional Service Parameters

- endDate: Items expiring before this date are displayed.
- isExpiredQuery: Setting this parameter to 1 (*true*) causes searching for only expired items. If set to *true* and the end date is unspecified, the current system time is used as a default end date. This lists content that has already expired unless constrained by a specific start date.

  If set to 0 (*false*) and the start date is empty, the current system time is used as a default start date. This lists content items that expire in the future, constrained by a specified end date.

  In all other situations, empty start or end dates are treated as empty, and no corresponding constraint is imposed upon the query.
- startDate: Items expiring after this date are displayed.

Results:

- Results Sets: EXPIRED_LIST (All fields from Revisions and DocMeta for expired or expiring items as defined by the dates and *isExpiredQuery* parameter.)
- Local Data:
  - dataSource
  - endDate
  - expiredQuery
  - isExpiredQuery
  - isQueryResult
  - MaxQueryRows
  - orderClause
  - resultName
  - startDate
  - userDefinedEndDate
  - userDefinedStartDate
  - whereClause
- Response Template: EXPIRED_PAGE (expired_page.htm)

Used By

- Resource Includes:
  - pne_nav_management_links
  - std_doc_man_pages
- Templates: EXPIRED_PAGE (expired_page.htm)
- Standard Navigation: commonNav.js

GET_PACKAGE_ENVIRONMENT_PAGE

Used to retrieve the page that initiates the packaging of the environment.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_UPDATE_FORM

Service used to generate the Info Update Form for an existing content item.

The most likely error is naming a content item that is no longer in the system.

Access Level: Write, Scriptable (34)

Queries Executed: QdocInfo

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Results

- Result Sets:
  - DOC_INFO (all fields from Revisions and DocMeta for the specified revision)
  - DocFormats (all rows and fields of the DocFormats database table)
  - DocTypes (all rows and fields of the DocTypes database table)
- Local Data:
  - dDocAccount
  - dDocName
  - dDocType
  - defaultAccount
  - dID
  - dSecurityGroup
  - dUser
- Response Template: UPDATE_DOC_INFO (update_docinfo.htm)

Example

```
IdcService=GET_UPDATE_FORM
dID=59
```

NOTIFY_INDEXER

Service used to notify the indexer that work is to be done. This service is not used because normally a checkin notifies the indexer or because the indexer has its own timer, which checks for "work to do" every five minutes.

This service should only be used if a NO_NOTIFY checkin has been performed and the client application does not want to wait for the normal indexing cycle.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_ARCHIVE_NO_NOTIFY (page 5-25)*

ODMA_DOC_INFO_SIMPLE

Service used by the ODMA client application to retrieve the content information for the specified content item.

This service is currently not in use.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dID: The unique ID for the item revision.

PACKAGE_ENVIRONMENT

Service that creates a packaged zip file containing content server environment files.

Location: <*install_dir*>/shared/config/resources/std_services.htm

REMOVE_METAFILE_SUB

Subservice called to alter document meta information.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *UPDATE_DOCINFO_METAFILE_BYREV (page 5-109)*

– *UPDATE_DOCINFO_SUB (page 5-110)*

REPLACE_METAFILE_SUB

Subservice called to replace document meta information.

Location: <*install_dir*>/shared/config/resources/std_services.htm

RESUBMIT_FOR_CONVERSION

Service that resubmits a content item for conversion when it fails the initial conversion attempt. The most likely errors are when the content item no longer exists in the system or when the user does not have the security level to perform this action.

This service can resubmit successful conversions as well as failed ones.

Access Level: Read, Scriptable (33)

Queries Executed: QdocInfo, QlatestID, Qdocuments

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Optional Service Parameters

- RedirectUrl: URL used for the redirected service.
- AlwaysResubmit: When set to 1 (*true*), allows the service to be used on items that were successfully converted. Otherwise the service only operates on failed conversions.

Results

- Local Data:
  - dConversion
  - dCurRevID
  - dDocID
  - dExtension
  - dFileSize

- dFormat
- dID
- dOriginalName
- dProcessingState
- dPublishState
- dRawDocID
- dReleaseState
- dRevClassID
- dStatus
- dWorkflowState
- IsNotLatestRev
- isStatusChanged
- prevReleaseState
- RedirectParams
- VaultfilePath
- WebfilePath
- wfAction

❖ Response Template:

- REDIRECT_TEMPLATE (redirect_template.htm)
- Default redirect service: DOC_INFO

Used By

❖ Applets: Repository Manager

❖ Resource Includes:

- classic_info_page_content
- docinfo_resubmit_option
- std_docinfo_error_msg

Example

```
IdcService=RESUBMIT_FOR_CONVERSION
dID=62
```

REV_HISTORY

Service that returns the revision history for a content item revision.

Access Level: Read, Scriptable (33)

Queries Executed: QdocInfo, QrevHistoryReleased

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Results

- Result Sets:
  - DOC_INFO (All fields from Revisions and DocMeta for the specified revision)
  - REVISIONS (Rows listing all the item's released revisions containing the fields: dDocAccount, dDocName, dDocType, dFormat, dID, dInDate, dProcessingState, dReleaseState, dRevLabel, dRevisionID, dSecurityGroup, dStatus, dWebExtension)
- Local Data: dID
- Response Template: REV_HISTORY (rev_history.htm)

Used By

- Resource Includes:
  - classic_search_result_item_revision
  - searchapi_result_table_content_end

Example

```
IdcService=REV_HISTORY
dID=62
```

SELECTDOC

Service that returns content item information and evaluates the user security level when selecting a content item using ODMA.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

```
IdcService=SELECTDOC
dID=62
```

SUBMIT_HTML_FORM

Service that submits HCSP or HCSF forms for processing. These are Dynamic Server Pages used to process HTML-based business forms. This service calls the GET_FILE subservice, called with Java code from intradoc.server.FormHandler.retrieveHtmlFormState() method.

Usually the Auto Content ID feature is enabled to submit HTML forms. If not, each submitted form must be assigned a unique Content ID.

The most likely error is an HTML form that does not exist.

Access Level: Write (2)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID. The specified revision must be an HTML form.

Optional Service Parameters

- RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.
- isFormFinished: Used on HCSP files. If this is set to 1 (*true*), the HCSP cannot be submitted again.
- Text fields: You can submit any text field in the form as an optional parameter and pass a value to that field. For example, if the form has a text field called *Product*, the string value *publisher* could be assigned to that field (`Product=publisher`).

Results

- Result Sets: DocTypes, DocFormats
- Local Data: dDocTitle, dID
- Response Template: null, default redirect server (GET_DOC_PAGE: HOME_PAGE)

Used By

- Resource Includes: std_html_form_submit_start

Example

- IdcCommand command file format:

```
IdcService=SUBMIT_HTML_FORM
dID=44
```

- HDA format with an optional parameter (assigns a value to a text field):

```
@Properties LocalData
IdcService=SUBMIT_HTML_FORM
dID=44
Product=publisher
@end
```

See Also

– *CACHE_SUBMIT_HTML_FORM (page 5-24)*

SUBSCRIBE

Service that subscribes a user to a content item or group of items. The most likely errors are when the content item no longer exists in the system or when the user does not have the security level to perform this action.

Access Level: Read (1)

Queries Executed:

- QdocInfo
- QuserSubscription
- Quser
- UuserEmail
- Isubscription

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.
- dSubscriptionEmail: The e-mail address for the subscription. The service uses the value to overwrite the user's e-mail value in their user profile. If no value is passed for this parameter, the profile not updated. Essentially, every time this service is executed, the user's e-mail address is updated.
- dSubscriptionType: The subscription type. You can optionally provide the name of a custom subscription. If this parameter is used, the service subscribes the user to a criteria-based subscription where the specified custom subscription's criteria fields match the value of the corresponding metadata of the item specified via the dID parameter. If dSubscriptionType is left blank, the Basic subscription is used by default.

Optional Service Parameters

- RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.

Results

- Local Data:
  - dID
  - dName
  - dSubscriptionAlias
  - dSubscriptionAliasType
  - dSubscriptionCreateDate
  - dSubscriptionID
  - dSubscriptionType
  - dUser
- Response Template:
  - DOC_INFO
  - Default redirect service: DOC_INFO

Used By

- Resource Includes:
  - docinfo_subscription_form
  - doc_subscription_unsubscription
- Other: SoapCustom:Wsdl:Subscription:Services

Example

These parameters are passed in the browser environment to execute this command:

```
<input type=hidden name=IdcService value="SUBSCRIBE">
<input type=hidden name=dID value="63">
<input type=hidden name=dSubscriptionType value=my_subscription>
<input type=hidden name=dSubscriptionEmail value=sysadmin@company.com>
```

SUBSCRIBE_DOC_USER

Service used in the browser environment when a user subscribes to a content item (builds the subscription page).

This service is executed when a user subscribes to a document (performs a search and clicks Subscribe). If the system has more than one subscription type, the user is redirected to the Subscription page where the user can subscribe to different criteria.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Note: Additional parameters are passed when the user subscribes to the content item (see example below).

Example

These parameters are passed in the browser environment to execute this command:

```
<form name=SubscriptionForm action="/intradoc-cgi/idc_cgi_isapi.dll"method="GET"">
<input type=hidden name=dID value="63">
<input type=hidden name=dDocName value="test_000045">
<input type=hidden name=IdcService value="SUBSCRIBE_DOC_USER">
<input type=hidden name=subscribeService value=SUBSCRIBE>
<input type=hidden name=exitUrl value="/intradoc-
cgi/idc_cgi_isapi.dll?IdcService=DOC_INFO&dID=63&dDocName=test_000045">
```

```
<input type=hidden name=title value="Subscriptions">
<input type=hidden name=unsubscribeService value=UNSUBSCRIBE>
<input type=submit value="Subscriptions">
</form>
```

SUBSCRIBE_EX

Service used in Repository Manager to add a user or alias to a subscription. It returns the list of users and aliases subscribed to this type. The most likely error is when the specified subscription alias type or subscription alias does not exist.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dSubscriptionAlias: The alias for the subscription.
- dSubscriptionAliasType: The type of user. This value must be either *user* or *alias*. It refers to the value defined in dSubscriptionAlias.
- dSubscriptionEmail: The email address for the subscription.
- dSubscriptionID: The subscription ID.
  - For a Basic subscription, this is the Content ID.
  - For a Criteria subscription, this is a comma-separated list of the values of the criteria fields.

  For example, if the criteria fields are Author and Security Group, the *dSubscriptionID* for a *Public* content item authored by *user1* would be *Public,user1*.
- dSubscriptionType: Name of the subscription type defined in the applet. A subscription type consists of a list of fields. When subscribing to a subscription type, the user must specify the values for each of the fields that make up the subscription type.

Optional Service Parameters

To return information about who is subscribed, this service requires these parameters:

- dataSource: The data source.
- resultName: The ResultSet name.
- whereClause: The programmatic clause.

Example

```
IdcService=SUBSCRIBE_EX
dSubscriptionID=user20
dSubscriptionAlias=sysadmin
dSubscriptionAliasType=user
dSubscriptionType=test
dSubscriptionEmail=user20@stellent.com
resultName=USER_LIST
dataSource=Subscriptions
whereClause=dSubscriptionType='test'
```

See Also

– *UNSUBSCRIBE_FROM_LIST_EX (page 5-100)*

SUBSCRIBE_FORM

Service that retrieves the subscription form used for content item subscriptions.

This service is identical to the UNSUBSCRIBE_FORM service with the exception of the template it returns.

Access Level: Read (1)

Queries Executed: QdocInfo, QSubscribed

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.
- unsubscribeService: For response pages (such as the default template for this service) that provide a link that allows the user to unsubscribe, a value must be passed as a parameter. If omitted an error does not appear but the unsubscribe link on the response page produces an error if clicked (the standard value should be UNSUBSCRIBE).
- subscribeService: Same as *unubscribeService* except a link to subscribe is provided if the user is not already subscribed to the subscription (the standard value should be SUBSCRIBE).

Results

- Result Sets:

- DOC_INFO (All fields from Revisions and DocMeta for the revision specified by dID)
- SUBSCRIPTION_LIST (All fields from the Subscription table plus the additional fields scpFields, scpDescription, scpEnabled, and notSubscribed for all defined subscriptions)

❖ Local Data:

- dID
- dSubscriptionAlias
- dSubscriptionID
- dSubscriptionType
- dUser

❖ Response Template: SUBSCRIBE_FORM (subscribe_form.htm)

Used By

❖ Resource Includes:

- docinfo_subscription_form
- doc_subscription_unsubscription

SUBSCRIPTION_LIST

Service that returns a list of all subscribed content items for a specific user.

No alias-based subscriptions are returned by this service, even if the user belongs to an alias that has subscriptions.

Access Level: Read, Global, Scriptable (49)

Queries Executed: QdocNameSubscription, QnotDocNameSubscriptions

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dUser: The user name of the currently logged-in user.

Results

❖ Result Sets: SUBSCRIPTION_LIST (Each row contains all fields from the Subscription table corresponding to each of the current user's subscriptions.

Additionally, for BASIC subscriptions, each row contains all fields from the Revisions table (metadata) for the latest revision of the item subscribed to.)

- Local Data: dUser
- Response Template: SUBSCRIPTION_LIST (subscription_list.htm)

Used By

- Resource Includes: pne_nav_userprofile_links
- Templates:
  - DOC_SUB_LIST (doc_sub_list.htm)
  - USER_INFO (user_info.htm)
- Standard Navigation: commonNav.js
- Other:
  - SoapCustom:Wsdl:Subscription:Services
  - SoapCustom:Wsdl:Subscription:Service:SubscriptionList:ResponseParams
  - Redirect service for: UNSUBSCRIBE_FROM_LIST

Example

```
IdcService=SUBSCRIPTION_LIST
dUser=sysadmin
```

UNDO_CHECKOUT

Service that reverses a content item checkout from a browser. The service fails if the content item does not exist in the system, if the content item is not checked out, or the user does not have sufficient permission to undo the checkout.

Access Level: Write (2)

- Queries Executed:
  - QdocID
  - Qrevisions
  - QdocInfo
  - IdocHistory
  - QdocWebFormat

- Uuncheckout

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dID: The generated content item revision ID.

Optional Service Parameters

❖ RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.

Results

❖ Local Data:

- CurRevCheckoutUser
- CurRevID
- CurRevIsCheckedOut
- dAction
- dCheckoutUser
- dClbraName
- dDocAccount
- dDocName
- dID
- dIsCheckedOut
- dReleaseState
- dRevClassID
- dRevLabel
- dSecurityGroup
- dStatus
- dUser
- isFinished
- IsWorkflow
- latestID

- noDocLock
- prevReleaseState
- wfAction

❖ Response Template:

- REDIRECT_TEMPLATE (redirect_template.htm)
- Default redirect service: GET_PORTAL_PAGE (Page=WF_INQUEUE_LIST)

Used By

❖ Applets:

- Repository Manager
- Workflow

❖ Resource Includes:

- checkin_list_action_popup
- doc_file_checkout
- docinfo_undo_checkout_form
- setup_checked_out_content_action_popups
- setup_workflow_action_popups
- wf_reviewer_doc_action_links
- workflow_action_popup

❖ Templates: WORKFLOW_REVIEW_FORM (workflow_review_form.htm)

❖ Other: SoapCustom:Wsdl:CheckIn:Services

Example

❖ IdcCommand command file format:

```
IdcService=UNDO_CHECKOUT
dID=44
```

❖ HDA format:

```
@Properties LocalData
IdcService=UNDO_CHECKOUT
dID=44
@end
```

UNDO_CHECKOUT_BY_NAME

Service that reverses a content item checkout from an applet or application. The service fails if the content item does not exist in the system, if the content item is not checked out, or the user does not have sufficient permission to undo the checkout.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

Optional Service Parameters

- dDocTitle: The content item title.

Example

- IdcCommand command file format:

```
IdcService=UNDO_CHECKOUT_BY_NAME
dDocName=myDocument
```

- HDA format with optional parameter:

```
@Properties LocalData
IdcService=UNDO_CHECKOUT_BY_NAME
dDocName=myDocument
dDocTitle=Just a title
@end
```

UNSUBSCRIBE

Service that cancels a content item subscription when only the Basic subscription is defined. This service is run by clicking the **Unsubscribe** button on a Content Information page.

This service is different from the UNSUBSCRIBE_FROM_LIST service primarily in what data it returns. UNSUBSCRIBE redirects to the Content Info page by default, so it loads content info for the item specified by dID. UNSUBSCRIBE_FROM_LIST redirects to the subscription listing page by default, so it loads the data necessary to display the user's subscriptions.

Access Level: Read (1)

Queries Executed: QdocInfo, Dsubscription

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.
- dSubscriptionID: The subscription ID.
  - For a Basic subscription, this is the Content ID.
  - For a Criteria subscription, this is a comma-separated list of the values of the criteria fields.

  For example, if the criteria fields are Author and Security Group, the *dSubscriptionID* for a *Public* content item authored by *user1* would be *Public,user1*.
- dSubscriptionType: The subscription type.

Optional Service Parameters

- RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.
- dSubscriptionAlias: The user or alias for the criteria subscription (set to a valid user name or alias name, depending upon the value for *dSubscriptionAliasType*).
- dSubscriptionAliasType: The alias for the subscription type. Valid values are *user* or *alias*.

If the current user has Admin privileges, the *dSubscriptionAlias* parameter with *dSubscriptionAliasType* can be used to unsubscribe a different user or alias from a subscription. If these parameters are omitted, the service defaults to a *dSubscriptionAliasType* of *user* and uses the current user (dUser) for the value of *dSubscriptionAlias*. If these parameters are included but the current user doesn't have Admin privileges, they are disregarded.

Results

- Local Data:
  - dCheckoutUser
  - dDocAccount
  - dDocName
  - dID

- dRevClassID
- dSecurityGroup
- dUser

❖ Response Template:

- DOC_INFO
- Default redirect service: DOC_INFO

Used By

❖ Resource Includes:

- docinfo_subscription_form
- doc_subscription_unsubscription

❖ Other: SoapCustom:Wsdl:Subscription:Services

Example

```
IdcService=UNSUBSCRIBE
dID=66
dSubscriptionType=test
dSubscriptionID=test_000048
```

UNSUBSCRIBE_FORM

Service that retrieves the form used when unsubscribing from a content item.

This service is identical to the SUBSCRIBE_FORM service with the exception of the template it returns.

Access Level: Read (1)

Queries Executed: QdocInfo, QisSubscribed

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dID: The generated content item revision ID.

❖ subscribeService: For response pages (such as the default template for this service) that provide a link that allows the user to unsubscribe, a value must be passed as a

parameter. If omitted an error does not appear but the unsubscribe link on the response page produces an error if clicked (the standard value should be SUBSCRIBE).

Results

- Result Sets:
  - DOC_INFO (All fields from Revisions and DocMeta for the revision specified by dID)
  - SUBSCRIPTION_LIST (All fields from the Subscription table plus the additional fields scpFields, scpDescription, scpEnabled, and notSubscribed for all defined subscriptions)
- Local Data:
  - dID
  - dSubscriptionAlias
  - dSubscriptionID
  - dSubscriptionType
  - dUser
- Response Template: UNSUBSCRIBE_FORM (unsubscribe_form.htm)

Used By

- Resource Includes:
  - docinfo_subscription_form
  - doc_subscription_unsubscription

UNSUBSCRIBE_FROM_LIST

Service that cancels a content item subscription from a browser when multiple subscriptions are defined.

This service is run from the Subscriptions page, either by clicking the **Unsubscribe** link, or by clicking the **Subscription Info** link for that content item and then clicking **Unsubscribe** on the Subscription Info page.

This service differs from the UNSUBSCRIBE service primarily in what data it returns. UNSUBSCRIBE_FROM_LIST redirects to the subscription listing page by default, so it loads the data necessary to display the user's subscriptions. UNSUBSCRIBE redirects to the Content Info page by default, so it loads content info for the item specified by dID.

Access Level: Read, Global (17)

Queries Executed:

- Dsubscription
- QdocNameSubscription
- QnotDocNameSubscriptions

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dSubscriptionID: The subscription ID. For a Basic subscription, this is the Content ID. For a Criteria subscription, this is a comma-separated list of the values of the criteria fields.

For example, if the criteria fields are Author and Security Group, the *dSubscriptionID* for a *Public* content item authored by *user1* would be *Public,user1*.

❖ dSubscriptionType: The subscription type.

Optional Service Parameters

❖ RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.

❖ dSubscriptionAlias: The user alias for the criteria subscription, set to a valid user name or alias name, depending on the value for *dSubscriptionAliasType*.

❖ dSubscriptionAliasType: The alias for the subscription type, set to either *user* or *alias*.

If the current user has Admin privileges, the *dSubscriptionAlias* parameter with *dSubscriptionAliasType* can be used to unsubscribe a different user or alias from a subscription. If these parameters are omitted, the service defaults to a *dSubscriptionAliasType* of *user* and uses the current user (dUser) for the value of *dSubscriptionAlias*. If these parameters are included but the current user doesn't have Admin privileges, they are disregarded.

Results

❖ Result Sets: SUBSCRIPTION_LIST. Each row contains all fields from the Subscription table corresponding to each of the current user's subscriptions. Additionally, for Basic subscriptions, each row contains all fields from the Revisions table (metadata) for the latest revision of the item subscribed to.

- Local Data:
  - dID
  - dSubscriptionAlias
  - dSubscriptionAliasType
  - dUser
- Response Template:
  - SUBSCRIPTION_LIST
  - Default redirect service: SUBSCRIPTION_LIST

Used By

- Applets: Repository Manager
- Resource Includes: pne_nav_userprofile_links
- Templates:
  - DOC_SUB_LIST (doc_sub_list.htm)
  - USER_INFO (user_info.htm)
- Standard Navigation: commonNav.js

Example

```
IdcService=UNSUBSCRIBE_FROM_LIST
dSubscriptionType=test
dSubscriptionID=test_000046
```

UNSUBSCRIBE_FROM_LIST_EX

Service used by the Repository Manager applet to remove a user or alias from the subscription type. See SUBSCRIBE_EX for information about adding a user or alias.

This service is normally run by deleting a user from a subscription in the Repository Manager applet.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dSubscriptionAlias: Name of the subscription type as defined in the applet.

- dSubscriptionAliasType: The type of user. This value must be either *user* or *alias*. It refers to the value defined in dSubscriptionAlias.
- dSubscriptionID: The subscription ID.
  - For a Basic subscription, this is the Content ID.
  - For a Criteria subscription, this is a comma-separated list of the values of the criteria fields.

  For example, if the criteria fields are Author and Security Group, the *dSubscriptionID* for a *Public* content item authored by *user1* would be *Public,user1*.
- dSubscriptionType: The subscription type.

Optional Service Parameters

To return information about who is subscribed, this service requires these parameters:

- dataSource: The data source.
- resultName: The ResultSet name.
- whereClause: The programmatic clause.

Example

Unsubscribes *user20* from any content that is checked in by *sysadmin*:

```
IdcService=UNSUBSCRIBE_FROM_LIST_EX
dSubscriptionID=sysadmin
dSubscriptionAlias=user20
dSubscriptionAliasType=user
dSubscriptionType=test
dSubscriptionEmail=sysadmin@stellent.com
resultName=USER_LIST
dataSource=Subscriptions
whereClause=dSubscriptionType='test'
```

See Also

– *SUBSCRIBE_EX (page 5-89)*

UPDATE_BYREV

Subservice used to check in content items based on specific parameters.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_ARCHIVE (page 7-6)*

UPDATE_DOCINFO

Service that updates the metadata for a content item from an applet or application. This service is used by the Repository Manager. and executes the UPDATE_DOCINFO_SUB subservice.

This service differs from the UPDATE_DOCINFO_BYFORM service which provides an option to redirect to a display template. UPDATE_DOCINFO provides no display option.

Access Level: Write (2)

Calls Subservice:

- UPDATE_DOCINFO_SUB. This subservice may also call REMOVE_METAFILE_SUB, REPLACE_METAFILE_SUB, and UPDATE_DOCINFO_STATUS.

Queries Executed (all queries executed within subservices):

- QdocInfo
- QlatestID
- Qdocuments
- Ddocument
- Idocument
- Urevision2
- Umeta
- IdocHistory

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- ❖ dDocName: The Content ID for the content item.
- ❖ dID: The generated content item revision ID.
- ❖ dRevLabel: The content item revision label.
- ❖ dSecurityGroup: The security group such as *Public* or *Secure*.

- dDocAccount: The account for the content item. Required only if accounts are enabled.

Optional Service Parameters

- Metadata fields: You can submit any metadata field as an optional parameter and pass a value to that field.

Results

- Local Data:
  - Content Server:
    - dAction
    - dActionDate
    - dClbraName
    - dCreateDate
    - dDocAccount
    - dDocType
    - dDocName
    - dExtension
    - dID
    - dOriginalName
    - dOutDate
    - dpAction
    - dpEvent
    - dPublishState
    - dPublishType
    - dReleaseState
    - dRevClassID
    - dRevLabel
    - dSecurityGroup
    - dStatus
    - dUser
    - isCurRevEmpty

- ❖ isDocProfileUsed
- ❖ isEditMode
- ❖ IsNotLatestRev
- ❖ IsUpdate
- ❖ IsUpdateMetaOnly
- ❖ prevReleaseState
- ❖ updateSideEffectServices

- Workflow:
  - ❖ dCurRevID
  - ❖ dWfComputed
  - ❖ dWfCurrentStepID
  - ❖ dWfDirectory
  - ❖ dWfDocState
  - ❖ dWfID
  - ❖ dWfName
  - ❖ dWfStatus
  - ❖ dWfStepDescription
  - ❖ dWfStepID
  - ❖ dWfStepIsAll
  - ❖ dWfStepName
  - ❖ dWfStepType
  - ❖ dWfStepWeight
  - ❖ dWfType
  - ❖ dWorkflowState
  - ❖ IsWorkflow
  - ❖ wfAction
  - ❖ wfCurrentStepPrefix
  - ❖ wfEditFinished
  - ❖ wfMessage
  - ❖ wfQueueActionState
  - ❖ wfStepCheckinType

- Plus any custom doc meta fields

- Response Template: null

Used By

- Applets: Repository Manger
- Other: SoapCustom:Wsdl:CheckIn:Services

Example

- IdcCommand command file format:

```
IdcService=UPDATE_DOCINFO
dID=66
dDocName=test_000048
dSecurityGroup=Secure
dRevLabel=2
```

- HDA format with additional and optional parameters (assigns a value to a metadata field):

```
@Properties LocalData
IdcService=UPDATE_DOCINFO
dID=66
dDocName=test_000048
dSecurityGroup=Secure
dRevLabel=2
dDocAccount=mainaccount
xComments=Preliminary
@end
```

See Also

– *UPDATE_DOCINFO_SUB (page 5-110)*

UPDATE_DOCINFO_BYFORM

Service that updates content information for a content item from a browser. This service executes the subservice UPDATE_DOCINFO_SUB. This subservice updates content item information.

The difference between this service and UPDATE_DOCINFO is that this service provides an option to redirect to a display template.

Access Level: Write (2)

Calls Subservice: UPDATE_DOCINFO_SUB, which may also call REMOVE_METAFILE_SUB, REPLACE_METAFILE_SUB, and UPDATE_DOCINFO_STATUS

Queries Executed (all queries executed within subservices):

- QdocInfo
- QlatestID
- Qdocuments
- Ddocument
- Idocument
- Urevision2
- Umeta
- IdocHistory

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
- dID: The generated content item revision ID.
- dRevLabel: The content item revision label.
- dSecurityGroup: The security group such as *Public* or *Secure*.
- dDocAccount: The account for the content item. Required only if accounts are enabled.

Optional Service Parameters

- Metadata fields: You can submit any metadata field as an optional parameter and pass a value to that field. For example, if the system has a metadata field called *xComments*, the string value *Preliminary* could be assigned to that field (`xComments=Preliminary`).
- RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.

Results

- Local Data:

- Content Server:
  - dAction
  - dActionDate
  - dClbraName
  - dCreateDate
  - dDocAccount
  - dDocName
  - dDocType
  - dExtension
  - dID
  - dOriginalName
  - dOutDate
  - dpAction
  - dpEvent
  - dPublishState
  - dPublishType
  - dReleaseState
  - dRevClassID
  - dRevLabel
  - dSecurityGroup
  - dStatus
  - dUser
  - isCurRevEmpty
  - isDocProfileUsed
  - isEditMode
  - IsNotLatestRev
  - IsUpdate
  - IsUpdateMetaOnly
  - prevReleaseState
  - updateSideEffectServices
  -

- Workflow:
  - ❖ dCurRevID
  - ❖ dWfComputed
  - ❖ dWfCurrentStepID
  - ❖ dWfDirectory
  - ❖ dWfDocState
  - ❖ dWfID
  - ❖ dWfName
  - ❖ dWfStatus
  - ❖ dWfStepDescription
  - ❖ dWfStepID
  - ❖ dWfStepIsAll
  - ❖ dWfStepName
  - ❖ dWfStepType
  - ❖ dWfStepWeight
  - ❖ dWfType
  - ❖ dWorkflowState
  - ❖ IsWorkflow
  - ❖ wfAction
  - ❖ wfCurrentStepPrefix
  - ❖ wfEditFinished
  - ❖ wfMessage
  - ❖ wfQueueActionState
  - ❖ wfStepCheckinType
  - ❖ Plus any custom doc meta fields

❖ Response Template:

- REDIRECT_TEMPLATE (redirect_template.htm)
- Default redirect service: DOC_INFO

Used By

- Resource Includes: std_doc_page_definitions

Example

❖ IdcCommand command file format:

```
IdcService=UPDATE_DOCINFO_BYFORM
dID=66
dSecurityGroup=Secure
dDocName=test_000048
dRevLabel=2
```

❖ HDA format with additional and optional parameters (assigns a value to a metadata field):

```
@Properties LocalData
IdcService=UPDATE_DOCINFO_BYFORM
dID=66
dSecurityGroup=Secure
dDocName=test_000048
dRevLabel=2
dDocAccount=mainaccount
xComments=Preliminary
@end
```

See Also

– *UPDATE_DOCINFO_SUB (page 5-110)*

UPDATE_DOCINFO_BYREV

Location: <*install_dir*>/shared/config/resources/std_services.htm

UPDATE_DOCINFO_METAFILE_BYREV

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *REMOVE_METAFILE_SUB (page 5-81)*

UPDATE_DOCINFO_STATUS

Location: <*install_dir*>/shared/config/resources/std_services.htm

UPDATE_DOCINFO_SUB

Subservice that updates content item information.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *UPDATE_DOCINFO_BYFORM (page 5-105)*

– *REMOVE_METAFILE_SUB (page 5-81)*

UPDATE_SUBSCRIPTION_NOTIFY

Currently unused in the core Content Server.

Service that updates the notification timestamp and redirects the user to the page specified in the RedirectUrl parameter.

Location: <*install_dir*>/shared/config/resources/std_services.htm

UPDATE_SUBSCRIPTION_TYPE

Service that updates a Criteria subscription. The most likely error is when the specified subscription type does not exist.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ scpType: The subscription type.

Optional Service Parameters

❖ scpDescription: The subscription description.

❖ scpEnabled: 1 (*true*)—Subscription notifications are enabled.

0 (*false*)—Subscription notifications are disabled.

❖ scpFields: A comma-separated list of the metadata fields that define the subscription criteria. For example, *dDocAuthor*,*dDocType*.

Example

❖ IdcCommand command file format:

```
IdcService=UPDATE_SUBSCRIPTION_TYPE
scpType=my_subscription
```

❖ HDA file format (changes the subscription for Author and disables the subscription):

```
@Properties LocalData
IdcService=UPDATE_SUBSCRIPTION_TYPE
scpType=my_subscription
scpFields=dDocAuthor
scpDescription=updating subscription
scpEnabled=false
@end
```

UPDATE_SUBSCRIPTION_USED

Service that updates a user's subscription records when they access a content item from a subscription notification. This service updates the date in the *dSubscriptionUsedDate* column of the *Subscription* database table.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dDocName: The Content ID for the content item.

❖ dSubscriptionAlias: The user alias for the criteria subscription.

❖ dSubscriptionAliasType: The alias subscription type.

❖ dSubscriptionID: The subscription ID.

- For a Basic subscription, this is the Content ID.
- For a Criteria subscription, this is a comma-separated list of the values of the criteria fields.

For example, if the criteria fields are Author and Security Group, the *dSubscriptionID* for a *Public* content item authored by *user1* would be *Public,user1*.

❖ dSubscriptionType: The subscription type.

Example

```
IdcService=UPDATE_SUBSCRIPTION_USED
dSubscriptionType=test_subscription
```

```
dSubscriptionID=sysadmin
dDocName=test_000056
dSubscriptionAlias=global
dSubscriptionAliasType=user
```

UPDATE_METADATA

Subservice that updates metadata.

Location: <*install_dir*>/shared/config/resources/std_services.htm

VALIDATE_DOCINFO

Service used for metadata-only checkins for a heavy client (InternalUpload = 1).

Location: <*install_dir*>/shared/config/resources/std_services.htm

WORK_IN_PROGRESS

Service that returns a list of all content items in the GENWWW or DONE status. It does not show items currently in a workflow.

Access Level: Write, Global, Scriptable (50)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Optional Service Parameters

- orderClause: Use to provide a field name on which to sort the list.
- MaxQueryRows: Use to control the number of items returned in the list.

Results

- Result Sets: DOC_LIST (All Fields from Revisions and DocMeta)
- Local Data:
  - copyAborted
  - dataSource
  - MaxQueryRows
- Response Template: WIPS (wips_list.htm)

Used By

- Resource Includes:
  - pne_nav_management_links
  - std_doc_man_pages
- Standard Navigation:
  - commonBundle.js
  - commonNav.js

DOC PROFILE SERVICES (CORE CONTENT SERVER)

The following services are used when creating, maintaining, or deleting document profiles:

- ADD_DOCPROFILE (page 5-113)
- ADD_DOCRULE (page 5-114)
- DELETE_DOCPROFILE (page 5-114)
- DELETE_DOCRULE (page 5-115)
- DOCPROFILE_PREVIEW (page 5-115)
- EDIT_DOCPROFILE (page 5-115)
- EDIT_DOCPROFILE_TRIGGER (page 5-116)
- EDIT_DOCRULE (page 5-116)
- GET_DOCPROFILE (page 5-116)
- GET_DOCPROFILES (page 5-117)
- GET_DOCRULE (page 5-117)
- GET_DOCRULES (page 5-117)

ADD_DOCPROFILE

Service that is used to add a document profile. The service adds the profile and description to the content profile listing and generates the profile definition file from the data which is passed in.

Note: The data provided for this service becomes the profile definition. When editing the profile (using EDIT_DOCPROFILE), the profile definition must first be read using GET_DOCPROFILE before adding or changing the desired values.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dpName: The name for the profile. This must be a unique name.
- dpDescription: A description of the new profile.
- dpTriggerValue: The trigger value associated with the profile.
- dpDisplayLabel: The display label for the profile.

Optional Service Parameters

- isValidateTrigger: When set to *true*, validates that the trigger value exists. If the value does not exist, the service fails. When set to *false*, the profile is added without validating the trigger.

ADD_DOCRULE

Service used to create a new content profile rule. This service adds the rule to the content rule listing and creates the rule definition file. The data provided to the service becomes the rule definition.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dpRuleName: The name for the new rule. This name must be unique.
- dpRuleDescription: A description for the rule.

DELETE_DOCPROFILE

Service used to remove a document profile from the profile listing and removes the content profile definition file.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dpName: The name for the profile.

DELETE_DOCRULE

Service used to remove a content rule from the rule listing and deletes the content rule definition file from the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dpRuleName: The name for the rule.

DOCPROFILE_PREVIEW

Service used in the Configuration Manager applet to simulate the use of a content profile. The preview executes in the desired context and returns the resulting data in a data binder. The context consists of the event (submit, request, or import) and the action (search, checkin, and so on). It also includes a user name and a content item when necessary. The data includes display information for each field as well as which rule determined the display for each field.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dpName: The name for the rule to be used in the preview.
- dpTriggerValue: the trigger value to use for the preview.

EDIT_DOCPROFILE

Service used to edit a document profile.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dpName: The name for the profile to be edited.
- dpDescription: The new description for the profile.

- dpTriggerValue: The new trigger value for the profile.
- dpDisplayLabel: The new display label.

See Also

– *ADD_DOCPROFILE (page 5-113)*

EDIT_DOCPROFILE_TRIGGER

Service that changes the metadata field that will be used as the trigger field for the content profile.

Note: Changing the metadata fields may invalidate all existing profiles.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dpName: The name of the profile to be edited.
- dpTriggerField: The new trigger field to be used for the profile. Changing the trigger to empty (`dpTriggerfield=`) invalidates all profiles.

EDIT_DOCRULE

Service that changes a document rule description.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dpRuleName: The name of the rule to be edited.
- dpRuleDescription: The new description for the rule.

GET_DOCPROFILE

Service that returns the profile definition file for the specified rule. The returned data is in a data binder and contains all information used to define the profile.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dpName: The name of the profile to be used.

GET_DOCPROFILES

Service that returns the listing file for the content profiles. The listing file contains the name, description, trigger value and display label for each profile.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_DOCRULE

Service that returns the rule definition file for the specified rule. The returned data is returned in a data binder and contains all the information used to define the rule.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dpRuleName: The name of the profile to be used.

GET_DOCRULES

Service that returns the listing file for the content rules. The listing file contains the name and description for each rule in the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

FILE SERVICES (CORE CONTENT SERVER)

File services are those which manipulate files, such as dynamic conversion and resource files. Frequently used services are marked with an asterisk (*) in the following list.

The following file services are described in this section:

❖ ADD_WEB_APP (page 5-118)

❖ APPEND_FILE_CACHING_INFO (page 5-118)

❖ GET_DYNAMIC_CONVERSION (page 5-118)

❖ GET_DYNAMIC_CONVERSION_SUB (page 5-120)

- *GET_DYNAMIC_URL (page 5-121)
- GET_EXTERNAL_DYNAMIC_CONVERSION (page 5-122)
- GET_TEMPLATE_CONVERSIONS (page 5-122)
- GET_WEB_APP_STATUS (page 5-122)
- LOAD_RESOURCE_FILE (page 5-122)
- REMOVE_WEB_APP (page 5-124)
- SAVE_TEMPLATE_CONVERSIONS (page 5-124)

ADD_WEB_APP

Used with the internal Tomcat engine.

Service used to add a WAR file to the Tomcat engine and to enable the contained JSP pages.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- pathToWAR: Complete path to the WAR file.

APPEND_FILE_CACHING_INFO

Subservice used by GET_SYSTEM_AUDIT_INFO to return information on the System Audit Info page.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *GET_SYSTEM_AUDIT_INFO (page 5-13)*

GET_DYNAMIC_CONVERSION

Service that returns a content item as an HTML or XML file converted by Dynamic Converter. This is available if the Dynamic Converter component is installed and enabled.

Given a *dID* or a *dDocName* and a *RevisionSelectionMethod* parameter, the service determines the filename of a particular rendition of the revision and returns that file to the

client. The most likely errors are mismatched parameters or a request for a content item that does not exist.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Note: *dDocName* should be included in all requests for content items where the requester knows the *dDocName*. Error messages in the content server assume that it is present, as do other features such as forms.

Additional Required Service Parameters

Important: Either the content item revision ID (*dID*) must be specified or a Content ID (*dDocName*) with the *RevisionSelectionMethod* parameter.

- dID: The generated content item revision ID.
  - If *dID* is not specified, *dDocName* and *RevisionSelectionMethod* must specified.
  - A rendition of the revision of the content item with this ID is returned, if it exists, and the *RevisionSelectionMethod* parameter does not exist or has the value *Specific*.
- dDocName: The Content ID of the content item.
  - If *dDocName* is not present, *dID* must be present and *RevisionSelectionMethod* must not be present.
  - If *RevisionSelectionMethod* is present, a rendition of a revision of the content item with this name is returned, if it exists. If *RevisionSelectionMethod* is not present, *dDocName* is used in error messages.
- RevisionSelectionMethod: The revision selection method.
  - If present, *dDocName* must be present. The value of this variable is the method used to compute a *dID* from the specified *dDocName*. Its value can be *Specific*, *Latest*, or *LatestReleased*.
  - If the value is *Specific*, the *dDocName* is ignored, and *dID* is required and is used to get a revision. If the value is *Latest*, the latest revision of the content item is used to compute the *dID*. If the value is *LatestReleased*, the latest released revision of the content item is used to compute the *dID*.

Optional Service Parameters

- conversionTemplate: The conversion template. This parameter can be passed to the service to override the template conversions criteria page.

- DCViewFormat: The file to be converted and displayed, *Native, Alternate,* or *WebViewable*.
- useDocInfoCache: Disables the caching mechanism while requesting an individual content item. It applies only to the single request. You might use this parameter to test or troubleshoot the optimized query feature. To use this parameter, append it to your Content Server URL as follows (parameter is in bold):

  ```
  http://<server name>/intradoc-cgi/idc_cgi_isapi.dll?IdcService=
  GET_DYNAMIC_CONVERSION&dDocName=aTest&RevisionSelectionMethod=
  LatestReleased&useDocInfoCache=false
  ```

- conversionRule: Specifies the conversion rule to be used, rather than allowing the conversion rule to be selected by the conversion rule criteria.

Example

- IdcCommand command file format:

  ```
  # Retrieve a web-viewable revision
  IdcService=GET_DYNAMIC_CONVERSION
  dDocName=corporatereport
  RevisionSelectionMethod=LatestReleased
  DCViewFormat=WebViewable
  ```

- HDA format:

  ```
  @Properties LocalData
  IdcService=GET_DYNAMIC_CONVERSION
  dID=54321
  @end
  ```

See Also

– *GET_DYNAMIC_CONVERSION_SUB (page 5-120)*

– *GET_EXTERNAL_DYNAMIC_CONVERSION (page 5-122)*

GET_DYNAMIC_CONVERSION_SUB

Subservice used by GET_DYNAMIC_CONVERSION.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *GET_DYNAMIC_CONVERSION (page 5-118)*

GET_DYNAMIC_URL

Service that is used internally to extract information from the web-viewable URL to determine if it maps into the /weblayout directory. If it does not map then it throws an exception.

This service is usually used to retrieve a dynamic page. This service is called from the web server to deliver dynamic content web-viewable files. It is called in the core to dynamically change the response page for a service.

Access Level: Read, Scriptable (33)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- fileUrl: The relative URL for the item. This can be a checked-in item in the Content Service, a Java Script file for Schema or Layout Manager, or an image file from the /images directory.

Results

- Result Sets: DocFormats, DocTypes
- Local Data:
  - ref:dDocAccount
  - ref:dDocName
  - ref:dDocType
  - ref:dExtension
  - ref:dSecurityGroup
  - ref:hasDocInfo
  - ref:isLatestRevision
  - SourceID
  - HCSP/F metadata (including custom XML data between idcbegindata and idcenddata tags)
- Response Template: null (Returns the file requested in the fileUrl parameter.)

GET_EXTERNAL_DYNAMIC_CONVERSION

Service used to dynamically convert a document residing in an external collection. It is equivalent to GET_DYNAMIC_CONVERSION with the exception that it retrieves the content information for the item from the external Verity search collection. Verity is an add-on product available for purchase for use with Content Server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- VdkVgwKey: The look-up key into Verity.
- sCollectionID: The identifier for the external Verity collection.

See Also

– *GET_DYNAMIC_CONVERSION (page 5-118)*

GET_TEMPLATE_CONVERSIONS

Service used by the Dynamic Converter to retrieve the list of templates that are used during conversion.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_WEB_APP_STATUS

Used with the internal Tomcat engine.

Service that returns the status of the internal Tomcat engine. For internal use only.

Location: <*install_dir*>/shared/config/resources/std_services.htm

LOAD_RESOURCE_FILE

Service that returns a resource file.

Given a *dID* or a *dDocName* and a *RevisionSelectionMethod* parameter, the service determines the filename of a particular rendition of the revision and returns that file to the client.

The most likely errors are some form of mismatched parameters or a request for a revision or rendition that does not exist.

Note: It is recommended that *dDocName* be included in all requests for content items where the requester knows the *dDocName*. Error messages in the content server assume that it is present, as do other features such as forms.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

Important: Either the content item revision ID (*dID*) must be specified or a content item name (*dDocName*) and a *RevisionSelectionMethod* parameter must be specified.

Optional Service Parameters

- dID: The generated content item revision ID.
  - If *dID* is not specified, *dDocName* and *RevisionSelectionMethod* must specified.
  - A rendition of the revision of the content item with this ID is returned, if it exists, and the *RevisionSelectionMethod* parameter does not exist or has the value *Specific*.
- dDocName: The Content ID for the content item.
  - If *dDocName* is not present, *dID* must be present and *RevisionSelectionMethod* must not be present.
  - If *RevisionSelectionMethod* is present, a rendition of a revision of the content item with this name is returned, if it exists. If *RevisionSelectionMethod* is not present, *dDocName* is used in error messages.
- RevisionSelectionMethod: The revision selection method.
  - If present, *dDocName* must be present. The value of this variable is the method used to compute a *dID* from the specified *dDocName*. Its value can be *Specific*, *Latest*, or *LatestReleased*.
  - If the value is *Specific*, the *dDocName* is ignored, and *dID* is required and is used to get a rendition. If the value is *Latest*, the latest revision of the content item is used to compute the *dID*. If the value is *LatestReleased*, the latest released revision of the content item is used to compute the *dID*.
- Rendition: The content item rendition. This parameter specifies the rendition of the content item and can be set to *Primary*, *Web*, or *Alternate*. If *Rendition* is not present, it defaults to *Primary*.
  - If the value is *Primary*, the primary rendition of the selected revision is returned.

- If the value is *Web*, the web-viewable rendition of the selected revision is returned.
- If the value is *Alternate*, the alternate rendition of the selected revision is returned.

Example

```
IdcService=LOAD_RESOURCE_FILE
dID=456
```

REMOVE_WEB_APP

Used with the internal Tomcat engine.

This service is used to remove a WAR file from use with the internal Tomcat engine. For internal use only.

Location: <*install_dir*>/shared/config/resources/std_services.htm

SAVE_TEMPLATE_CONVERSIONS

Service used by the Dynamic Converter to save the conversion template.

Location: <*install_dir*>/shared/config/resources/std_services.htm

INDEXER SERVICES (CORE CONTENT SERVER)

Indexer services are used to control the search index. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- ❖ CANCEL_SEARCH_INDEX (page 5-124)
- ❖ CONTROL_SEARCH_INDEX (page 5-125)
- ❖ *GET_FILE (page 5-129)
- ❖ START_SEARCH_INDEX (page 5-132)

CANCEL_SEARCH_INDEX

Service that cancels the current indexing session.

Location: <*install_dir*>/shared/config/resources/std_services.htm

CONTROL_SEARCH_INDEX

Service that updates or rebuilds the search index. Updates and rebuilds are performed automatically in a background thread.

Location: <*install_dir*>*/shared/config/resources/std_services.htm*

Important: Rebuilding the search index is necessary only when you change or add metadata fields. Depending on the quantity and size of your files, this process can take several days. Rebuilding is system-intensive, so you should plan to rebuild during non-peak system usage times.

- This service is called when changes are made to the Automatic Update Cycle and Collection Rebuild Cycle options provided on the Repository Manager: Indexer Tab.
- The options defined on the Configure Automatic Update Cycle or Configure Collection Rebuild Cycle screen map to these parameters:
  - Content Items Per Indexer Batch (MaxCollectionSize)
  - Content Items Per Checkpoint (IndexerCheckpointCount)
  - Indexer Debug Level (SearchDebugLevel)
  - Indexer Auto Updates (sEnableAutoUpdate)

Additional Required Service Parameters

- cycleID: The index cycle type:
  - **update—**Incrementally updates the index database.
  - **rebuild—**The search index is entirely rebuilt, and the old index collection is replaced with a new index collection once the rebuild has successfully completed.
- action: The action to perform. This entry can be set to these values:
  - **start**—Begins the update or rebuild cycle.
  - **cancel**—Stops the update or rebuild cycle, and does not permit a future restart.
  - **suspend**—Stops the update or rebuild cycle, and permits a future restart.
  - **restart**—Restarts the update or rebuild cycle after a suspend was executed.
  - **setConfiguration**—Used when enabling or disabling the update cycle. Setting this value enables you to adjust the checkpoint (*IndexerCheckpointCount*), batch

size (*MaxCollectionSize*), debug level (*SearchDebugLevel*), and auto update (*sEnableAutoUpdate*) options.

Additional Optional Service Parameters

This service might require these parameters, depending on the action to execute (see examples below):

GetCurrentIndexingStatus: Setting this parameter to 1 (*true*) returns the status of the indexer cycle:

- **Idle**—An indexer cycle is complete.
- **Active**—An indexer cycle is currently running.
- **Interrupt**—An unexpected event, that abruptly ends the indexing cycle. For example, a power, database, or file system failure.
- **Suspend**—The indexing cycle was stopped in a controlled manner. For example, using the Suspend button on the Repository Manager: Indexer Tab.
- **Restart**—An interrupted or suspended cycle was started again. If you are restarting after an interrupted cycle, ensure that you have corrected the problem that caused the interrupt to occur.
- **Cancel**—The indexing cycle was stopped with no intent to restart.

❖ getStatus: Setting this parameter to 1 (*true*) returns the indexer settings and status information.

❖ PerformProcessConversion: Enables the process conversion when starting or restarting the Indexer rebuild process.

❖ 1 (*true*)—The process conversion is performed. 0 (*false*)—The process conversion is not performed.

❖ IndexerCheckpointCount: The number of files that go through each indexing state at a time.

- You can have multiple batches of files indexed per checkpoint.
- To update this entry, the action must be set to *setConfiguration*.

❖ MaxCollectionSize: The maximum number of files that the search index processes at one time.

- The default is 25. This means 25 files are indexed together, then the next 25 files are indexed.

- You can change this setting to 1 if you are experiencing problems with the search engine indexing large and complicated files. However, slow system performance can result.
- To update this entry, the action must be set to *setConfiguration*.

❖ SearchDebugLevel: The indexer debug level.

- The more debug information listed in the server window, the slower the indexing progresses.
- To update this entry, the action must be set to *setConfiguration*.

These are the debug levels from the least to the most debug information:

- **none**—No information for each file accessed is displayed.
- **verbose**—Displays information for each file accessed. Indicates indexed, ignored, or failed.
- **debug**—Displays the medium level of information.
- **trace**—Displays the lowest level of information.
- **all**—Displays the highest level of information.

❖ sEnableAutoUpdate: To update this entry, the action must be set to *setConfiguration*.

- 1 (*true*)—Automatic update cycles are enabled. 0 (*false*)—Automatic update cycles are enabled.

Example

These examples are for controlling the indexer from another process. This is useful because the indexer should not be running when the Verity database is being optimized and in some configurations it is important to optimize the Verity collection on a regularly scheduled basis. Verity is an add-on product available for purchase for use with Content Server.

❖ HDA format to return the defined settings and status information:

```
@Properties LocalData
IdcService=CONTROL_SEARCH_INDEX
getStatus=1
@end
```

❖ HDA format to enable the update cycle:

```
@Properties LocalData
IdcService=CONTROL_SEARCH_INDEX
```

```
MaxCollectionSize=500
sEnableAutoUpdate=1
cycleID=update
action=setConfiguration
SearchDebugLevel=none
IndexerCheckpointCount=5000
GetCurrentIndexingStatus=1
@end
```

- HDA format to disable the update cycle:

```
@Properties LocalData
IdcService=CONTROL_SEARCH_INDEX
MaxCollectionSize=500
sEnableAutoUpdate=0
cycleID=update
action=setConfiguration
SearchDebugLevel=none
IndexerCheckpointCount=5000
GetCurrentIndexingStatus=1
@end
```

- HDA format to start the Indexer rebuild:

```
@Properties LocalData
IdcService=CONTROL_SEARCH_INDEX
cycleID=rebuild
action=start
getStatus=1
GetCurrentIndexingStatus=1
PerformProcessConversion=1
@end
```

- HDA format to cancel the Indexer rebuild:

```
@Properties LocalData
IdcService=CONTROL_SEARCH_INDEX
cycleID=rebuild
action=cancel
getStatus=1
GetCurrentIndexingStatus=1
@end
```

- HDA format to suspend the Indexer rebuild:

```
@Properties LocalData
IdcService=CONTROL_SEARCH_INDEX
cycleID=rebuild
action=suspend
getStatus=1
GetCurrentIndexingStatus=1
@end
```

- HDA format to restart the Indexer rebuild after being interrupted or suspended:

```
@Properties LocalData
IdcService=CONTROL_SEARCH_INDEX
cycleID=rebuild
action=restart
getStatus=1
GetCurrentIndexingStatus=1
PerformProcessConversion=1
@end
```

See Also

– *START_SEARCH_INDEX (page 5-132)*

GET_FILE

Service that returns a specific rendition of a content item revision to a browser. A copy of the file is retrieved without performing a checkout.

- Given a *dID* or a *dDocName* and a *RevisionSelectionMethod* parameter, the service determines the file name of a particular rendition of the revision and returns that file to the client.
- The most likely errors are some form of mismatched parameters or a request for a revision or rendition that does not exist.

Note: It is recommended that *dDocName* be included in all requests for content items where the requester knows the *dDocName*. Error messages in the content server assume that it is present, as do other features such as forms.

Access Level: Read, Scriptable (33)

Queries Executed:

- QdocInfo
- QlatestIdByName
- QlatestReleasedIDByName

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

Important: Either the content item revision ID (*dID*) must be specified or a Content ID (*dDocName*) along with a *RevisionSelectionMethod* parameter must be defined.

- dID: The generated content item revision ID.
  - If *dID* is not specified, *dDocName* and *RevisionSelectionMethod* must specified.
  - A rendition of the revision of the content item with this ID is returned, if it exists, and the *RevisionSelectionMethod* parameter does not exist or has the value *Specific*.
- dDocName: The Content ID for the content item.
  - If *dDocName* is not present, dID must be present and *RevisionSelectionMethod* must not be present.
  - If *RevisionSelectionMethod* is present, a rendition of a revision of the content item with this name is returned, if it exists. If *RevisionSelectionMethod* is not present, *dDocName* is used in error messages.

Optional Service Parameters

- allowInterrupt: If set to 1 (*true*), this suppresses an error if the user cancels the file download.
- RevisionSelectionMethod: The revision selection method.
  - If present, *dDocName* must be present. The value of this variable is the method used to compute a *dID* from the specified *dDocName*. Its value can be *Specific*, *Latest*, or *LatestReleased*.
  - If the value is *Specific*, the *dDocName* is ignored, and *dID* is required and is used to get a rendition. If the value is *Latest*, the latest revision of the content item is used to compute the *dID*. If the value is *LatestReleased*, the latest released revision of the content item is used to compute the *dID*.

- Rendition: The content item rendition. This parameter specifies the rendition of the content item and can be set to *Primary*, *Web*, or *Alternate*. If *Rendition* is not present, it defaults to *Primary.*
  - If the value is *Primary*, the primary rendition of the selected revision is returned.
  - If the value is *Web*, the web-viewable rendition of the selected revision is returned.
  - If the value is *Alternate*, the alternate rendition of the selected revision is returned.

  Other rendition types may be available depending on the add-on products used with your instance (for example, *thumbnail* if the Inbound Refinery product is used).
- IsXml: When set to *true* or 1, returns the XML data island which is present in some HCSP, HCST, and HCSF pages. It returns the data island that is wrappered inside the std_html_form_xml_wrapper include, which can be modified to pass additional information if desired.

Results

- Response Template: null (Only the requested file is returned.)

Used By

- Applets: Configuration Manager
- Resource Includes:
  - doc_file_get_copy
  - doc_odma_select_get
  - download_form_fields
  - email_docinfo_body_by_id
  - email_docinfo_body_by_name
  - legacy_workflow_in_queue_table
  - subscription_info_cell
  - subscription_info_cells
  - wf_in_queue_display
  - wf_review_cannot_view_msg
- Templates:
  - CHECKOUT_OK (chkook.htm)

- DOWNLOAD_OK (dwnldok.htm)
- QUERY_NOTIFICATION (query_notification_mail.htm)
- REDIRECTION_FILE_TEMPLATE (redirectionfile_template.htm)
- REV_HISTORY (rev_history.htm)
- SELECTDOC_OK (slctdcok.htm)
- WORKFLOW_REVIEW_FORM (workflow_review_form.htm)
- WORKFLOW_REVIEW_FRAMES (workflow_review_frames.htm)

❖ Other: SoapCustom:Wsdl:GetFile:Services

Example

❖ IdcCommand command file format:

```
# Retrieve a web-viewable revision
IdcService=GET_FILE
dDocName=corporatereport
RevisionSelectionMethod=LatestReleased
Rendition=Web
```

❖ HDA format:

```
@Properties LocalData
IdcService=GET_FILE
dID=54321
@end
```

See Also

– *GET_ARCHIVED_FILE (page 7-17)*

START_SEARCH_INDEX

Service that updates or rebuilds the search index. This service is asynchronous and the action is performed in a background thread. This service can only be executed in the context of the content server and only completes successfully if a connection is made to the content server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Optional Service Parameters

- IsRebuild: Sets the scope of the indexing process.
  - 1 (*true*)—The Indexer performs a complete rebuild of the search index.
    0 (*false*)—The Indexer performs an incremental update of the search index.
  - Default is 0 (false).

Example

- IdcCommand command file format:

```
# Rebuild the entire search index
IdcService=START_SEARCH_INDEX
IsRebuild=true
```

- HDA format (rebuild the search index).

```
@Properties LocalData
IdcService=START_SEARCH_INDEX
IsRebuild=true
@end
```

INTERNAL SERVICES (CORE CONTENT SERVER)

The following services are called internally by the content server. **Do not use or modify these services**:

- CANCEL_COMPONENT_INSTALL (page 5-134)
- CLEAR_SERVER_OUTPUT (page 5-134)
- DOWNLOAD_COMPONENT (page 5-134)
- GET_COMPONENT_CONFIG (page 5-134)
- GET_COMPONENT_INSTALL_FORM (page 5-134)
- GET_COMPONENT_INSTALL_PROMPTS_FORM (page 5-134)
- GET_COMPONENT_INSTALL_SETTINGS' (page 5-134)
- GET_LOCAL_REGISTRATION_FORM (page 5-135)
- GET_MANIFEST_INFO (page 5-135)
- GET_SERVER_OUTPUT (page 5-135)

- PROXIED_REQUEST (page 5-135)
- UNINSTALL_COMPONENT (page 5-135)
- UPDATE_COMPONENT_CONFIG (page 5-135)
- UPDATE_LICENSE (page 5-135)
- UPLOAD_NEW_COMPONENT (page 5-135)

CANCEL_COMPONENT_INSTALL

Do not use.

CLEAR_SERVER_OUTPUT

Do not use.

DOWNLOAD_COMPONENT

Do not use.

GET_COMPONENT_CONFIG

Do not use.

GET_COMPONENT_INSTALL_FORM

Do not use.

GET_COMPONENT_INSTALL_PROMPTS_FORM

Do not use.

GET_COMPONENT_INSTALL_SETTINGS’

Do not use.

GET_LOCAL_REGISTRATION_FORM

Do not use.

GET_MANIFEST_INFO

Do not use.

GET_SERVER_OUTPUT

Do not use.

PROXIED_REQUEST

Do not use.

UNINSTALL_COMPONENT

Do not use.

UPDATE_COMPONENT_CONFIG

Do not use.

UPDATE_LICENSE

Do not use.

UPLOAD_NEW_COMPONENT

Do not use.

META SERVICES (CORE CONTENT SERVER)

The Meta Services are used to manage and alter metadata in the Content Server. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- ADD_METADEF (page 5-136)
- DEL_METADEF (page 5-138)
- EDIT_METADEF (page 5-138)
- GET_ADVANCED_SEARCH_OPTIONS (page 5-139)
- *GET_DOC_METADATA_INFO (page 5-139)
- GET_OPTION_LIST (page 5-140)
- GET_USER_METADATA_INFO (page 5-140)
- MOVE_METADEF (page 5-140)
- UPDATE_ADVANCED_SEARCH_OPTIONS (page 5-141)
- UPDATE_USER_META (page 5-141)
- UPDATE_USER_META_TABLE (page 5-141)
- UPDATE_META_TABLE (page 5-142)
- UPDATE_OPTION_LIST (page 5-142)

ADD_METADEF

Service that creates a new metadata field.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Note: See UPDATE_META_TABLE and START_SEARCH_INDEX for more details. These services must be run before storing data in a new metadata field. See also UPDATE_OPTION_LIST for details about updating the *OptionLists* database table.

Additional Required Service Parameters

- dName: Internal name of the field.
- dCaption: User-visible caption used to label the field in the content server pages.
- dIsRequired: Prevents files from being checked in if the field does not contain a value. Values are 1 (required) or 0 (not required). Default is 0.
- dIsEnabled: Enables the field to be displayed on user interface pages. Values are **1** (enabled) or **0** (disabled). Default is 1.

- dIsSearchable: Enables the field to be indexed and searchable. Values are **1** (searchable) or **0** (not searchable). Default is 1.
- dIsOptionList: Allows the use of a user-selectable option list on content server pages. Values are **1** (enabled) or **0** (disabled). Default is 0. If 1 is used then dOptionListKey is also required.
- dOptionListKey: Name of the option list to be used.
- dOptionListType: Specifies the type of option list to be used. Values can be `choice` (Select List Validated), `chunval` (Select List Not Validated), `combo` (Edit and Select List), `multi2` (Multiselect List) or `multi` (Edit and Multiselect List).
- dType: The type of field. Values can be Text (Text), BigText (Long Text), Int (Integer), Date (Date), Memo (Memo). Default: Text.
- dOrder: Sequence in which the field is displayed on content server pages. The default value is the highest current value of any existing field, plus one.
- dDefaultValue: Default value for the metadata field being created.

Example

```
IdcService=ADD_METADEF
dIsRequired=0
dOptionListKey=Web_SectionList
dOptionListType=choice
dIsOptionList=1
dOrder=5
dName=xWeb_Section
dIsSearchable=1
dIsEnabled=1
dType=Text
FieldName=Web_Section
dCaption=Web Section
```

See Also

– *START_SEARCH_INDEX (page 5-132)*
– *UPDATE_META_TABLE (page 5-142)*
– *UPDATE_OPTION_LIST (page 5-142)*

DEL_METADEF

Service that deletes an existing custom metadata field. You cannot delete the standard metadata fields such as *dDocName*, *dSecurityGroup*, and so forth.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dName: The metadata field name, including the "x" prefix.

Example

```
IdcService=DEL_METADEF
dName=xCustomField
```

EDIT_METADEF

Service that modifies an existing custom metadata field.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dName: The metadata field name, including the "x" prefix. For example, *xDepartment*.
- dCaption: The metadata field caption.
- dType: The metadata field type, such as *Text*, *BigText*, *Memo*, *Date*, or *Integer*.
- dOrder: The display order.
- dIsRequired: 1 (*true*)—The field is required. 0 (*false*)—The field is optional.
- dIsEnabled: 1 (*true*)—The field is enabled on the user interface. 0 (*false*)—The field is disabled on the user interface.
- dIsSearchable: 1 (*true*)—The field is searchable. 0 (*false*)—The field is not searchable.
- dDefaultValue: The default value for the metadata field.
- dIsOptionList: 1 (*true*)—The field has an option list. 0 (*false*)—The field does not have an option list.
- dOptionListKey: The option list key.

❖ dOptionListType: The option list type.

Example

```
IdcService=EDIT_METADEF
dName=xDepartment
dType=BigText
dIsRequired=1
dIsEnabled=1
dIsSearchable=1
dCaption=Caption_Changed
dIsOptionList=1
dDefaultValue=two
dOptionListKey=xDepartmen_fieldList
dOptionListType=chunval
dOrder=4
```

GET_ADVANCED_SEARCH_OPTIONS

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_DOC_METADATA_INFO

Service that returns a list of custom metadata fields from the DocMetaDefinition table, and a listing of information for the available DocTypes as result sets. Also returns two option lists of available values for SecurityGroups and Accounts filtered to show only those the user can access. Used with SOAP retrievals and other remote applications.

Access Level: Read, Global, Scriptable (49)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results

❖ Result Sets:

- DocMetaDefinition (All rows and fields from DocMetaDefiniton database table.)
- Doctypes (All rows and fields from DocTypes database table.)

❖ Option Lists:

- SecurityGroups (Option list of security groups filtered based upon user's privileges/)
- Accounts (Option list of Accounts filtered based upon user's privileges.)

❖ Local Data:

- isAutoNumber
- useAccounts

Used By

❖ Other: SoapCustom:Wsdl:MetaData:Services

GET_OPTION_LIST

Service that returns a table of all option lists. Returns the common key value, option value, and order for each option list.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_USER_METADATA_INFO

Service that returns metadata information for users. Used with SOAP retrievals.

Location: <*install_dir*>/shared/config/resources/std_services.htm

MOVE_METADEF

Service used by the Configuration Manager to move a metadata field up or down in the list. Moving the metadata in the list changes the default order in which it is displayed on the Checkin, Update, Info, and Search pages.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dName: Name of the metadata field to move.

Optional Service Parameters

❖ isMoveUp: Default is *false.*Set to *true* to move the field up, not down.

UPDATE_ADVANCED_SEARCH_OPTIONS

Location: <*install_dir*>/shared/config/resources/std_services.htm

UPDATE_USER_META

Location: <*install_dir*>/shared/config/resources/std_services.htm

UPDATE_USER_META_TABLE

Service that updates the user information fields in the database. This service alters the database by adding, altering, and deleting columns.

- By default, fields will only be added or changed, not deleted. To delete a user information field, it must be specifically mentioned in the optional parameter *MetaFieldsToDelete*.
- This service is run when you add or delete user information fields in the User Admin applet and click the **Update Database Design** button.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Caution: Data may be lost if a column is deleted. This service should be used carefully.

Optional Service Parameters

- MetaFieldsToDelete: Specifies a comma-separated list of the database columns to delete.

Example

- IdcCommand command file format:

```
IdcService=UPDATE_USER_META_TABLE
```

- HDA format (deletes two fields from the database):

```
@Properties LocalData
IdcService=UPDATE_USER_META_TABLE
MetaFieldsToDelete=xUserLocation,xDivision
@end
```

UPDATE_META_TABLE

Service that updates the metadata fields in the database.

This service alters the database by adding, altering, and deleting columns in the *DocMeta* table to conform to the *DocMetaDefinition* table. By default, fields will only be added or changed, not deleted. To delete a metadata field, it must be specifically mentioned in the optional parameter *MetaFieldsToDelete*.

This service is run when you add or delete metadata fields in the Configuration Manager applet and click the **Update Database Design** button.

Caution: Data may be lost if a column is deleted. This service should be used carefully.

Optional Service Parameters

- MetaFieldsToDelete: Specifies a comma-separated list of the database columns to delete from the *DocMeta* table.

Example

- IdcCommand command file format:

```
# Synchronize the DocMeta table with the design in the DocMetaDefinition table
IdcService=UPDATE_META_TABLE
```

- HDA format (deletes two fields from the database):

```
@Properties LocalData
IdcService=UPDATE_META_TABLE
MetaFieldsToDelete=xInteger,xMemo
@end
```

UPDATE_OPTION_LIST

Service that adds or updates an option list for a metadata field. Updates or adds an option list in the *OptionLists* database table.

Caution: The option list values specified in the *OptionListString* parameter replace any existing values. To retain existing values, you must include the existing values along with any new values.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dKey: The option list key.
- OptionListString: The list of options, separated by the \n escape sequence.

Example

- IdcCommand command file format:

```
# Add the new options list 'LocationList' and
# Add values Madrid, Tokyo, London, Washington
IdcService=UPDATE_OPTION_LIST
dKey=LocationList
OptionListString=Madrid\nTokyo\nLondon\nWashington
```

- HDA format (adds the option list *LocationList* with the values *Madrid*, *Tokyo*, *London*, and *Washington*):

```
@Properties LocalData
IdcService=UPDATE_OPTION_LIST
dKey=LocationList
OptionListString=Madrid\nTokyo\nLondon\nWashington
@end
```

MISCELLANEOUS SERVICES (CORE CONTENT SERVER)

The services in this section are used for a variety of tasks, including working with batch loads, retrieving pages, and chunking files for uploading. The following services are described in this section:

- CHUNKED_UPLOAD (page 5-143)
- DOWNLOAD_LISTBOX_ITEMS (page 5-144)
- LOAD_USER_LOCALIZATION (page 5-144)

CHUNKED_UPLOAD

This service adds support for calling a Content Server service that is very large. It chunks the request.

Location: <*install_dir*>/shared/config/resources/std_services.htm

DOWNLOAD_LISTBOX_ITEMS

Service used by the SelectUser applet to provide type-ahead capabilities to the Users or Alias table. It returns ListBoxServiceItems, a resultset containing the results of the query.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dataSource: The name of the data source to use (for example, select query stub).

Optional Service Parameters

- limit: The number of items to download. The default is 100.
- op: The operator to use when creating the WHERE clause.

LOAD_USER_LOCALIZATION

Service that loads localization information for user.

Location: <*install_dir*>/shared/config/resources/std_services.htm

PAGE HANDLER/PAGE REQUEST SERVICES (CORE CONTENT SERVER)

Page Request Services retrieve HTML pages. Page Handler Services manage library web pages created by the Web Layout Editor. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- DELETE_RESULT_TEMPLATE (page 5-145)
- *GET_ADMIN_PAGE (page 5-145)
- *GET_DOC_PAGE (page 5-146)
- *GET_DYNAMIC_PAGE (page 5-149)
- GET_PORTAL_PAGE (page 5-150)
- *GET_SECURE_PAGE (page 5-151)
- LOAD_GLOBALINCLUDES (page 5-153)

- PAGE_HANDLER (page 5-153)
- PNE_SAVE_QUERY (page 5-154)
- PNE_UPDATE_PERSONAL_URLS (page 5-154)
- PNE_UPDATE_PORTAL_INFO (page 5-155)
- SAVE_GLOBALINCLUDES (page 5-155)
- UPDATE_RESULT_TEMPLATE (page 5-155)

DELETE_RESULT_TEMPLATE

Service that deletes an existing search results template.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- name: The search results template name.

Example

```
IdcService=DELETE_RESULT_TEMPLATE
name=test_template
```

GET_ADMIN_PAGE

Service that returns the Administration page in a browser. It does not load the server's configuration. If the user is not assigned the admin role or sub administrator rights, the service returns an error message.

Access Level: N/A (0)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- PageName: The name of the page template.

Optional Service Parameters

- Action: The action to execute. Usually set to *GetTemplatePage*.

Results

- Local Data:
  - TemplateType
  - TemplateClass
  - TemplateFilePath
- Response Template: Returns whatever template was passed as the value of the *Page* parameter. In standard usage this is ADMIN_LINKS (admin.htm).

Used By

- Resource Includes:
  - pne_nav_admin_links
  - std_admin_pages
- Templates:
  - ADMIN_LINKS (admin.htm)
  - IDC_ADMIN_PAGE (idc_admin_page.htm)
  - std_home_page.htm
- Standard Navigation: commonNav.js

Example

To get the standard Administration page:

```
IdcService=GET_ADMIN_PAGE
Page=ADMIN_LINKS
Action=GetTemplatePage
```

GET_DOC_PAGE

Service that executes an HTML page request. This service is usually called from the browser interface. It executes the LOAD_DOC_ENVIRONMENT subservice.

Access Level: Read, Global, Scriptable (49)

Calls Subservice: LOAD_DOC_ENVIRONMENT

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ PageName: The name of the page template.

Optional Service Parameters

❖ Action: The action to execute. Usually set to *GetTemplatePage*.

Results

❖ Result Sets:

- DocFormats (All rows and fields of the DocFormats database table.)
- DocTypes (All rows and fields of the DocTypes database table.)

❖ Local Data:

- TemplateClass
- TemplateFilePath
- TemplateType

❖ Response Template: Returns whatever template was passed as the value of the *Page* parameter.

Used By

❖ Resource Includes:

- calculate_doc_profile_urls
- determine_wf_review_rendition_url
- home_page_static_content
- pne_nav_bookmark_links
- pne_nav_shared_links
- search_template_user_info_settings
- std_js_bootstrap_vars
- std_main_page_begin
- std_query_page_link_args
- xui_searchapi_results_action_form

❖ Templates:

- PNE_PORTAL_DOC_PROFILES_PAGE (pne_portal_doc_profiles_page.htm)

- PNE_PORTAL_PERSONAL_URLS_PAGE (pne_portal_personal_urls_page.htm)
- PNE_PORTAL_SAVED_QUERIES_PAGE (pne_portal_saved_queries_page.htm)
- PNE_PORTAL_SYSTEM_LINKS_PAGE (pne_portal_system_links_page.htm)
- PREVIEW_FRAMES (preview_frames.htm)
- USER_INFO (user_info.htm)

❖ Standard Navigation:

- commonNav.js
- Trays/layout.js
- Trays/search_tray_tabs.htm

❖ Other:

- Redirect service for: SUBMIT_HTML_FORM, LOGIN, EDIT_USER_PROFILE, CONTINUE_SUBMIT_HTML_FORM, UPLOAD_NEW_COMPONENT, CANCEL_COMPONENT_INSTALL, UPDATE_COMPONENT_CONFIG, UNINSTALL_COMPONENT, SAVE_USER_TOPICS, PUBLISH_SCHEMA, EDIT_SCHEMA_VIEW_VALUES

Example

❖ To get the home page the parameters are:

```
IdcService=GET_DOC_PAGE
Page=HOME_PAGE
Action=GetTemplatePage
```

❖ To get the standard query page the parameters are:

```
IdcService=GET_DOC_PAGE
Page=STANDARD_QUERY_PAGE
Action=GetTemplatePage
```

See Also

– *LOAD_DOC_ENVIRONMENT (page 5-14)*

GET_DYNAMIC_PAGE

Service that returns a Library page for dynamic assembly. This service can be used in two ways:

- To load a library page for display directly in Content Server.
- Used in the background in a hidden frame to dynamically generate or update nodes in the Library tree display.

The default template used by the DIRECTORY_PAGE service (dir_page.htm) is branched based upon the value of lmXML to perform one or the other of the two uses for the service. If lmXML is set to 1 (*true*), the template generates an XML object instead of an HTML page, then fires the libraryNodeLoadSequence to update the Library navigation tree.

Access Level: Read, Scriptable (33)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- PageName: The name of the page template.

Results

- Result Sets:
  - PageMap (result set containing local pages contained within this library node; fields: PageName, PageParent)
  - LinkList (result set containing listing of links contained within this library node; fields: LinkType, LinkData, LinkTitle, LinkDescription)
- Local Data:
  - dSecurityGroup
  - HeaderText
  - LinkSelectedIndex
  - LocationInfo
  - OutOfDate
  - PageFunction
  - PageLastChanged
  - PageName

- PageParent
- PageTitle
- PageType
- PageUrl
- restrictByGroup
- TemplatePage

❖ Response Template: null (During execution of the service, the TemplatePage value is set to DIRECTORY_PAGE (dir_page.htm)).

Used By

❖ Applets: Installer

❖ Resource Includes:

- calculate_all_doc_profile_urls
- home_page_static_content
- pne_nav_shared_links
- std_main_page_begin
- std_page_nav_bar

❖ Standard Navigation: commonNav.js

Example

To get the index page the required parameters are:

```
IdcService=GET_DYNAMIC_PAGE
PageName=index
Action=GetTemplatePage
```

GET_PORTAL_PAGE

Service that displays a Portal Design page. The most likely errors are when there is an error loading configuration information, an error retrieving the options list, or when the referenced HTML page is not found.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- Action: The action to execute. Usually Ge*tTemplatePage*.
- PageName: The name of the page template. The following standard pages are displayed using this parameter:

  Portal Design page—`PNE_PORTAL_DESIGN_PAGE`

  System Links page—`PNE_PORTAL_SYSTEM_LINKS_PAGE`

  Saved Queries page—`PNE_PORTAL_SAVED_QUERIES_PAGE`

  Personal URLs page—`PNE_PORTAL_PERSONAL_URLS_PAGE`

Example

- To get the Portal Design page, the parameters are:

```
IdcService=GET_PORTAL_PAGE
Action=GetTemplatePage
Page=PNE_PORTAL_DESIGN_PAGE
```

- To get the Saved Queries page, the parameters are:

```
IdcService=GET_PORTAL_PAGE
Action=GetTemplatePage
Page=PNE_PORTAL_SAVED_QUERIES_PAGE
```

GET_SECURE_PAGE

Service that executes a secure HTML page request. This service is usually called from the browser interface and is restricted to users with Write permission to at least one group. The most likely error is when the referenced HTM page is not found.

This service is identical to GET_DOC_PAGE except this service requires Write privileges to at least one security group, whereas GET_DOC_PAGE only requires Read access.

This service executes the LOAD_DOC_ENVIRONMENT subservice.

Access Level: Write, Global, Scriptable (50)

Calls Subservice: LOAD_DOC_ENVIRONMENT

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- Page: The name of the page template.

Optional Service Parameters

- Action: The action to execute. Usually GetTemplate*Page*.

Results

- Result Sets:
  - DocFormats (all rows and fields of the DocFormats database table)
  - DocTypes (all rows and fields of the DocTypes database table)
- Local Data:
  - TemplateClass
  - TemplateFilePath
  - TemplateType
- Response Template: Returns whatever template was passed as the value of the Page parameter

Used By

- Resource Includes: pne_nav_management_links
- Templates: std_home_page.htm

Example

To get the content management page the parameters are:

```
IdcService=GET_SECURE_PAGE
Page=DOC_MANAGEMENT_LINKS
Action=GetTemplatePage
```

See Also

– *LOAD_DOC_ENVIRONMENT (page 5-14)*

LOAD_GLOBALINCLUDES

Service that returns the portal page content. This service is used during the page assembly process for dynamic pages. Using the Web Layout Editor applet, select **Options—Update Portal**. The content shown on that screen is the content that is returned from content server if you run this service.

Location: <*install_dir*>/shared/config/resources/std_services.htm

PAGE_HANDLER

Service that rebuilds the static web layout structure as defined in the Web Layout Editor. Returns the entire page list in HDA format. This service is asynchronous and the action is performed in a background thread. This service can only be executed in the context of the content server and only completes successfully if a connection is made to the content server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IsRebuild: Enables a rebuild.
  - Must be set to *true* (1) for the server to rebuild the web layout pages.
  - Default is *false* (0).
- PageFunction: Must be set to *GetPageList*.

Example

- IdcCommand command file format (rebuild the web layout pages):

```
# Rebuild the entire search index
IdcService=PAGE_HANDLER
PageFunction=GetPageList
IsRebuild=1
```

- HDA format (rebuild the web layout pages):

```
@Properties LocalData
IdcService=PAGE_HANDLER
PageFunction=GetPageList
IsRebuild=1
```

```
@end
```

PNE_SAVE_QUERY

Service that saves a search query in the User Profile for the current user.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

The parameters for this are exhibited in the std_page.htm in the dynamic HTML definition query_save_for_personalization_list_form_common_fields. Parameters include the following:

- queryText
- queryTitle
- sortField
- ResultCount

PNE_UPDATE_PERSONAL_URLS

Service that updates the personal URLs in the User Profile for the current user.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- Action: The action to execute. Usually set to *GetTemplatePage*.
- Page: The name of the page template.
- titleEd: The link name to be displayed in the portal navigation bar.
- websiteEd: The URL to be saved as a link.

Example

```
IdcService=PNE_UPDATE_PERSONAL_URLS
Page=PNE_PORTAL_PERSONAL_URLS_PAGE
Action=GetTemplatePage
titleEd=Stellent
websiteEd=http://www.stellent.com
```

PNE_UPDATE_PORTAL_INFO

Service that updates the System Links in the User Profile for the current user.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- Action: The action to execute. Usually set to *GetTemplatePage*.
- PageName: The name of the page template.

Example

```
IdcService=PNE_UPDATE_PORTAL_INFO
Page=PNE_PORTAL_DESIGN_PAGE
Action=GetTemplatePage
```

SAVE_GLOBALINCLUDES

Service that saves all global includes used during the page assembly process for dynamic pages. This service runs when you update the portal page from the Web Layout Editor.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IsRebuild: Enables a rebuild of the portal page.
  - Must be set to 1 (*true*) for the content server to update the portal page.
  - Default is 0 (*false*).
- PageFunction: Must be set to *GetPageList*.

Example

```
IdcService=SAVE_GLOBALINCLUDES
PageFunction=GetPageList
IsRebuild=1
```

UPDATE_RESULT_TEMPLATE

Service that updates a search results template.

When you launch the Web Layout Editor applet, select Options, select **Query Result Page**, and click **Add**, **Delete**, or **Edit**, enter template information, and click **OK**, this service is executed.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Note: See below for an example of the entries required to add a new query result template in HDA format.

Example

- IdcCommand command file format:

```
IdcService=UPDATE_RESULT_TEMPLATE
```

- HDA file format (adds a new query result template called *my_research_template*).

```
@Properties LocalData
IdcService=UPDATE_RESULT_TEMPLATE
@end
@ResultSet ResultPageUpdates
6
name
formtype
filename
outfilename
flexdata
description
my_research_template
ResultsPage


Text2<$dDocAuthor$><$dSecurityGroup$>Text1<$dDocTitle$>
testing the update template sevice
@end
```

Important: Notice that there are two blank lines between the *ResultsPage* entry and the *Text2* entry. This is required because there are no values for *filename* and *outfilename* (the blank lines define the empty values).

PROVIDER MANAGER SERVICES (CORE CONTENT SERVER)

Provider Manager services handle the providers that establish a connection to outside entities. The following services are described in this section:

- ADD_EDIT_PROVIDER (page 5-157)
- APPEND_DATABASE_AUDIT_INFO (page 5-158)
- DELETE_PROVIDER (page 5-158)
- ENABLE_DISABLE_PROVIDER (page 5-159)
- GET_ADD_EDIT_PROVIDER_FORM (page 5-160)
- GET_ALL_PROVIDERS (page 5-161)
- GET_PROVIDER_INFO (page 5-161)
- NOTIFY_CHANGE (page 5-161)
- REQUEST_SECURITYINFO (page 5-162)
- TEST_PROVIDER (page 5-162)

ADD_EDIT_PROVIDER

Service that creates a new provider. The most likely error is when the provider name is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- pName: The provider name.
- ProviderClass: The provider class.
- pDescription: The description for the provider.
- pType: The provider type:
  - **database**—An information repository server that provides an API for connecting and communicating with it. This retrieves information and enables information to be changed in the database. Examples of this type are system databases and LDAP.

- **incoming**—A connection initiated from an outside entity like a browser or client application. The server listens and is aware of incoming connections
- **outgoing**—A connection initiated to an outside entity. You can use this type to communicate between Content Server instances.
- **preview**—The API that establishes connections between content server and Preview technology like the DTM server for HTML Preview and Content Categorizer.
- **LDAP:** The Lightweight Directory Access Protocol.

Example

```
IdcService=ADD_EDIT_PROVIDER
pName=admin_provider
pType=Database
pDescription=provider description
ProviderClass=providerclass
```

APPEND_DATABASE_AUDIT_INFO

Subservice used by GET_SYSTEM_AUDIT_INFO to return information on the System Audit Info page.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *GET_SYSTEM_AUDIT_INFO (page 5-13)*

DELETE_PROVIDER

Service that deletes an existing provider. The most likely error is a provider name not in the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ pName: The provider name.

Example

```
IdcService=DELETE_PROVIDE
pName=Proxied_2_on_test13
```

ENABLE_DISABLE_PROVIDER

Service that reverses the enable/disable state of an existing provider. If the provider is enabled, the service disables it. If the provider is disabled, the service enables it.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- pName: The provider name.
- pDescription: The provider's description.
- pType: Used to create the provider type. Any component can add its own provider type. Currently, the following types are accepted:
  - **database**—An information repository server that provides an API for connecting and communicating with it. This retrieves information and enables information to be changed in the database. Examples of this type are system databases and LDAP.
  - **incoming**—A connection initiated from an outside entity like a browser or client application. The server listens and is aware of incoming connections
  - **outgoing**—A connection initiated to an outside entity. You can use this type to communicate between Content Server instances.
  - **preview**—The API that establishes connections between content server and Preview technology like the DTM server for HTML Preview and Content Categorizer.
  - **LDAP:** The Lightweight Directory Access Protocol.

Optional Service Parameters

- IsEnabled: The desired state of the provider, *true* or *false*.

Example

```
IdcService=ENABLE_DISABLE_PROVIDER
pName=Proxied_2_on_test13
```

```
pType=outgoing
```

GET_ADD_EDIT_PROVIDER_FORM

Service that returns a form for adding or editing a provider in a browser.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- isEdit: 1 (*true*)—An existing provider is edited. 0 (*false*)—A new provider is added.
- pName: The provider name. Required only when editing a provider.
- pType: The provider type:
  - **Database—**An information repository server that provides an API for connecting and communicating with it. This retrieves information and enables information to be changed in the database. Examples of this type are system databases and LDAP.
  - **Incoming—**A connection initiated from an outside entity like a browser or client application. The server listens and is aware of incoming connections
  - **Outgoing—**A connection initiated to an outside entity. You can use this type to communicate between Content Server instances.
  - **Preview—**The API that establishes connections between content server and Preview technology like the DTM server for HTML preview and Content Categorizer.
  - **LDAP:** The Lightweight Directory Access Protocol.
- ResourceTemplate: The resource template for the form. Required only when adding a provider.

Example

- If you are adding a provider, the parameters you pass are similar to the following:

```
IdcService=GET_ADD_EDIT_PROVIDER_FORM
pType=preview
ResourceTemplate=PROVIDER_ADD_PREVIEW
isEdit=0
```

- If you are editing a provider, the parameters you pass are similar the following:

```
IdcService=GET_ADD_EDIT_PROVIDER_FORM
pType=outgoing
pName=Proxied_2_on_server02
isEdit=1
```

GET_ALL_PROVIDERS

Service that returns a list of all providers.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_PROVIDER_INFO

Service that returns information about a provider.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- pName: The provider name.
- ResourceTemplate: The resource template for page display. Required only when running the service from a browser interface. For example:

  ```
  ResourceTemplate=DATABASE_PROVIDER_INFO
  ```

Example

```
IdcService=GET_PROVIDER_INFO
pName=SystemDatabase
```

NOTIFY_CHANGE

Service that notifies a content server instance of changes to another content server instance through an outgoing provider.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the content server instance to be notified.

Example

```
IdcService=NOTIFY_CHANGE
IDC_Name=Master_on_server01
```

REQUEST_SECURITYINFO

Service that returns all security and user information from the content server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

TEST_PROVIDER

Service that tests a provider.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- pName: The provider name.

Example

```
IdcService=TEST_PROVIDER
pName=provider22
```

SCHEMA SERVICES (CORE CONTENT SERVER)

These services manage the server-side publishing of JavaScript files of database tables, such as option lists. The following services are described in this section:

- ADD_SCHEMA_FIELD (page 5-163)
- ADD_SCHEMA_RELATION (page 5-164)
- ADD_SCHEMA_VIEW (page 5-164)
- ADDOREDIT_SCHEMA_TABLE (page 5-164)
- CONTROL_SCHEMA (page 5-164)
- DELETE_SCHEMA_FIELD (page 5-164)
- DELETE_SCHEMA_RELATION (page 5-165)

- DELETE_SCHEMA_TABLE (page 5-165)
- DELETE_SCHEMA_VIEW (page 5-165)
- EDIT_SCHEMA_FIELD (page 5-165)
- EDIT_SCHEMA_NODE (page 5-166)
- EDIT_SCHEMA_RELATION (page 5-166)
- EDIT_SCHEMA_VIEW (page 5-166)
- EDIT_SCHEMA_VIEW_VALUES (page 5-167)
- GET_SCHEMA_FIELD_INFO (page 5-167)
- GET_SCHEMA_FIELDS (page 5-167)
- GET_SCHEMA_RELATIONS (page 5-167)
- GET_SCHEMA_STATS (page 5-167)
- GET_SCHEMA_TABLE_INFO (page 5-168)
- GET_SCHEMA_TABLES (page 5-168)
- GET_SCHEMA_VIEW_EDIT_INFO (page 5-168)
- GET_SCHEMA_VIEW_FRAGMENT (page 5-168)
- GET_SCHEMA_VIEW_INFO (page 5-169)
- GET_SCHEMA_VIEW_VALUES (page 5-169)
- GET_SCHEMA_VIEWS (page 5-169)
- PUBLISH_SCHEMA (page 5-169)

ADD_SCHEMA_FIELD

Service that adds a field to a schema table.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schFieldName: The name of the field to be included.

ADD_SCHEMA_RELATION

Service that adds a relation to an existing schema.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schRelationName: The name of the relation to be included.

ADD_SCHEMA_VIEW

Service used to add a new view for a schema.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be added.
- schViewType: The type of the schema view to be added.
- schTableName: The name of the table that the view is related to.

ADDOREDIT_SCHEMA_TABLE

Service used to add or edit a table used in a schema.

Location: <*install_dir*>/shared/config/resources/std_services.htm

CONTROL_SCHEMA

Location: <*install_dir*>/shared/config/resources/std_services.htm

DELETE_SCHEMA_FIELD

Service used to delete a field from a schema table.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ schFieldName: The name of the field to be deleted.

DELETE_SCHEMA_RELATION

Service used to delete a relation from a schema.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ schRelationName: The name of the relation to be deleted.

DELETE_SCHEMA_TABLE

Service used to delete a table from a schema.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ schTableName: The name of the relation to be deleted.

DELETE_SCHEMA_VIEW

Service used to delete a view from a schema.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ schViewName: The name of the view to be deleted.

EDIT_SCHEMA_FIELD

Service used to edit a field in a schema.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schFieldName: The name of the field to be edited.

EDIT_SCHEMA_NODE

Service used to edit a node in a schema.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- editViewValueAction: The action to be used for the node.
- schViewName: The name of the view used for the node.

EDIT_SCHEMA_RELATION

Service used to edit a schema relation.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schRelationName: The name of the relation to be edited.
- schTable*n*Table: The number of the table to be used for the relation (for example, Table1 or Table2). If this value is not provided, no table is used for the relation.

EDIT_SCHEMA_VIEW

Service used to edit a schema view.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schViewType: The type of the view to be edited.
- schViewName: The name of the view to be edited.
- schTableName: The name of the table associated with the schema.

EDIT_SCHEMA_VIEW_VALUES

Service used to edit values in a schema view.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

GET_SCHEMA_FIELD_INFO

Service that returns information about a specific schema field.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schFieldName: The name of the field to be used.

GET_SCHEMA_FIELDS

Service that returns information about fields used in schemas.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_SCHEMA_RELATIONS

Service that returns information about relations used with schemas.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_SCHEMA_STATS

Service that gathers statistics regarding schema usage for the Edit Active Console Output Tracing Report on the System Audit Info Page.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_SCHEMA_TABLE_INFO

Service that returns information about all tables used in schemas.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_SCHEMA_TABLES

Service that returns information about the tables and information in the tables that are used in schemas.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_SCHEMA_VIEW_EDIT_INFO

Service that returns the editable information for a schema view.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

GET_SCHEMA_VIEW_FRAGMENT

Service that returns a fragment of a Schema-based option list.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

Optional Service Parameters

- schRelationName: If this field is a Dependent Choice List (DCL), this is the name of the Schema relation used to filter the list results.
- schParentValue: If this field is a DCL, this is the value of the parent field used with the relation to filter the list.

GET_SCHEMA_VIEW_INFO

Service that returns table and other information for the specified view.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

GET_SCHEMA_VIEW_VALUES

Service that displays values in the specified view.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

GET_SCHEMA_VIEWS

Service that retrieves all defined schema views.

Location: <*install_dir*>/shared/config/resources/std_services.htm

PUBLISH_SCHEMA

Service that initiates a publish of all schemas.

Location: <*install_dir*>/shared/config/resources/std_services.htm

SEARCH SERVICES (CORE CONTENT SERVER)

The Search Services are used to manage searching within Content Server. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- APPEND_SEARCH_AUDIT_INFO (page 5-170)
- GET_EXTERNAL_DOC_INFO (page 5-170)

- GET_EXTERNAL_HIGHLIGHT_INFO (page 5-171)
- GET_EXTERNAL_XML_HIGHLIGHT_INFO (page 5-171)
- GET_HIGHLIGHT_INFO (page 5-172)
- *GET_SEARCH_RESULTS (page 5-174)
- GET_SEARCH_RESULTS_FORCELOGIN (page 5-175)
- GET_XML_HIGHLIGHT_INFO (page 5-175)
- PNE_GET_SEARCH_RESULTS (page 5-176)
- VIEW_DOC (page 5-177)

APPEND_SEARCH_AUDIT_INFO

Subservice used by GET_SYSTEM_AUDIT_INFO to return information on the System Audit Info page.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *GET_SYSTEM_AUDIT_INFO (page 5-13)*

GET_EXTERNAL_DOC_INFO

Service that retrieves content information from an external Verity collection. Used with the Lightly Managed Content component. Verity is an add-on product available for purchase for use with Content Server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
- sCollectionID: The collection ID used by the content server to locate the collection.

Example

```
IdcService=GET_EXTERNAL_DOC_INFO
dDocName=adminform113
```

GET_EXTERNAL_HIGHLIGHT_INFO

Service that returns PDF or HTML highlight information for a content item in an external collection.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
- HighlightType: The highlight type:
  - For PDF, use *PdfHighlight*.
  - For HTML, use *HtmlHighlight*.
- QueryText: The full-text search expression.
- SortField: The name of the metadata field to sort on.
  - Examples: *dInDate*, *dOutDate*, *alternateFile*.
  - Defaults to *DocId*.
  - SortOrder: The sort order. Allowed values are *ASC* (ascending) and *DESC* (descending).

Example

```
IdcService=GET_EXTERNAL_HIGHLIGHT_INFO
dDocName=test113
HighlightType=PdfHighlight
QueryText=test
SortField=dInDate
SortOrder=Desc
```

GET_EXTERNAL_XML_HIGHLIGHT_INFO

Service that returns XML highlight information for a content item in an external collection.This service is called if a user is doing a full-text search for a PDF document in a browser environment. It occurs when the user selects the link for the PDF on the result page and views the highlighted word on the PDF.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
- HighlightType: The highlight type:
  - For PDF, use *PdfHighlight*.
  - For HTML, use *HtmlHighlight*.
- QueryText: The full-text search expression.
- SortField: The name of the metadata field to sort on.
  - Examples: *dInDate*, *dOutDate*, *alternateFile*.
  - Defaults to *DocId*.
- SortOrder: The sort order. Allowed values are *ASC* (ascending) and *DESC* (descending).

Example

```
IdcService=GET_EXTERNAL_XML_HIGHLIGHT_INFO
dDocName=test113
HighlightType=PdfHighlight
QueryText=test
SortField=dInDate
SortOrder=Desc
```

GET_HIGHLIGHT_INFO

Service that returns PDF or HTML highlight information for a content item.

This service is run from the browser interface when you do a full-text search and click the Content ID or thumbnail on the search result page. On the displayed page, the words that you searched for are highlighted. For HTML document the words are bold and for PDF document they are highlighted.

The most likely errors are when the content item no longer exists in the system or if the user fails the security check.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

- QueryText: The full-text search expression.
- SortField: The name of the metadata field to sort on.
  - Examples: *dInDate*, *dOutDate*, *alternateFile*.
  - Defaults to *DocId*.
- SortOrder: The sort order. Allowed values are *ASC* (ascending) and *DESC* (descending).

Optional Service Parameters

- dWebExtension: The file extension of the web-viewable content. For example, *html*, *pdf*, or *txt*.
- HighlightType: The highlight type:
  - For PDF, use *PdfHighlight*.
  - For HTML, use *HtmlHighlight*.

Example

- IdcCommand command file format:

```
IdcService=GET_HIGHLIGHT_INFO
dDocName=test_000043
QueryText=service
SortField=dInDate
SortOrder=Desc
```

- HDA format with optional parameters:

```
@Properties LocalData
IdcService=GET_HIGHLIGHT_INFO
dDocName=test_000043
QueryText=service
SortField=dInDate
SortOrder=Desc
HighlightType=HtmlHighlight
dWebExtension=html
@end
```

GET_SEARCH_RESULTS

Service that returns a list of content items that match specific search criteria.

Access Level: Read (1)

Calls Subservice: SUB

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- QueryText: The search expression.

  You can append values for Title, Content ID, and so forth, on the *QueryText* parameter to refine this service.

Optional Service Parameters

- ResultCount: The number of results to return. Defaults to 25.
- SearchEngineName: The name of the search engine to be used. The default is the value specified in the config/config.cfg file.

  Values can be `verity`, `databasefulltext`, or `database`. If set to `database` or `databasefulltext`, you must pass SQL in the QueryText parameter, as in this example:

  ```
  dDocTitle like 'test'
  ```

  This is equivalent to the Verity query:

  ```
  dDocTitle <substring> 'test'
  ```

- SortField: The name of the metadata field to sort on.
  - Examples: *dInDate*, *dDocTitle*, *Score*.
  - Defaults to *dInDate*.
- SortOrder: The sort order. Allowed values are *ASC* (ascending) and *DESC* (descending).
- SortSpec: Enables sorting on more than one field. Set this parameter to the following sequence:

  ```
  <sort field> <sort order> <sort field> <sort order>...
  ```

  For example, `SortSpec=dDocTitle ASC dInDate DESC`.

- StartRow: The row to begin the search results display. For example, if *ResultCount=25*, setting *StartRow=26* displays the second page of results.

❖ EndRow: The row to end the search results display.

Example

```
IdcService=GET_SEARCH_RESULTS
QueryText=benefits
```

See Also

– *PNE_GET_SEARCH_RESULTS (page 5-176)*

– *GET_SEARCH_RESULTS_FORCELOGIN (page 5-175)*

GET_SEARCH_RESULTS_FORCELOGIN

Service that forces the user to be logged in before retrieving the search results. This service is equivalent to GET_SEARCH_RESULTS.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *GET_SEARCH_RESULTS (page 5-174)*

GET_XML_HIGHLIGHT_INFO

Service that returns XML highlight information for a content item.

This service is called if a user is doing a full-text search for a PDF document in a browser environment. When the user selects the link for the PDF on the result page and views the highlighted word on the PDF.

The most likely error is a content item name that does not exist.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dDocName: The Content ID for the content item.

❖ HighlightType: The highlight type:

- For PDF, use *PdfHighlight*.
- For HTML, use *HtmlHighlight*.

❖ QueryText: The full-text search expression.

- SortField: The name of the metadata field to sort on.
  - Examples: *dInDate*, *dOutDate*, *alternateFile*.
  - Defaults to *DocId*.
- SortOrder: The sort order. Allowed values are *ASC* (ascending) and *DESC* (descending).

Example

```
IdcService=GET_XML_HIGHLIGHT_INFO
dDocName=test13
QueryText=service
SortField=dInDate
SortOrder=Desc
dDocName=TEST13
HighlightType=PdfHighlight
```

PNE_GET_SEARCH_RESULTS

Service used to retrieve the search results where the presentation is controlled by the user's preferences (that is, the PNE settings).

The QueryText for the service targets the defined search engine and as such can use all the parameters that can be used by the GET_SEARCH_RESULTS service.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- QueryText: The search expression.

Example

```
IdcService=PNE_GET_SEARCH_RESULTS
QueryText=benefits
```

See Also

– *GET_SEARCH_RESULTS (page 5-174)*

VIEW_DOC

Service that returns highlight information.

- Given a content item name, the service evaluates security information and displays the content item highlight information.
- Usually this service is used when you have an external collection rather than a content server search collection. But in either case, the search collection must be created by the Verity search engine. Verity is an add-on product available for purchase for use with Content Server. For example, if you want to do a search on an external collection, you must provide the VdkVgwKey and sCollectionID parameters so the collection can be located by the content server.
- Usually, the VdkVgwKey is the *dDocName*. Verity gives a unique ID to each content item, and the content server maps that Verity ID (*VdkVgwKey*) with *dDocName* (Content ID).

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- QueryText: The search expression.
- sCollectionID: The collection ID used by the content server to locate the collection.
- SortField: The name of the metadata field to sort on.
  - Examples: *dInDate*, *dOutDate*, *alternateFile*.
  - Defaults to *docID*.
- SortOrder: The sort order. Allowed values are *Asc* (ascending) and *Desc* (descending).

Optional Service Parameters

- ViewType: The display format. If this parameter is not defined, *ViewText* is used by default.
  - Set to *ViewText* to display as Text.
  - Set to *ViewHtml* to display as HTML.

Example

```
IdcService=VIEW_DOC
IsCollectionID=external
SortField=dInDate
```

```
SortOrder=Desc
QueryText=test
ViewType=ViewText
```

USER SERVICES (CORE CONTENT SERVER)

User Services assist in managing user-related activities, such as adding accounts, aliases, and roles. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- ADD_ALIAS (page 5-179)
- ADD_GROUP (page 5-181)
- ADD_ROLE (page 5-181)
- ADD_USER (page 5-182)
- CHANGE_USER_AUTH_TYPE (page 5-184)
- *CHECK_USER_CREDENTIALS (page 5-185)
- DELETE_ALIAS (page 5-189)
- DELETE_GROUP (page 5-190)
- DELETE_ROLE (page 5-190)
- DELETE_USER (page 5-191)
- EDIT_ALIAS (page 5-191)
- EDIT_GROUP (page 5-194)
- EDIT_ROLE (page 5-194)
- EDIT_USER (page 5-195)
- *EDIT_USER_PROFILE (page 5-198)
- GET_ALIASES (page 5-200)
- GET_FILTER_ADMIN_PAGE (page 5-200)
- GET_SELF_REGISTER_PAGE (page 5-200)
- *GET_USER_INFO (page 5-200)
- GET_USERS (page 5-202)
- LOAD_PNE_PORTAL (page 5-202)

- *LOAD_USER_TOPIC (page 5-203)
- QUERY_GROUP (page 5-203)
- QUERY_USER_ATTRIBUTES (page 5-203)
- REGISTER_USER (page 5-204)
- *SAVE_USER_TOPICS (page 5-205)
- UPDATE_FILTER_INFO (page 5-209)
- UPDATE_USEROPTION_LIST (page 5-209)

ADD_ALIAS

Used to create a new alias. The most likely errors are mismatched parameters or when a user or alias already exists in the system.

If you add an alias using the User Admin applet, you must add at least one user. However, you do not need to add a user using IdcCommand or IdcCommandX. To add users using IdcCommand or IdcCommandX, the optional parameter *AliasUsersString* must be included.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dAlias: The alias name.
- dAliasDescription: The alias description.

Optional Service Parameters

- AliasUsersString: The alias or user. To add multiple users:
  - In HDA format: Use the `\n` escape sequence between each user name when using HDA format, as in this example:
  - `AliasUsersString=sysadmin\nuser1`
  - When using SOAP format, put the unencoded line feed directly into the XML or encode the line feed as `(
)`.

Example

To create an alias and add a user the required parameters are:

```
IdcService=ADD_ALIAS
dAlias=my_alias
dAliasDescription=admin users
AliasUsersString=sysadmin
```

In HDA format, adding a user with the *AliasUsersString* parameter:

```
@Properties LocalData
IdcService=ADD_ALIAS
dAlias=my_alias
dAliasDescription=admin users
AliasUsersString=sysadmin
@end
```

In HDA format, adding a user with the AliasUserMap and Alias ResultSets:

```
@Properties LocalData
IdcService=ADD_ALIAS
changedSubjects=aliases,1008291537850
dAliasDescription=Test Alias
refreshSubjects=
blFieldTypes=StatusMessage message
blDateFormat=M/d{/yy} {h:mm[:ss] {aa}[zzz]}!tAmerica/Chicago!mAM,PM
loadedUserAttributes=1
dUser=sysadmin
IsJava=1
changedMonikers=
refreshSubMonikers=
refreshMonikers=
dAlias=NewAlias
@end
@ResultSet AliasUserMap
2
dAlias 6 30
dUserName 6 50
YourAlias
Gillian
@end
@ResultSet Alias
```

```
2
dAlias 6 30
dAliasDescription 6 50
MyAlias
This is a test of adding an alias
NewAlias
Test Alias
YourAlias
Test Alias
@end
```

ADD_GROUP

Service that creates a new security group. The most likely error is when the security group name is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dGroupName: The security group name.
- dDescription: The security group description.
- dPrivilege: The permission setting.

  If this value is set to 0, only the *admin* role has RWDA permission on the security group; no other roles are able to access the security group. If the value is set to 15, all roles get RWDA permission on the security group. For more information about numbering of permissions, see the *Managing Security and User Access Guide*.

Example

```
IdcService=ADD_GROUP
dGroupName=NEW_GROUP
dPrivilege=15
dDescription=admin privileges
```

ADD_ROLE

Service that creates a new role. The most likely error is when the role name is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dRoleName: The role name.
- dPrivilege: The permission setting.

  If this value is set to 0 only the *admin* role has RWDA permission on the group. No other roles are able to access the group. If the value is set to 15 all roles get RWDA permission on the group. For more information about numbering of permissions, see the *Managing Security and User Access Guide*.

Example

```
IdcService=ADD_ROLE
dRoleName=specialuser
dPrivilege=15
```

ADD_USER

Service used to create a new user. The most likely error is when the user name is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dName: The user name.
- dUserAuthType: The user authorization type. This value must be set to either *Local* or *Global*.

Optional Service Parameters

- dFullName: The full name of the user.
- dPassword: The password for the user.
- dEmail: The email address for the user.

Optional Attribute Information

Optional attribute information is specified in a ResultSet that contains the user's attribute information and references the roles the user belongs to and the accounts the user has access to. Attribute information consists of a list of three comma-separated strings. The first string indicates the type of attribute, the second the name of the attribute, and the third is the access number for accounts or default entry for role.

Important: The user attribute information is not predefined. By default, a new user belongs to no roles or accounts, and becomes a guest in the system.

- **Attribute Type**: When defining a role, the first string specifies that this is a role attribute, the second string is the name of the role, and the third is the default entry of 15.
  - When defining an account, the first string specifies that this is an account attribute, the second string is the name of the account, and the third is the access level.
  - For a role attribute, the information is in the form:
    `role,contributor,15`
  - For an account attribute, the information is in the form:
    `account,books,1`
- **Attribute Name**: A user can belong to multiple roles and accounts, so there can be multiple role and account information strings separated by commas in the attribute information column. If the user is to have the *admin* role, define the user attribute information as follows:

```
@ResultSet UserAttribInfo
2
dUserName
AttributeInfo
jsmith
role,contributor,15
@end
```

  If the user is to belong to both the *contributor* and *editor* roles and have Read permission on the *books* account, define the user attribute information as:

```
@ResultSet UserAttribInfo
2
dUserName
AttributeInfo
```

```
jsmith
role,contributor,15,role,editor,15,account,books,1
@end
```

- **Access Number**: These access numbers can be assigned to the user.
  - 1: Read only
  - 3: Read and write
  - 7: Read, write, delete
  - 15: Administrative permissions

Example

- IdcCommand command file format:

```
IdcService=ADD_USER
dName=specialuser
dUserAuthType=LOCAL
```

- HDA format with optional parameters:

```
@Properties LocalData
IdcService=ADD_USER
dName=jsmith
dUserAuthType=LOCAL
dFullName=Jennifer Smith
dPassword=password
dEmail=jsmith@stellent.com
@end
@ResultSet UserAttribInfo
2
dUserName
AttributeInfo
jsmith
role,admin,15,role,contributor,15
@end
```

CHANGE_USER_AUTH_TYPE

Service that changes the authentication type (global, local, or external) for users.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- curUserAuthType: Current user authentication type.
- dUserAuthType: New user authentication type.

CHECK_USER_CREDENTIALS

Service that checks the credentials of a user. This service can be used to get information about a user (for example, roles and accounts), or it can be used to check a user/password pair to see if they match. This is called during the authentication process. The data is cached so it is not called for every service request.

Access Level: N/A (0)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- userName: The user name

Optional Service Parameters

- getUserInfo: 0 (*false*)—Does not retrieve extended user information. 1 (*true*)—Retrieves extended user information (full name, e-mail address, locale, and user type)
- hasSecurityInfo: 0 (*false*)—Retrieves the user accounts. 1 (*true*)—Does not retrieve the user accounts.
- authenticateUser: 0 (*false*)—Does not authenticate the user (validate that the user/password pair match). 1 (*true*)—Authenticates the user (validate that the user/password pair match). In this case, the *userPassword* parameter must be specified.
- userPassword: The password for the specified user name. If the *authenticateUser* parameter is included and set to 1 (*true*), the *userPassword* parameter must be specified.
- userExtendedInfo: This parameter, combined with the getUserInfo parameter, returns the user's accounts and groups in the data binder. 0 (*false*)—Does not retrieve the information. 1 (*true*)—Retrieves the information.

Results

❖ Local Data:

- extendedInfo: HDA-encoded string containing user info fields (ndUserLocale, nblFieldTypes, ndEmail, nblDateFormat, ndFullName, ndUserType, ndUserAuthType, ndUserChangeDate, ndUserTimeZone, and ndUserArriveDate). Only returned if optional parameter 'getUserInfo' is 1 (*true*).
- accounts: Comma-separated list of user's accounts. Only returned if optional parameter *hasSecurityInfo* is 0 (*false*).
- roles: Comma-separated list if user's roles. Only returned if optional parameter 'hasSecurityInfo' is 0 (*false*).
- hasSecurtyInfo: Returns 1 (*true*) if security information is present in response.
- isAuthenticated: Returns 1 (*true*) if optional parameter *authenticateUser* is set to 1 (*true*), and the optional parameter *userPassword* matches the password that corresponds to *userName*.
- isPromptLogin: Returns 1 (*true*) if *userPassword* does not correspond to *userName*. No other data is returned in this situation.

Example

The following is an example CHECK_USER_CREDENTIALS service call to get information about a user:

```
@Properties LocalData
IdcService=CHECK_USER_CREDENTIALS
userName=user1
getUserInfo=1
@end
```

The following is an example of the results that would be returned by this service call to get user information (note that the accounts are returned):

```
Content-type: text/plain
<?hda version="6.3 dev (build-date)" jcharset=Cp1252 encoding=iso-8859-1?>
@Properties LocalData
dUser=user1
blFieldTypes=
dUserOrgPath=
refreshSubMonikers=
accounts=#none,prj(RWD)
```

```
blDateFormat='{ts' ''yyyy-MM-dd HH:mm:ss'''}''!tAmerica/Chicago
StatusCode=0
dUserSourceOrgPath=
hasSecurityInfo=1
roles=contributor
refreshSubjects=
changedSubjects=
dName=user1
refreshMonikers=
changedMonikers=
extendedInfo=\@Properties LocalData\nu9=\nu8=\nu7=\ndUserArriveDate={ts '2003-02-
11
08:34:35'}\nu6=\ndUserTimeZone=\nu5=\nu4=\nu3=\nu2=\ndUserAuthType=LOCAL\nu1=\ndUs
erType=\ndUserChangeDate={ts '2003-04-03
11:57:29'}\nuPhone=\nuCompany=\ndUserLocale=English-
US\nblFieldTypes=\nu24=\nu23=\nu22=\nu21=\nu20=\ndFullName=user1\nblDateFormat=M/d
{/yyyy} {h:mm{:ss}{.SSSS}
{aa}}!tAmerica/Chicago\ndEmail=\nu19=\nu18=\nu17=\nu16=\nu15=\nu14=\nu13=\nu12=\nu
11=\nu10=\n\@end\n\n
@end
```

The following is an example of the results that would be returned by this service call to get user information (note that the accounts are ***not*** returned):

```
Content-type: text/plain
<?hda version="6.3 dev (build-date)" jcharset=Cp1252 encoding=iso-8859-1?>
@Properties LocalData
dUser=user1
blFieldTypes=
refreshSubMonikers=
blDateFormat='{ts' ''yyyy-MM-dd HH:mm:ss'''}''!tAmerica/Chicago
StatusCode=0
changedSubjects=
refreshSubjects=
dName=user1
changedMonikers=
refreshMonikers=
extendedInfo=\@Properties LocalData\nu9=\nu8=\nu7=\ndUserArriveDate={ts '2003-02-
11
08:34:35'}\nu6=\ndUserTimeZone=\nu5=\nu4=\nu3=\nu2=\ndUserAuthType=LOCAL\nu1=\ndUs
erType=\ndUserChangeDate={ts '2003-04-03
11:57:29'}\nuPhone=\nuCompany=\ndUserLocale=English-
```

```
US\nblFieldTypes=\nu24=\nu23=\nu22=\nu21=\nu20=\ndFullName=user1\nblDateFormat=M/d
{/yyyy} {h:mm{:ss}{.SSSS}
{aa}}!tAmerica/Chicago\ndEmail=\nu19=\nu18=\nu17=\nu16=\nu15=\nu14=\nu13=\nu12=\nu
11=\nu10=\n\@end\n\n
@end
```

The following is an example CHECK_USER_CREDENTIALS service call to see if a user name/password pair match:

```
@Properties LocalData
IdcService=CHECK_USER_CREDENTIALS
userName=user1
authenticateUser=1
userPassword=idc
@end
```

The following is an example of the results that would be returned by this service call to check a user name/password pair match (note that the user name and password match; the key returned value being *isAuthenticated=1*):

```
Content-type: text/plain
<?hda version="6.3 dev (build-date)" jcharset=Cp1252 encoding=iso-8859-1?>
@Properties LocalData
dUser=user1
blFieldTypes=
refreshSubMonikers=
blDateFormat='{ts' ''yyyy-MM-dd HH:mm:ss'''}'!tAmerica/Chicago
StatusCode=0
changedSubjects=
refreshSubjects=
dName=user1
changedMonikers=
refreshMonikers=
isAuthenticated=1
@end
```

The following is another example CHECK_USER_CREDENTIALS service call to see if a user name/password pair match:

```
@Properties LocalData
IdcService=CHECK_USER_CREDENTIALS
userName=user1
```

```
authenticateUser=1
userPassword=pppp
@end
```

The following is an example of the results that would be returned by this service call to check a user name/password pair match (note that the user name and password ***do not*** match; *isPromptLogin=1* is returned instead of *isAuthenticated=1*):

```
Content-type: text/plain
<?hda version="6.3 dev (build-date)" jcharset=Cp1252 encoding=iso-8859-1?>
@Properties LocalData
dUser=user1
blFieldTypes=
refreshSubMonikers=
blDateFormat='{ts' ''yyyy-MM-dd HH:mm:ss'''}'!tAmerica/Chicago
StatusCode=0
isPromptLogin=1
changedSubjects=
refreshSubjects=
dName=user1
changedMonikers=
refreshMonikers=
@end
```

See Also

– *GET_USER_INFO (page 5-200)*

DELETE_ALIAS

Service that deletes an existing alias. The most likely errors are mismatched parameters, when the alias is being used in the workflow, or when the system is unable to delete the alias or the user.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ dAlias: The alias name.

Example

```
IdcService=DELETE_ALIAS
dAlias=admin_alias
```

DELETE_GROUP

Service that deletes an existing security group.The most likely errors are when content items or workflows associated with this group still exist in the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dGroupName: The security group name.

Example

```
IdcService=DELETE_GROUP
dGroupName=admingroup
```

DELETE_ROLE

Service that deletes an existing role. The most likely errors are when the specified role does not exist or when a user still has this role assigned.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dRoleName: The role name.

Example

```
IdcService=DELETE_ROLE
dRoleName=test_role
```

DELETE_USER

Service that deletes an existing user. The most likely error is when the user has been assigned to an alias.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dName: The user name.

Example

- IdcCommand command file format:

```
IdcService=DELETE_USER
dName=jsmith
```

- HDA format:

```
@Properties LocalData
IdcService=DELETE_USER
dName=jsmith
@end
```

EDIT_ALIAS

Service that modifies an existing alias. To edit and add users, the *AliasUsersString* parameter must be included.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Caution: Editing an alias deletes all existing data (Description and Users) from the alias. To retain existing data, you must include the existing data along with any new data.

Additional Required Service Parameters

- dAlias: The alias name.
- dAliasDescription: The alias description.
- AliasUsersString: The alias or user. To add multiple users:

- In HDA format: Use the `\n` escape sequence between each user name when using HDA format, as in this example:

  ```
  AliasUsersString=sysadmin\nuser1
  ```

- When using SOAP format, put the unencoded line feed directly into the XML or encode the line feed as `(
)`.

Example

Edits an alias:

```
IdcService=EDIT_ALIAS
dAlias=my_alias
dAliasDescription=new user
AliasUsersString=sysadmin\nuser1
```

❖ In HDA format, adding users with the *AliasUsersString* parameter:

```
@Properties LocalData
monitoredTopics=appcommongui,1000382277000
IdcService=EDIT_ALIAS
dAliasDescription=Test Alias
blFieldTypes=
blDateFormat=M/d{/yy} {h:mm[:ss] {aa}[zzz]}!tAmerica/Chicago!mAM,PM
monitoredSubjects=usermetaoptlists,1008541017549,aliases,1008541017549,userlist,10
08541017549,users,1008541017549,metadata,1008541017549,config,1008541017549,accoun
ts,1008541017549
watchedMonikers=
AliasUsersString=Gillian\nMonique\nsysadmin\n
dAlias=YourAlias
@end
```

In HDA format, adding users with the AliasUserMap and Alias ResultSets:

```
@Properties LocalData
refreshMonikers=
IsJava=1
refreshSubMonikers=
refreshSubjects=aliases,1008541017551
dAlias=YourAlias
refreshTopics=
AliasUsersString=Gillian\nMonique\nsysadmin\n
```

```
ClientEncoding=Cp1252
dUser=sysadmin
blDateFormat=M/d{/yy} {h:mm[:ss] {aa}[zzz]}!tAmerica/Chicago!mAM,PM
monitoredSubjects=usermetaoptlists,1008541017549,aliases,1008541017549,userlist,10
08541017549,users,1008541017549,metadata,1008541017549,config,1008541017549,accoun
ts,1008541017549
loadedUserAttributes=1
dUserName=sysadmin
watchedMonikers=
IdcService=EDIT_ALIAS
blFieldTypes=StatusMessage message
changedMonikers=
dAliasDescription=Test Alias
changedSubjects=
monitoredTopics=appcommongui,1000382277000
@end
@ResultSet AliasUserMap
2
dAlias 6 30
dUserName 6 50
YourAlias
Gillian
YourAlias
Monique
YourAlias
sysadmin
@end
@ResultSet UpdatedUserTopics
3
topicName
topicValue
topicTS
@end
@ResultSet Alias
2
dAlias 6 30
dAliasDescription 6 50
MyAlias
This is a test of adding an alias
```

```
NewAlias
Test Alias
YourAlias
Test Alias
@end
```

EDIT_GROUP

Service that modifies an existing security group.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dGroupName: The security group name.
- dDescription: The security group description.
- dPrivilege: The permission setting.

  If this value is set to 0 only the *admin* role has RWDA permission on the group. No other roles are able to access the group. If the value is set to 15, all roles get RWDA permission on the group.

Example

```
IdcService=EDIT_GROUP
dGroupName=MY_GROUP
dPrivilege=0
dDescription=admin privileges
```

EDIT_ROLE

Service that modifies an existing role.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dRoleName: The existing role name.
- dGroupName: The name of the security group.

Optional Service Parameters

- dPrivilege: The permission setting.

  If this value is set to 0 only the *admin* role has RWDA permission on the group. No other roles are able to access the group. If the value is set to 15, all roles get RWDA permission on the group.

Example

```
IdcService=EDIT_ROLE
dRoleName=test_role
dGroupName=MY_GROUP
dPrivilege=15
```

EDIT_USER

Service that modifies an existing user. The most likely error is the user not having the security level to perform this action or the user not existing.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Caution: Editing a user deletes all existing user attributes (role and account permissions). To retain existing attributes, you must include the existing attributes along with any new or changed data. If user attributes are not defined, the user belongs to no roles or accounts, and becomes a guest in the system.

Additional Required Service Parameters

- dName: The user name.
- dUserAuthType: The user authorization type. This value must be set to either *Local* or *Global*.

Optional Service Parameters

- dFullName: The full name of the user.
- dPassword: The password for the user.
- dEmail: The email address of the user.
- dUserLocale: The locale designation, such as *English-US*, *English-UK*, *Deutsche*, *Français*, or *Español*.

- dUserType: The user type.

Optional Attribute Information

Optional attributes are specified as a ResultSet that contains the user's attribute information and specifies the roles the user belongs to and the accounts the user has access to. Attribute information consists of a list of three comma-separated strings. The first string indicates the type of attribute, the second the name of the attribute, and the third is the access number for accounts or default entry for role.

- Attribute Type: When defining a role, the first string specifies that this is a role attribute, the second string is the name of the role, and the third is the default entry of 15. When defining an account, the first string specifies that this is an account attribute, the second string is the name of the account, and the third is the access level.
  - For a role attribute, the information is in the form:
  - role,contributor,15
  - For an account attribute, the information is in the form:

    ```
    account,books,1
    ```

- Attribute Name: A user can belong to multiple roles and accounts, so there can be multiple role and account information strings separated by commas in the attribute information column. If the user is to have the admin role, define the user attribute information as follows:

  ```
  @ResultSet UserAttribInfo
  2
  dUserName
  AttributeInfo
  jsmith
  role,contributor,15
  @end
  ```

  If the user is to belong to both the *contributor* and *editor* roles and have Read permission on the *books* account, define the user attribute information as:

  ```
  @ResultSet UserAttribInfo
  2
  dUserName
  AttributeInfo
  jsmith
  role,contributor,15,role,editor,15,account,books,1
  ```

```
@end
```

- Access Number: These access numbers can be assigned to the user.
  - 1: Read only
  - 3: Read and write
  - 7: Read, write, delete
  - 15: Administrative permissions

Example

- IdcCommand command file format:

```
IdcService=EDIT_USER
dName=user20
dUserAuthType=Local
```

- HDA format with optional parameters and attribute information:

```
@Properties LocalData
IdcService=EDIT_USER
dName=jsmith
dFullName=Jennifer Smith
dUserAuthType=Local
dPassword=password
dEmail=jsmith@stellent.com
dUserType=MKT
dUserLocale=English-US
@end

@ResultSet UserAttribInfo
2
dUserName
AttributeInfo
jsmith
role,admin,15,role,contributor,15
@end
```

EDIT_USER_PROFILE

Service that modifies the user profile for an existing user. and saves profile settings. This cannot be used to change a user's security credentials.

Access Level: Read, Global (17)

Location: <*install_dir*>/shared/config/resources/std_services.htm

- Any of the fields in the Users table (except dName) can be updated by this service.
- This service can be used to edit personalization information that is not stored in the database. Some of a user's information is stored in the database and some is stored in personalization data .hda files.
- For information in the database, simply passing the corresponding field name and a value causes the field to be updated.
- For personalization data, encoded topic strings must be used (for example, `topicString1=updateKeys:pne_portal:lm_Layout:Trays`).
- The Password value cannot be updated for proxied users.
- The most likely reason this service would fail is if it cannot find the user (dName) in the system, or if the user name passed in dName does not match the current user.

Additional Required Service Parameters

- dName: The user name.

Optional Service Parameters

- dFullName: The full name of the user.
- dPassword: The password for the user.
- dEmail: The email address of the user.
- dUserLocale: The locale designation, such as *English-US*, *English-UK*, *Deutsche*, *Français*, or *Español*.
- dUserType: The user type.
- emailFormatList: Set to *HTML* for HTML-based emails or *text* for text-based email.
- numTopics: The number of additional personalization topics in this request.
- topicString1: An encoded string of a personalization topic to update.
- numTopics: The total number of topic strings being passed.

- Any field from the User's table (except dName): dFullName, dEmail, dPasswordEncoding, dPassword, dUserType, dUserAuthType, dUserOrgPath, dUserSourceOrgPath, dUserSourceFlags, dUserArriveDate, dUserChangeDate, dUserLocale, dUserTimeZone.
- topicString1 - topicStringN: Any number of strings encoded to define personalization topic edit operations.
- RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.

Optional parameters for version 7.0 and later:

- Layout: The API to use (for example, *Top Menu* or *Trays*).
- Skin: The skin to use (for example, *Windows*).
- XuiSearchTemplate: The template to use for the user's personalized Search Results page.

Results

- Result Sets: UserTopicEdits (Returned only if personalization data was edited. Contains info about what topic edits were performed. Has the following fields: topicName, topicEditAction, topicKey, topicValue. Note that topicValue is an HDA-encoded string.)
- Local Data:
  - changedSubjects
  - dUser
- Response Template:
  - REDIRECT_TEMPLATE (redirect_template.htm)
  - Default redirect service: GET_DOC_PAGE (Page=HOME_PAGE)

Used By

- Resource Includes: user_info_submit_form

Example

- IdcCommand command file format:

```
IdcService=EDIT_USER_PROFILE
dName=sysadmin
```

- HDA format with optional parameters:

```
@Properties LocalData
IdcService=EDIT_USER_PROFILE
dName=jsmith
dFullName=Jennifer Smith
dPassword=password
dEmail=jsmith@stellent.com
dUserType=MKT
dUserLocale=English-US
@end
```

GET_ALIASES

Service that returns a list of all aliases.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_FILTER_ADMIN_PAGE

Service that retrieves the Configure Web Server Filter page in a browser.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_SELF_REGISTER_PAGE

Service that returns the self-registration page and loads the user information for a self-registered user.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_USER_INFO

Service that returns the User Profile page for the current user. Use CHECK_USER_CREDENTIALS (page 5-185) to verify the roles and accounts for the user.

This service takes no parameters and derives its information based upon the value of the dUser server variable.

Access Level: Read, Global, Scriptable (49)

Queries Executed: Quser

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results:

- Result Sets:
  - USER_INFO (fields from Users database table for current user)
  - UserMetaDefinition (result set that defines properties of user meta fields; matches contents of usermeta.hda file in data directory)
  - LmLayouts (listing of names of available layouts; fields: layout)
  - LmLayoutSkinPairs (listing of all available layout/skin combinations; fields: layout, skin)
- Option Lists Returned: Users_UserLocaleList
- Local Data:
  - dName
  - dUser
- Response Template: USER_INFO (user_info.htm)

Used By

- Resource Includes: pne_nav_userprofile_links
- Templates:
  - MY_VIEW_FORM (my_view_form.htm)
  - PNE_PORTAL_DESIGN_PAGE (pne_portal_design_page.htm)
  - PNE_PORTAL_DOC_PROFILES_PAGE (pne_portal_doc_profiles_page.htm)
  - PNE_PORTAL_PERSONAL_URLS_PAGE(pne_portal_personal_urls_page.htm)
  - PNE_PORTAL_SAVED_QUERIES_PAGE (pne_portal_saved_queries_page.htm)
  - PNE_PORTAL_SYSTEM_LINKS_PAGE (pne_portal_system_links_page.htm)
  - std_home_page.htm
  - SUBSCRIPTION_LIST (subscription_list.htm)
- Standard Navigation: commonNav.js

See Also

– CHECK_USER_CREDENTIALS *(page 5-185)*

GET_USERS

Service that returns a list of all users and their attributes. Returns a list of all users registered in the system with their primary attributes: user name, full name, password, email address, directory, type, and password encoding.

Location: <*install_dir*>/shared/config/resources/std_services.htm

LOAD_PNE_PORTAL

Loads all data from the user's PNE file (pne_portal.htm), including standard personalization properties and result sets, and custom personalization data, for a user's saved queries. This is used by remote applications to obtain a user's personalization settings.

This is used as a remote application service. It is a raw data service with no associated template display.

Access Level: Read, Scriptable (33)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results

- Result Sets: All result sets from the user's pne_portal.htm file. Typically this includes personal URLS and saved queries, but can also include any custom personalization result sets used by the specific implementation.
- Local Data: All LocalData properties from the user's pne_portal.hda file. This includes standard personalization properties (XuiSearchTemplate, showDefaultQuery, defaultQueryRows, lm_Skin, lm_Layout, portalDesignLink, quickSearchLink, defaultQuery, searchFormType, emailFormat), but can also include any custom personalization properties used by the specific implementation.

Used By

- Other: SoapCustom:Wsdl:PortalInfo:Services

LOAD_USER_TOPIC

Loads personalization data based on the topic chosen. The topics correspond to HDA files in the <*install_dir*>/data/users/profiles/ directories.

This is used as a remote application service. It is a raw data service with no associated template display.

Access Level: Read, Scriptable (33)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- userTopic: The name of the user topic, such as *pne_portal* or *wf_in_queue*.

Results

- Result Sets: All result sets in the specified topic file.
- Local Data: All LocalData properties in the specified topic file.

QUERY_GROUP

Service that returns the description of a security group.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dGroupName: The security group name.

Example

```
IdcService=QUERY_GROUP
dGroupName=Public
```

QUERY_USER_ATTRIBUTES

Service that returns user attributes for a specific user.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dName: The user name.

Example

```
IdcService=QUERY_USER_ATTRIBUTES
dName=jsmith
```

REGISTER_USER

Service that registers a user. If only the user name parameter (*dName*) is provided, the new user is a local user with the *guest* role and a blank password.

The most likely error is a user name that is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dName: The user name.

Optional Service Parameters

- dFullName: The full name of the user.
- dPassword: The password for the user.
- dEmail: The email address for the user.
- dUserLocale: The locale designation, such as *English-US*, *English-UK*, *Deutsche*, *Français*, or *Español*.
- dUserType: The user authorization type, either *Local* or *Global.*

Example

```
IdcService=REGISTER_USER
dName=user20
```

SAVE_USER_TOPICS

Service used to save personalization information for the user. Seven actions can be performed with this service. The required parameters for the service vary depending upon which action is being performed.

A form is normally used to submit the service request because of the complexity of the data and the number of additional parameters required.

Access Level: Read, Global (17)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Actions

The following actions are used with the *topicString* parameter:

- updateKeys: Updates a single LocalData key variable. The *topicString* takes 4 values: the first 3 required values plus a fourth that contains the literal string value to be placed in the key variable.
- updateKeyByName: This action is new for Content Server version 10.0. Like updateKeys, this action updates a single LocalData key variable. It also takes 4 *topicString* values. Instead of containing a literal string, the fourth value contains the name of an additional field/parameter that contains the value to be used in the update.
- deleteKeys: Deletes a single key variable. The *topicString* takes the 3 required values only.
- updateRows: This action is used to add and/or update one or more result set rows using a single topic string. The *topicString* takes 5 values: the first 3 required values; a fourth that contains a comma-separated list of all result set column names in correct order; a fifth containing a number indicating the total number of rows being updated/added.

  A set of additional fields/parameters with names corresponding to the column names must be provided for each row of data based upon the number of rows indicated in the fifth value of the topicString. Sequential numeric suffixes must be appended to the column names of the fields in each row's data set beginning with one and ending with total row number provided in the topicString (if any of these are missing, Content Server throws an error).

  If the value for the first column (the unique key) of a row in the update data matches that of an existing row in the result set, the existing row will be updated; otherwise a new row will be appended to the bottom of the result set.

- addMruRow: The "Mru" in the name is an acronym for "most-recenly-used". This action adds a single row to the top of a result set. The number of rows in the result set is not allowed to exceed a specified maximum. If a row is added to a result set that already contains the maximum number of rows, the last row is deleted at the same time that the new one is added (essentially keeping only the N most recently used rows in a result set).

  The maximum number of rows can be specified using an optional *mruNumber* parameter. If no *mruNumber* parameter is provided, the maximum defaults to 10.

  The *topicString* takes 4 values: the first 3 required values and a fourth that contains a comma-separated list of all result set column names in correct order. Similar to updateRows, a set of additional fields/parameters with names corresponding to the column names must be provided; but since only one row is added/updated at a time, no numeric suffixes are required on the names.

  If the value of the first column (the unique key) matches that of an existing row, that row is updated and moved to the top of the result set rather than adding a new row. The number of rows doesn't change, and the affected row is now considered the most-recently-used row.

- deleteRows: Deletes one or more rows from a result set. The *topicString* takes 4 values: the first 3 required values, and a fourth that contains a comma-separated list of strings that correspond to the DATA VALUES of the first column (unique key) of existing rows in the specified result set.

- deleteSets: Deletes one or more result sets from a topic file. The *topicString* takes only 3 values, but in this case, the third value can be a comma-separated list of names rather than a single key name, allowing multiple result sets to be deleted using a single topicString.

Optional Service Parameters

- numTopics: Tells the service the number of topicStrings to expect.

- topicKeys: Identifies the topic string parameter fields by name. Composed of a colon-separated list of one or more names that correspond to fields or parameters that contain the topic strings. A numTopics value must be present when this parameter is used, even if you don’t use the enumeration method of identifying topic string values. (Any value can be used for numTopics.)

- topicString*n:* Composed of three to five values, separated by colons. The number of topic strings depends on the value in *numTopics*. For example, if *numTopics* is 2, topicString1 and topicString2 are required.

The values are as follows:

- The first value is the action to be taken by the service.
- The second value is the name of the topic file that is acted on.
- The third value is the key name of the item being modified (either the name of a LocalData key variable, or the name of a result set in the topic file).
- The fourth value contains data information for an add or update action. It either contains the data string itself, or it points to additional fields or parameters that contain the data values.
- The fifth value is used only with the updateRows action to indicate the number of rows of data being sent.

All of the update actions perform "add" operations if a key of the given name is not found in the specified topic file. If a topic file matching the name given in the second value is present, but doesn't contain a key (variable or result set) matching the key name given in the third value, then a new variable or result set will be created using the data provided for the update operation. If the topic file itself isn't found, it will also be created.

❖ RedirectUrl: Used to display another page. If omitted, the user is redirected to the Content Server home page.

❖ mruNumber: Used to specify the maximum number of rows in a result set when using the addMruRow - "Add Most-Recently-Used Row" action.

Results

❖ Result Sets: UserTopicEdits (Rows represent each of the topic edits performed by the service call. Rows contain the fields: topicName, topicEditAction, topicKey, topicValue.)

❖ Response Template:

- REDIRECT_TEMPLATE (redirect_template.htm)
- Default redirect service: GET_DOC_PAGE: HOME_PAGE

Used By

❖ Resource Includes: xui_searchapi_results_action_form

❖ Templates:

- MY_VIEW_FORM (my_view_form.htm)

- PNE_PORTAL_DOC_PROFILES_PAGE (pne_portal_doc_profiles_page.htm)
- PNE_PORTAL_PERSONAL_URLS_PAGE (pne_portal_personal_urls_page.htm)
- PNE_PORTAL_SAVED_QUERIES_PAGE (pne_portal_saved_queries_page.htm)
- PNE_PORTAL_SYSTEM_LINKS_PAGE (pne_portal_system_links_page.htm)
- WF_INQUEUE_LIST (workflow_queue.htm)

Examples

Caution: The following example is not a core configuration of Content Server. This is a custom feature extension and is only supported via the purchase of Consulting Services or through the purchase of our Developer Support Token Program. As with all customizations, when you upgrade your software, the custom changes may be overwritten.

To add a personal URL, use the following parameters and values:

```
numTopics=1
topicString1=updateRows:pne_portal:PersonalURLS:title,website:1
```

These values add rows in the PersonalURLS resultset in the *pne_portal.hda* user topic. The columns in the resultset are *title* and *website*, which are taken from the fourth value of the topic string. The last value (*1*) tells the service that one row of data is being sent.

You need to use additional parameters to the service to specify the row data. These parameters are taken from the *title* and *website* column names specified in the topicString. Because it is possible to specify more than one row of data, a numeric suffix is added to each of the column names to form the service parameter name. The first row would require the *title1* and *website1* parameters and a second row would require the *title2* and *website2* parameters.

To add a row of data that is a link to the home page, set the values of the *title1* and *website1* parameters to *Home Page* and *http://www.stellent.com*. These values must be URL-encoded if you are adding them directly to the URL in the browser address bar; if you use a form in a web page to submit the data, encoding is handled. Using a form, the complete service specification would be the following:

```
<FORM name="addPersonalUrlForm" method="POST"
action="<$HttpCgiPath$>">
<INPUT type="hidden" name="IdcService" value="SAVE_USER_TOPICS">
<INPUT type="hidden" name="numTopics" value="1">
<INPUT type="hidden" name="topicString1"
```

```
value="updateRows:pne_portal:PersonalURLS:title,website:1">
<INPUT type="hidden" name="title1" value="Stellent Home Page">
<INPUT type="hidden" name="website1" value="http://www.stellent.com">
<INPUT type="submit" value="Submit">
</FORM>
```

UPDATE_FILTER_INFO

Service that updates the web server filter settings.

Location: <*install_dir*>/shared/config/resources/std_services.htm

UPDATE_USEROPTION_LIST

Service that adds or updates an option list for a user information field.

Caution: The option list values specified in the *OptionListString* parameter replace any existing values. To retain existing values, you must include the existing values along with the new values.

Option list values for the User Type field and all custom user information fields are not stored in the database but in the *useroptions.hda* file, which is normally located in the <*install_dir*>/data/users/config/ directory.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dKey: The option list key.
- OptionListString: The list of options, separated by the *n* escape sequence.

Example

- IdcCommand command file format:

```
# Add values NY, DC, CA to UserTypeList
IdcService=UPDATE_USEROPTION_LIST
dKey=Users_UserTypeList
OptionListString=NY\nDC\nCA
```

- HDA format (adds the values *NY*, *DC*, and *CA* to *UserTypeList*):

```
@Properties LocalData
IdcService=UPDATE_USEROPTION_LIST
```

```
dKey=Users_UserTypeList
OptionListString=NY\nDC\nCA
@end
```

Chapter

6

WORKFLOW SERVICES

OVERVIEW

This chapter describes the services available when using and customizing workflows, as well as other services which are stored in the workflow.htm file in the <*install_dir*>/shared/config/resources directory.

It covers the following topics:

- About Workflow Services (page 6-1)
- Doc and General Services (Workflows) (page 6-2)
- Workflow Template Services (page 6-13)
- Workflow Services (page 6-20)

ABOUT WORKFLOW SERVICES

Note: The most commonly used services have more extensive descriptions.

A typical service entry is similar to the following:

```
GET_WORKFLOW
WorkflowService 40 null null null!csWfRetrieveInfoError(dWfName)
5:QworkflowDocuments:WfDocuments:0:!csWfRetrieveInfoError(dWfName)
3:getWorkflowStepAliasesInfo::0:null
3:getWorkflowDesignInfo::0:null
3:getWorkflowDocumentInfo:WfDocuments:0:null
```

In this example:

- The first line indicates the service name: `GET_WORKFLOW`
- The second line lists the service's attributes:
  - Service class: `WorkflowService`
  - Access level: `40`
  - Template: `null`
  - Service type: `null`
  - Subjects Notified: `null`
  - Error message and parameter used for error message: `Retrieve Info Error`
- The remainder of the entry lists the service's actions, each line designating one action:
  - action type: for example, `5`, indicating a Select Cache Query, which specifies a query to retrieve information from the database and store it for later use.
  - action name: for example, `QworkflowDocuments`
  - action parameters: for example, `WfDocuments`. If no parameters are specified, double colons (`::`) are included.
  - action control mask: for example, `0`, indicating "Ignore error". If no control mask is specified, double colons (`::`) or 0 is included.
  - action error message, or null if no message is available.

Each element of the service's action is separated by a colon (`:`).

Important: All services have at least one required parameter. The IdcService parameter takes the name of the service as its argument. If other parameters are required, they are noted in the description of the service.

See Chapter 4 (*Customizing Services)* for details about a service's attributes and actions.

DOC AND GENERAL SERVICES (WORKFLOWS)

Doc Services are those that are used to manage and manipulate documents or provide information about documents. General services are common or default services. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- ADD_PROBLEMREPORT (page 6-3)

- DELETE_PROBLEMREPORT (page 6-4)
- GET_CRITERIA_WORKFLOWS_FOR_GROUP (page 6-5)
- GET_DOCUMENT_PROBLEMREPORTS (page 6-5)
- GET_PROBLEMREPORTS_SEARCH_FORM (page 6-6)
- GET_PROBLEMREPORTS_SEARCH_RESULTS (page 6-6)
- GET_UPDATE_PROBLEMREPORT_FORM (page 6-7)
- GET_WORKFLOWDOCUMENTS (page 6-7)
- GET_WORKFLOWS_FOR_ALL (page 6-8)
- *LOAD_WORKFLOW_QUEUE (page 6-8)
- NOTIFY_CONTRIBUTOR (page 6-9)
- PROBLEMREPORT_INFO (page 6-9)
- RESEND_PROBLEMREPORT (page 6-9)
- REVIEW_WORKFLOW_DOC (page 6-10)
- UPDATE_PROBLEMREPORT (page 6-10)
- WORKFLOW_CHECKIN_SUB (page 6-11)
- WORKFLOW_EDIT_REV (page 6-11)
- WORKFLOW_NEW_REV (page 6-12)
- WORKFLOW_REJECT_FORM (page 6-12)

ADD_PROBLEMREPORT

Service that adds a problem report to a content item. This is a Content Publisher related service used on Staging Workflow.

The most likely errors are mismatched parameters, when the system is unable to add a problem report to the content item, or when the content item no longer exists in the system.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dDocName: The content item identifier (Content ID).
- dID: The generated content item revision ID.
- dPrCaption: The caption for the problem report.
- dPrSeverity: The problem report severity level such as *Critical*, *Moderate*, or *Minor*. This option has no effect on how a problem report is handled within the workflow process.
- dPrState: The problem report state such as *Open*, *Closed*, or *Fixed*.

Example

Note: Do not confuse the Content ID (*dDocName*) with the internal content item revision identifier (*dID*). The *dID* is a generated reference to a specific rendition of a content item.

```
IdcService=ADD_PROBLEMREPORT
dPrCaption=Problem report for this content.
dDocName=PublicDoc1_ia3c488971
dID=67
dPrState=OPEN
dPrSeverity=MINOR
```

DELETE_PROBLEMREPORT

Service that deletes an existing problem report. The most likely errors are mismatched parameters or when the content item no longer exists in the system.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report ID.

Example

```
IdcService=DELETE_PROBLEMREPORT
dPrID=1
```

GET_CRITERIA_WORKFLOWS_FOR_GROUP

Service that returns a list of Criteria workflows and workflow steps for a specific security group.

Returns the WorkflowsForGroup and WorkflowStepsForGroup ResultSets:

- WorkflowsForGroup lists all of the workflows for this group (*dWfID*, *dWfName*).
- WorkflowStepsForGroup lists all of the steps in all of the workflows for this group (*dWfID*, *dWfName*, *dWfStepID*, *dWfStepName*).

The most likely error is a security group that does not exist or a user failing the security check.

Service class: Service (general service).

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dSecurityGroup: The security group such as *Public* or *Secure*.

Example

IdcCommand command file format:

```
# Retrieves criteria workflow information
IdcService=GET_CRITERIA_WORKFLOWS_FOR_GROUP
dSecurityGroup=Public
```

HDA format:

```
@Properties LocalData
IdcService=GET_CRITERIA_WORKFLOWS_FOR_GROUP
dSecurityGroup=Public
@end
```

GET_DOCUMENT_PROBLEMREPORTS

Service that returns all problem reports for a specific content item. This service is available from the content information page and is only available when documents are being staged by Content Publisher.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
- dProjectID: The Content Publisher project ID generated by the content server.

Example

```
IdcService=GET_DOCUMENT_PROBLEMREPORTS
dProjectID=servG0SGXDW5YD55T80NHPR57MWW40
dDocName=security_000015
```

GET_PROBLEMREPORTS_SEARCH_FORM

Service that returns the problem report search form.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

GET_PROBLEMREPORTS_SEARCH_RESULTS

Service that returns a list of problem reports that match search criteria as specified on the Problem Reports search page. The search criteria can be any of the columns as specified in the Problem Reports table.

Search criteria that are used can take any of the columns specified in the Problem Reports table. The service is passed in name/value pairs, which are then turned into a database query's `WHERE` clause.

The Problem Reports data source (as specified in the resource.htm file) is used with the WHERE clause generated by the name/value pair to construct the query. The results of this query are then passed back for presentation.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Optional Service Parameters

- dPrSeverity: The problem report severity level such as *Critical*, *Moderate*, *Minor*. This option has no bearing on how a problem report is handled within the workflow process.
- dPrAuthor: The problem report author.
- dPrCaption: The caption for the problem report.
- dPrState: The problem report state such as *Open*, *Closed*, *Fixed*.
- dDocTitle: The content item title.
- dDocName: The Content ID for the content item.
- dPrjDescription: The project description.

GET_UPDATE_PROBLEMREPORT_FORM

Service that returns the update form for a problem report.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report ID.

Example

```
IdcService=GET_UPDATE_PROBLEMREPORT_FORM
dPrID=1
```

GET_WORKFLOWDOCUMENTS

Service that returns a list of content item revisions that are in a specific workflow.

The service provides a list of all content items in workflows and is updated by the system server to keep track of the status of content items (state and step) that are in workflows.

The most likely error is a workflow name that does not exist.

Service class: Service (general service)

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- IdcService: Must be set to GET_WORKFLOWDOCUMENTS.
- dWfName: The workflow name.

Example

```
IdcService=GET_WORKFLOWDOCUMENTS
dWfName=mktg_review
```

GET_WORKFLOWS_FOR_ALL

Service used by the Workflow applet to return information about all workflows and their steps.

Service Class: Service (general service)

Location: <*install_dir*>/shared/config/resources/workflow.htm

LOAD_WORKFLOW_QUEUE

Service that returns the personalization data for the user, which contains a list of content items in a workflow that require action. The data comes from the wf_in_queue.hda file in the <*install_dir*>/data/users/profiles directories.

This is intended to be a remote applicaiton service because it is a raw data service with no associated display template.

Service Class: User Service

Access Level: Read, Scriptable (33)

Location: <*install_dir*>/shared/config/resources/std_services.htm

Results

- Results Sets: WorkflowInQueue

Used By

- Other: SoapCustom:Wsdl:Workflow:Services

NOTIFY_CONTRIBUTOR

Service that notifies a contributor on the problem report for the specified content item. The most likely error is a content item no longer in the system.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

```
IdcService=NOTIFY_CONTRIBUTOR
dID=55
```

PROBLEMREPORT_INFO

Service that returns all information about a problem report. The information for a problem report is in the file system and in the database. This service takes the user (when used in context) to the problem report information page. This allows the user to resend, delete, or update the problem report from the user interface.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report's unique ID.

Example

```
IdcService=PROBLEMREPORT_INFO
dPrID=1
```

RESEND_PROBLEMREPORT

Service that resends email to all interested users that are to be notified about an activity. This service is accessible from the Problem Report page.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report ID.

Example

```
IdcService=RESEND_PROBLEMREPORT
dPrID=1
```

REVIEW_WORKFLOW_DOC

Service that generates the page allowing a user to review a document in a workflow.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

UPDATE_PROBLEMREPORT

Service that updates problem report information.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report ID.
- dPrState: The problem report state, such as *Open*, *Closed*, *Fixed*.

Optional Service Parameters

- dPrCaption: The caption for the problem report.
- dPrSeverity: The problem report severity level, such as *Critical*, *Moderate*, *Minor*. This option has no effect on how a problem report is handled within the workflow process.
- prMessage: The problem report message.

Example

❖ IdcCommand command file format:

```
IdcService=UPDATE_PROBLEMREPORT
dPrID=1
dPrState=OPEN
```

❖ HDA file format with optional parameters:

```
@Properties LocalData
IdcService=UPDATE_PROBLEMREPORT
dPrID=1
dPrState=OPEN
prMessage=Description is stored in hda file.
dPrCaption=change this caption
dPrSeverity=SERIOUS
@end
```

WORKFLOW_CHECKIN_SUB

Subservice that checks in a content item if the item already exists and is in a workflow.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

See Also

– *CHECKIN_UNIVERSAL (page 5-48)*

– *WORKFLOW_EDIT_REV (page 6-11)*

WORKFLOW_EDIT_REV

Subservice used by the WORKFLOW_CHECKIN_SUB subservice to determine what kind of workflow check is being performed. This service edits the workflow revision in place. When the workflow design is evaluated, a decision is made whether to use WORKFLOW_EDIT_REV or WORKFLOW_NEW_REV.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

See Also

– *WORKFLOW_CHECKIN_SUB (page 6-11)*

– *WORKFLOW_NEW_REV (page 6-12)*

WORKFLOW_NEW_REV

Subservice used by the WORKFLOW_CHECKIN service to determine the type of workflow checkin being performed. This service creates a new revision for the workflow document.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

See Also

– *WORKFLOW_CHECKIN (page 6-45)*

WORKFLOW_REJECT_FORM

Service that returns the reject form for a rejected workflow revision. This service is executed when a reviewer rejects content in a workflow step (when the reviewer clicks Reject).

This service is normally performed in the browser environment, which uses JavaScript to assign values. Parameters that are passed in the browser environment are listed below.

Service Class: Doc Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.
- dWfName: The workflow name.

Example

```
IdcService=WORKFLOW_REJECT_FORM
dWfName=mktg_review
dID=44
```

WORKFLOW TEMPLATE SERVICES

Workflow Template Services are used to manage or alter workflow templates. All services listed here have a Service Class of Workflow Template Service. The following services are described in this section:

- ADD_WF_TEMPLATE (page 6-13)
- DELETE_WF_TEMPLATE (page 6-13)
- EDIT_WF_TEMPLATE (page 6-14)
- GET_WF_TEMPLATE (page 6-20)
- GET_WF_TEMPLATES (page 6-20)

ADD_WF_TEMPLATE

Service that creates a new workflow template. The most likely error is when the workflow template name is not unique or when the system is unable to retrieve the workflow template list.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfTemplateName: The workflow template name.
- dWfTemplateDescription: The workflow template description.

Example

```
IdcService=ADD_WF_TEMPLATE
dWfTemplateName=mark_temp
dWfTemplateDescription=marketing template
```

DELETE_WF_TEMPLATE

Service that deletes an existing workflow template. The most likely error is when the system is unable to retrieve workflow templates.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfTemplateName: The workflow template name.

Example

```
IdcService=DELETE_WF_TEMPLATE
dWfTemplateName=wf_template_10
```

EDIT_WF_TEMPLATE

Service that modifies an existing workflow template. This service is similar to EDIT_WORKFLOWSTEP, as a workflow template has steps in it. However, this service allows the editing of more than one step and other parts of the template.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dAliases: The list of alias users separated by the */n* escape sequence.
- For example: `dAliases=subadmins/nservices`
- dWfStepDescription: The step description.
- dWfStepHasWeight: Enables the limited reviewer option.
  - Set to 1 (*true*), and define the number of reviewers with the *dWfStepWeight* parameter.
  - Set to 0 (*false*) if *dWfStepIsAll* is enabled.
- dWfStepIsAll: Enables the all reviewer option.
  - Set to 1 (*true*) to require all users assigned to the step to approve the revision before the workflow passes to the next step.
  - Set to 0 (*false*) if *dWfStepHasWeight* is enabled.
- dWfStepName: The workflow step name. Typically this is the type of review (such as *initial review* or *copy edit*) or the function of the reviewer (such as *manager* or *copy editor*).
- dWfStepType: The workflow step type:
  - Reviewer—Approves or rejects the revision.
  - Reviewer/Contributor—Can edit the revision and approves or rejects it.
- dWfStepWeight: Defines the number of reviewers for the limited reviewer option.

- Enter a numeric value for the number of reviewers. The workflow passes to the next step as soon as the number of users specified have approved the revision.
- You can enter zero (0) in this field to notify reviewers that the revision has reached the step, but reviewers will not be able to approve, reject, or edit the revision at that step. The workflow will pass to the next step automatically.

❖ dWfTemplateDescription: The workflow template description.

❖ dWfTemplateName: The workflow template name.

Example

The following is an example EDIT_WF_TEMPLATE service call:

```
@Properties LocalData
dUser=fradiche
dWfStepIsAll=0
watchedMonikers=
blFieldTypes=StatusMessage message
monitoredTopics=appcommongui,1060430685890
dAliases=fradiche*09user*09user1*09user
monitoredSubjects=collaborations,1061386025875,config,1061386026266,metadata,10613
86026216,metaoptlists,1061386026236,wftemplates,1061386026267,projects,10613860265
86,users,1061386026216,workflows,1061386026296,aliases,1061386026266,wfscripts,106
1386026266,usermetaoptlists,1061386026256,doctypes,1061386026216,accounts,10613860
26216
refreshSubMonikers=
dWfStepID=
dWfStepHasWeight=1
wfExitScript=
wfUpdateScript=
ClientEncoding=Cp1252
dWfStepDescription=
wfUpdateScriptSummary=
dWfStepType=Reviewer
dWfID=
HasAdditionalExitCondition=0
IsTemplateScript=1
dWfStepName=s1
IsJava=1
wfExitConditionSummary=
```

```
dAliasType=user
ConditionKeys=HasAdditionalExitCondition,wfAdditionalExitCondition
dWfTemplateDescription=test
dWfTemplateName=test
blDateFormat=M/d{/yyyy} {h:mm{:ss}{.SSSS} {aa}}!tAmerica/Chicago
IdcService=EDIT_WF_TEMPLATE
dAlias=user1
dWfStepWeight=1
wfEntryScriptSummary=
wfExitScriptSummary=
wfEntryScript=
@end
@ResultSet WorkflowSteps
9
dWfStepName
dWfStepID
dWfID
dWfStepType
dWfStepIsAll
dWfStepWeight
dWfStepDescription
dAliases
dWfStepHasWeight
s1

Reviewer
0
1

fradiche*09user*09user1*09user
1
@end
@ResultSet WorkflowStepEvents
4
dWfStepName
wfEntryScript
wfExitScript
```

```
wfUpdateScript
s1

@end
@ResultSet UpdatedUserTopics
3
topicName
topicValue
topicTS
@end
```

The following is an example of the results that would be returned:

```
Content-type: text/plain

Content-Length: 2446

<?hda version="6.3 dev (build-date)" jcharset=Cp1252 encoding=iso-8859-1?>

@Properties LocalData
wfUpdateScript=
dUser=sysadmin
refreshSubMonikers=
refreshTopics=appcommongui,1040486396000
dWfTemplateDescription=test
dAlias=user1
wfExitConditionSummary=
monitoredSubjects=collaborations,1061386025875,config,1061386026266,metadata,10613
86026216,metaoptlists,1061386026236,wftemplates,1061386026267,projects,10613860265
86,users,1061386026216,workflows,1061386026296,aliases,1061386026266,wfscripts,106
1386026266,usermetaoptlists,1061386026256,doctypes,1061386026216,accounts,10613860
26216
wfExitScript=
ConditionKeys=HasAdditionalExitCondition,wfAdditionalExitCondition
dWfStepDescription=
dAliases=fradiche*09user*09user1*09user
dWfStepType=Reviewer
watchedMonikers=
HasAdditionalExitCondition=0
```

```
changedMonikers=
changedSubjects=
blFieldTypes=StatusMessage message
IdcService=EDIT_WF_TEMPLATE
wfExitScriptSummary=
dWfStepHasWeight=1
wfEntryScript=
wfUpdateScriptSummary=
dAliasType=user
dWfStepName=s1
dWfStepWeight=1
wfEntryScriptSummary=
dWfStepID=
refreshMonikers=
blDateFormat=M/d{/yyyy} {h:mm{:ss}{.SSSS} {aa}}!tAmerica/Chicago
ClientEncoding=Cp1252
IsJava=1
dWfStepIsAll=0
IsTemplateScript=1
refreshSubjects=wftemplates,1061386026272
dWfTemplateName=test
monitoredTopics=appcommongui,1060430685890
dWfID=
@end
@ResultSet WorkflowSteps
9
dWfStepName
dWfStepID
dWfID
dWfStepType
dWfStepIsAll
dWfStepWeight
dWfStepDescription
dAliases
dWfStepHasWeight
s1
```

```
Reviewer
0
1

fradiche*09user*09user1*09user
1
@end
@ResultSet WorkflowStepEvents
4
dWfStepName
wfEntryScript
wfExitScript
wfUpdateScript
s1

@end
@ResultSet UpdatedUserTopics
3
topicName
topicValue
topicTS
appcommongui
\@Properties LocalData\nblFieldTypes=\nblDateFormat=M/d{/yy} {h:mm{:ss.SSSS}
{aa}[zzz]}!tAmerica/Chicago\n\@end\n\@ResultSet
UserSelectView:filter\n3\nfilterColumn\nfilterValue\nisEnabled\ndUserAuthType\nLOC
AL\ntrue\n\@end\n\@ResultSet
UserView:filter\n3\nfilterColumn\nfilterValue\nisEnabled\ndUserAuthType\nLOCAL\ntr
ue\n\@end\n\@ResultSet
DocView:filter\n3\nfilterColumn\nfilterValue\nisEnabled\ndProcessingState\nM\nfals
e\ndIsCheckedOut\n1\nfalse\ndStatus\nGENWWW\ntrue\ndDocTitle\nCanceled:%\nfalse\nd
ReleaseState\nN\nfalse\nisLatestRev\n\ntrue\ndSecurityGroup\ntest\nfalse\ndDocType
\ntest\nfalse\n\@end\n\n
1040486396000
@end
@ResultSet WfTemplates
2
dWfTemplateName
dWfTemplateDescription
collaboration_template
```

```
Basic Collaboration template
test
test
@end
```

GET_WF_TEMPLATE

Service that returns the description of a specific workflow template. The most likely error is a workflow template name that does not exist.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfTemplateName: The workflow template name.

Example

```
IdcService=GET_WF_TEMPLATE
dWfTemplateName=servicestemplate
```

See Also

– *GET_WF_TEMPLATES (page 6-20)*

GET_WF_TEMPLATES

Service that returns a list of all workflow templates and their descriptions. The most likely error is a workflow name that does not exist.

Service Class: Workflow Template Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

WORKFLOW SERVICES

Workflow services are those used to manage workflows, including adding a workflow, adding tokens, and enabling or disabling a workflow. Frequently used services are marked with an asterisk (*) in the following list.

All of these services have a Service Class of Workflow Service.

The following services are described in this section:

- ADD_WORKFLOW (page 6-22)
- ADD_WORKFLOW_SCRIPT (page 6-23)
- ADD_WORKFLOW_TOKEN (page 6-23)
- ADD_WORKFLOWALIASES (page 6-24)
- ADD_WORKFLOWDOCUMENT (page 6-24)
- ADD_WORKFLOWDOCUMENT_SUB (page 6-25)
- ADD_WORKFLOWDOCUMENTS (page 6-25)
- ADD_WORKFLOWSTEP (page 6-26)
- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)
- CRITERIAWORKFLOW_ENABLE (page 6-28)
- DELETE_WFCONTRIBUTORS (page 6-28)
- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOW_SCRIPT (page 6-29)
- DELETE_WORKFLOW_TOKEN (page 6-30)
- DELETE_WORKFLOWCRITERIA (page 6-30)
- DELETE_WORKFLOWDOCUMENTS (page 6-30)
- DELETE_WORKFLOWSTEP (page 6-32)
- EDIT_WORKFLOW (page 6-32)
- EDIT_WORKFLOW_SCRIPT (page 6-33)
- EDIT_WORKFLOW_TOKEN (page 6-34)
- EDIT_WORKFLOWCRITERIA (page 6-34)
- EDIT_WORKFLOWSTEP (page 6-35)
- *GET_ACTIVE_WORKFLOWS (page 6-37)
- GET_ALL_WORKFLOWDOCREVISIONS (page 6-37)
- GET_CRITERIA_WORKFLOWS_FOR_GROUP (page 6-38)
- GET_WF_COMPANION_INFO (page 6-38)

- GET_WORKFLOW (page 6-39)
- *GET_WORKFLOW_INFO (page 6-39)
- *GET_WORKFLOW_INFO_BYNAME (page 6-40)
- GET_WORKFLOW_SCRIPT (page 6-42)
- *GET_WORKFLOWDOCREVISIONS (page 6-42)
- GET_WORKFLOW_INFO (page 6-39)
- GET_WORKFLOWS (page 6-44)
- TEST_WORKFLOW_SCRIPT (page 6-44)
- WORKFLOW_APPROVE (page 6-45)
- WORKFLOW_CANCEL (page 6-45)
- *WORKFLOW_CHECKIN (page 6-45)
- WORKFLOW_EDIT_APPROVE (page 6-50)
- *WORKFLOW_REJECT (page 6-50)
- WORKFLOW_START (page 6-53)

ADD_WORKFLOW

Service that creates a new workflow. This service is executed by the Workflow Admin applet. Workflows and sub-workflows can be added, edited, enabled, disabled, and deleted from the Workflow Admin administration applet.

The most likely error is when the workflow name is not unique.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dWfDescription: The workflow description.
- dWfType: The workflow type such as *Basic* or *Criteria*.
- dSecurityGroup: The security group such as *Public* or *Secure*.
- dWfCriteriaName: The workflow criteria field.
- dWfCriteriaOperator: The workflow criteria operator *matches*.

- dWfCriteriaValue: The workflow criteria value.

Example

```
IdcService=ADD_WORKFLOW
dWfName=test_workflow
dSecurityGroup=Public
dWfType=Criteria
dWfCriteriaName=dDocAuthor
dWfCriteriaOperator=matches
dWfCriteriaValue=sysadmin
dWfDescription=testing workflow
```

ADD_WORKFLOW_SCRIPT

Service used to add a script template for a workflow.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- wfScriptName: Name of the script template to be used.
- wfScriptDescription: Description used for the script.

ADD_WORKFLOW_TOKEN

Service used to add tokens for workflows.

Service Class: Workflow Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- wfToken: Default token to use to create the new token.
- wfTokenName: Name of the new token to be added.
- wfTokenDescription: Description used for the token.

ADD_WORKFLOWALIASES

Service that adds a user or an alias to a workflow step. To add a user, set the dAliasType parameter to *user*. To add an alias, set dAliasType to *alias*. The most likely error is when the specified workflow does not exist.

Note: The alias is not defined here. The alias must already exist. It is usually created using the UserAdmin applet.

Service Class: Workflow Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- aliases: The alias name.
- dAliasType: The assigned alias type. Values can be *alias* or *user*.
- dWfID: The workflow ID.
- dWfName: The workflow name.
- dWfStepID: The workflow contribution stage ID. This auto-generated value can be retrieved from the database table. The content server automatically assigns *dWfStepID* for that contribution stage.

Example

```
IdcService=ADD_WORKFLOWALIASES
dWfName=mktg_review
dWfID=12
dAliasType=user
aliases=sammy
dWfStepID=26
```

ADD_WORKFLOWDOCUMENT

Service that adds a new content item to a Basic workflow. The content item specified by dDocName does not have to exist in the system.

This service is called from the Workflow Admin applet by creating a Basic workflow and clicking **New** in the Content section. This service executes the ADD_WORKFLOWDOCUMENT_SUB subservice.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dDocName: The Content ID of the content item.

Example

```
IdcService=ADD_WORKFLOWDOCUMENT
dWfName=mktg_review
dDocName=test1234
```

See Also

– *ADD_WORKFLOWDOCUMENT_SUB (page 6-25)*

ADD_WORKFLOWDOCUMENT_SUB

Subservice called by ADD_WORKFLOWDOCUMENTS and ADD_WORKFLOWDOCUMENT to add a content item to the Basic workflow.

Location: <*install_dir*>/shared/config/resources/workflow.htm

See Also

– *ADD_WORKFLOWDOCUMENT (page 6-24)*

– *ADD_WORKFLOWDOCUMENTS (page 6-25)*

ADD_WORKFLOWDOCUMENTS

Service that adds the selected content items to a Basic workflow. The list of items to add is specified in the dDocName parameter.

This service is called from the Workflow Admin applet by creating a Basic workflow and clicking **Select** in the Content section.

This service executes the subservice ADD_WORKFLOWDOCUMENT_SUB.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dDocName: The Content ID of the content items to be added. If multiple items are to be added, use a tab-separated list.

Example

```
IdcService=ADD_WORKFLOWDOCUMENTS
dWfName=mktg_review
dDocName=test_000035
dataSource=Documents
resultName=DOCUMENTS
dWfType=Basic
whereClause=dSubscriptionType='test'
dWfDirectory=public
```

ADD_WORKFLOWSTEP

Service that creates a new workflow step.

- These parameters can be executed in a browser environment using the Workflow Admin applet. For example, the *dWfStepName*, *dWfStepDescription*, and *dWfStepType* entries correspond to the *Name* entry, the *Description* field, and the *Type* drop-down list of the Workflow Admin applet Edit Step Reviewer screen.
- Setting *dWfStepWeight* to a numeric value is the same as entering a value in the field associated with the "At least this many reviewers" option for Workflow Steps in the Workflow Admin applet. If defined, the workflow passes to the next step as soon as the number of users specified in *dWfStepWeight* have approved the revision.
- Setting *dWfStepIsAll* to 1 (*true*) is the same as enabling the "All Reviewers" option for Workflow Steps in the Workflow Admin applet. If set to *true*, all users assigned to the step must approve the revision before the workflow passes to the next step.
- The most likely error is when the specified workflow does not exist or when the specified step is not unique.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dWfStepName: The workflow step name.
- dWfStepDescription: The step description.
- dWfStepType: The workflow step type:
  - **Reviewer—**Approves or rejects the revision.
  - **Reviewer/Contributor—**Can edit the revision and approves or rejects it.
  - dWfStepIsAll:
  - 1 (*true*)—All users assigned to the step must approve the revision before the workflow passes to the next step.
  - 0 (*false*)—The number of approvals required is specified by the *dWfStepWeight* parameter.
- dWfStepWeight: The number of reviewers that must approve the revision before the workflow passes to the next step.
  - If *dWfStepIsAll* is 1 (*true*), this parameter is ignored. If *dWfStepIsAll* is 0 (*false*), this parameter is enabled.
  - Setting this parameter to **zero (0)** will notify reviewers that the revision has reached the step, but reviewers will not be able to approve, reject, or edit the revision at that step. The workflow will pass to the next step automatically.

Optional Service Parameters

- wfEntryScript: The step entry script.
- wfExitScript: The step exit script.
- wfUpdateScript: The step update script.

Note: Scripts must be placed within `<$` and `$>` delimiters. For example, `<$if ConditionExpression$>`.

CRITERIAWORKFLOW_DISABLE

Service that disables a Criteria workflow. The most likely error is when the specified workflow does not exist.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=CRITERIAWORKFLOW_DISABLE
dWfName=mktg_review
```

CRITERIAWORKFLOW_DISABLE_SUB

Service that disables a sub-workflow in a Criteria workflow.

Location: <*install_dir*>/shared/config/resources/workflow.htm

CRITERIAWORKFLOW_ENABLE

Service that enables a Criteria workflow. The most likely error is when the specified workflow does not exist.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

dWfName: The workflow name.

Example

```
IdcService=CRITERIAWORKFLOW_ENABLE
dWfName=mktg_review
```

DELETE_WFCONTRIBUTORS

Service used to delete the users and aliases from the first step (the contribution step) of a Basic workflow. The most likely errors are mismatched parameters or when the specified workflow or alias does not exist.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- aliases: The aliases and users to be deleted from the Basic workflow. If multiple aliases are to be deleted, use a tab-separated list.

- dWfName: The workflow name.
- dWfStepID: The workflow contribution step ID. This auto-generated value can be retrieved from the database table.

Example

```
IdcService=DELETE_WFCONTRIBUTORS
dWfName=mktg_review
dWfStepID=26
aliases=user20
```

DELETE_WORKFLOW

Service that deletes an existing Basic workflow. The most likely error is when the specified workflow does not exist.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=DELETE_WORKFLOW
dWfName=mktg_review
```

DELETE_WORKFLOW_SCRIPT

Service that deletes a workflow script.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- wfScriptName: Name of the script template to be used.
- wfScriptDescription: Description used for the script.

DELETE_WORKFLOW_TOKEN

Service that deletes a workflow token.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- wfTokenName: Name of the token to be deleted.

DELETE_WORKFLOWCRITERIA

Service that deletes an existing Criteria workflow. If the workflow is active the user is prompted that this action will cause all content items in this workflow to exit the workflow. The most likely error is when the specified workflow does not exist.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=DELETE_WORKFLOWCRITERIA
dWfName=mktg_review
```

DELETE_WORKFLOWDOCUMENTS

Service that deletes content items from a Basic workflow. This service is executed by the Workflow Admin applet.

The most likely error is when the specified workflow does not exist.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The Basic workflow name.
- docNames: The content item names of the documents to be deleted. If multiple items are to be deleted, use a tab-separated list.

Note: It is recommended that you include both the *docNames* parameter and the *dDocName* parameter, as both parameters are used in the execution of this service.

Example

The following is an example DELETE_WORKFLOWDOCUMENTS service call:

```
@Properties LocalData
IdcService=DELETE_WORKFLOWDOCUMENTS
dWfName=w3
docNames=rr
dDocName=rr
@end
```

The following is an example of the results that would be returned:

```
Content-type: text/plain

Content-Length: 931

<?hda version="6.3 dev (build-date)" jcharset=Cp1252 encoding=iso-8859-1?>

@Properties LocalData
dUser=sysadmin
blFieldTypes=xThreadParentDocName bigtext,xCollectionID int,xMailType
text,xZoomLevel int,dCompletionDate date,xTargetCompression int,xTemplateType
text,xEmailFrom bigtext,xEmailCC memo,dMessage message2,xHidden
text,xClbraAliasList memo,xWebsiteID int,StatusMessage message,xComments
memo,xEmailDate date,xWebsiteObjectType text,xCollectionInhibitUpdateMeta
text,xClbraUserList memo,xFileFormat bigtext,dReleaseDate date,xMessageID
text,dInDate date,xDiscussionType text,dCreateDate date,dOutDate
date,xDiscussionCount int,xReadOnly text
docNames=rr
refreshSubMonikers=
blDateFormat=M/d{/yyyy} {h:mm{:ss}{.SSSS} {aa}}!tAmerica/Chicago
dWfName=w3
refreshSubjects=
changedSubjects=workflows,1061386026294
dDocName=rr
refreshMonikers=
```

```
changedMonikers=
dWfDocState=INIT
IdcService=DELETE_WORKFLOWDOCUMENTS
IsJava=1
@end
```

DELETE_WORKFLOWSTEP

Service that deletes a specified step from a workflow. This service is used from the Workflow applet. The user selects a step and clicks the **Delete Step** button. If this service is run outside the applet, then the controlling application must specify the workflow step. The step ID and name is stored in the database as part of the workflow design information and can be retrieved from there for use.

The most likely error is when the specified workflow or workflow step does not exist or if the workflow is active.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dWfStepName: The workflow step name.
- dWfStepID: The workflow contribution stage ID.

Example

```
IdcService=DELETE_WORKFLOWSTEP
dWfName=c2
dWfStepID=12
dWfStepName=edit
```

EDIT_WORKFLOW

Service that modifies an existing workflow. This service is executed by the Workflow Admin applet. The most likely error is a workflow name that does not exist or a user failing the security check.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dSecurityGroup: The security group such as *Public* or *Secure*.

Optional Service Parameters

- dWfCriteriaName: The workflow criteria name.
- dWfCriteriaOperator: The workflow criteria operator such as *Matches*, *Starts*, *Ends*.
- dWfCriteriaValue: The workflow criteria value.
- dWfDescription: The workflow description.
- dWfType: The workflow type such as *Basic* or *Criteria*.

Example

- IdcCommand command file format:

```
IdcService=EDIT_WORKFLOW
dWfName=c2
dSecurityGroup=Public
```

- HDA format with optional parameters:

```
@Properties LocalData
IdcService=EDIT_WORKFLOW
dWfName=test_workflow
dSecurityGroup=Public
dWfType=Criteria
dWfCriteriaName=dDocAuthor
dWfCriteriaOperator=matches
dWfCriteriaValue=sysadmin
dWfDescription=testing workflow
@end
```

EDIT_WORKFLOW_SCRIPT

Service used to edit a workflow script.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- wfScriptName: Name of the script template to be used.
- wfScriptDescription: Description used for the script.

EDIT_WORKFLOW_TOKEN

Service used to edit a workflow token.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- wfTokenName: Name of the token to be edited.

EDIT_WORKFLOWCRITERIA

Service that modifies the criteria for an existing Criteria workflow. Editing the criteria may include changing the workflow to a project or collaboration workflow, or changing the metadata conditions that a content item needs to meet in order to enter the workflow.

Two types of Criteria workflows exist: *criteria*, which is a workflow that requires a metadata condition and *subworkflow*, which does not have a metadata condition. Because of this, dWfType is a required parameter to designate which type of Criteria workflow is to be used.

This service is executed by the Workflow Admin applet. The most likely error is a workflow name that does not exist or a user failing the security check.

Service Class: Workflow Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dSecurityGroup: The security group such as *Public* or *Secure*.
- dWfType: The workflow type.
- dWfCriteriaName: The workflow criteria name.
- dWfCriteriaOperator: The workflow criteria operator such as *Matches*, *Starts*, *Ends*.
- dWfCriteriaValue: The workflow criteria value.

Optional Service Parameters

- dWfDescription: The workflow description.

Example

```
IdcService=EDIT_WORKFLOWCRITERIA
dWfName=c2
dSecurityGroup=Public
dWfType=Criteria
dWfCriteriaName=dDocType
dWfCriteriaOperator=matches
dWfCriteriaValue=ADACCT
```

EDIT_WORKFLOWSTEP

Service that modifies an existing workflow step.

These parameters can be executed in a browser environment using the Workflow Admin applet. For example, the *dWfStepName*, *dWfStepDescription*, and *dWfStepType* entries correspond to the *Name* entry, the *Description* field, and the *Type* drop-down list of the Workflow Admin applet Edit Step Reviewer screen.

Setting *dWfStepWeight* to a numeric value is the same as entering a value in the field associated with the "At least this many reviewers" option for Workflow Steps in the Workflow Admin applet. If defined, the workflow passes to the next step as soon as the number of users specified in *dWfStepWeight* have approved the revision.

Setting *dWfStepIsAll* to 1 (*true*) is the same as enabling the "All Reviewers" option for Workflow Steps in the Workflow Admin applet. If set to *true*, all users assigned to the step must approve the revision before the workflow passes to the next step.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dWfStepName: The workflow step name.
- dAliases: A tab-delimited list of aliases and users to be used for the step. The format for this parameter is (*alias/user)\t(alias/user type)\t*(*alias/user)\t(alias/user type)*.

- dWfStepDescription: The step description.
- dWfStepType: The workflow step type:
  - **Reviewer—**Approves or rejects the revision.
  - **Reviewer/Contributor—**Can edit the revision and approves or rejects it.
- dWfStepIsAll: 1 (*true*)—All users assigned to the step must approve the revision before the workflow passes to the next step. 0 (*false*)—The number of approvals required is specified by the *dWfStepWeight* parameter.
- dWfStepWeight: The number of reviewers that must approve the revision before the workflow passes to the next step.
  - If *dWfStepIsAll* is 1 (*true*), this parameter is ignored. If *dWfStepIsAll* is 0 (*false*), this parameter is enabled.
  - Setting this parameter to zero (0) will notify reviewers that the revision has reached the step, but reviewers will not be able to approve, reject, or edit the revision at that step. The workflow will pass to the next step automatically.

Optional Service Parameters

- wfEntryScript: The step entry script.
- wfExitScript: The step exit script.
- wfUpdateScript: The step update script.

Note: Scripts must be placed within `<$` and `$>` delimiters. For example, `<$if ConditionExpression$>`.

Example

Note in the following example, the convention `\t` denotes a tab separator.

```
IdcService=EDIT_WORKFLOWSTEP
dWfName=test_workflow
dWfStepName=edit
dWfStepType=Reviewer/Contributor
dWfStepDescription=copy edit step
dAliases=AllUsers\taliases\tsysadmin\tuser
dWfStepIsAll=0
dWfStepWeight=2
wfEntryScript=
```

```
wfExitScript=
wfUpdateScript=
```

GET_ACTIVE_WORKFLOWS

Service that returns the Active Workflows page where a user can select a workflow and view all documents in a workflow.

Access Level: Read, Global, Scriptable (49)

Location: <*install_dir*>/shared/config/resources/workflow.htm

Results

- Result Sets:
  - StdWorkflows (dWfName, dWfID, dWfDescription, dCompletionDate, dSecurityGroup, dWfStatus, dWfType, dProjectID, dIsCollaboration)
  - ClbraWorkflows (same fields as StdWorkflows)
  - ClbraAccessList (dClbraName, userList, aliasList)
- Response Template: WORKFLOW_LIST (workflow_list.htm)

Used By

- Resource Includes: pne_nav_management_links, std_doc_man_pages
- Templates:
  - WORKFLOW_REJECT_FORM (reject_doc.htm)
  - ALL_WORKFLOW_DOCS (workflow_all_docs_list.htm)
  - WORKFLOW_DOCS (workflow_docs_list.htm)
  - WORKFLOW_INFO (workflow_info.htm)
  - WF_INQUEUE_LIST (workflow_queue.htm)
  - WORKFLOW_REVIEW_FORM (workflow_review_form.htm)
- Standard Navigation: commonNav.js

GET_ALL_WORKFLOWDOCREVISIONS

Service that returns a result set of documents in a workflow and their revision information.

Location: <*install_dir*>/shared/config/resources/workflow.htm

GET_CRITERIA_WORKFLOWS_FOR_GROUP

Service used by the Workflow applet that returns workflows for a specified security group. It returns the step information for the returned workflows. The most likely error is a security group that does not exist or a user failing the security check.

Returns the *WorkflowsForGroup* and *WorkflowStepsForGroup* ResultSets:

- *WorkflowsForGroup* lists all of the workflows for this group (*dWfID*, *dWfName*).
- *WorkflowStepsForGroup* lists all of the steps in all of the workflows for this group (*dWfID*, *dWfName*, *dWfStepID*, *dWfStepName*).

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

❖ dSecurityGroup: The security group such as *Public* or *Secure*.

Example

❖ IdcCommand command file format:

```
# Retrieves criteria workflow information
IdcService=GET_CRITERIA_WORKFLOWS_FOR_GROUP
dSecurityGroup=Public
```

❖ HDA format:

```
@Properties LocalData
IdcService=GET_CRITERIA_WORKFLOWS_FOR_GROUP
dSecurityGroup=Public
@end
```

GET_WF_COMPANION_INFO

Returns the companion information file for a document in a workflow.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

❖ dDocName: The name of the document in the workflow.

❖ dWfName: The workflow name.

GET_WORKFLOW

Service that returns information about a specific workflow. The most likely error is a workflow name that does not exist.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=GET_WORKFLOW
dWfName=mktg_review
```

GET_WORKFLOW_INFO

Service that returns workflow step information for a content item. This service is executed in a browser interface by selecting Active Workflows from PNE links, clicking on the workflow name, and then clicking the link for step name.

The most likely error is a content item or workflow name that does not exist.

Access Level: Read, Global, Scriptable (49)

Queries Executed: QdocInfo, QworkflowDocument, QworkflowForID, QwfDocState, QworkflowSteps

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.
- dWfStepID: The workflow contribution stage ID. This auto-generated value can be retrieved from the database table.

Results

- Result Sets:
  - WorkflowSteps (All fields from all rows of WorkflowSteps DB table for specified workflow)

- WorkflowStep (All WorkflowSteps fields for current step plus dUsers and dHasTokens fields)
- DOC_INFO (All fields from Revisions and DocMeta for the specified revision)
- WorkflowInfo (All fields from Workflows DB table for specified workflow)
- WorkflowState (dUserName field only from WorkflowState DB table for specified content item)
- WorkflowActionHistory (Result set from corresponding HDA file in data\workflow\states directory)

❖ Local Data:

- AuthorAddress
- dID
- dUser
- dWfCurrentStepID
- dWfName
- dWfStepID
- RemainingStepUsers

❖ Response Template: WORKFLOW_INFO (workflow_info.htm)

Used By

❖ Resource Includes: clbra_wf_doc_list

Example

```
IdcService=GET_WORKFLOW_INFO
dID=61
dWfStepID=19
```

GET_WORKFLOW_INFO_BYNAME

Service used to retrieve information about a workflow based on the name of a document in the workflow. It returns the Workflow Information page for a content item, which lists the history of the document, what steps remain in the workflow, and who is currently reviewing the workflow.

Access Level: Read, Global, Scriptable (49)

Queries Executed:

- QdocNameMeta
- QworkflowDocument
- QworkflowForID
- QwfDocState
- QworkflowSteps

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

❖ dDocName: The name of a document in an active workflow.

Results

❖ Result Sets:

- WorkflowSteps (All fields from all rows of WorkflowSteps DB table for specified workflow)
- WorkflowStep (All WorkflowSteps fields for current step plus dUsers and dHasTokens fields)
- DOC_INFO (All fields from Revisions and DocMeta for the specified content item)
- WorkflowInfo (All fields from Workflows DB table for specified workflow)
- WorkflowState (dUserName field only from WorkflowState DB table for specified content item)
- WorkflowActionHistory (Result set from corresponding HDA file in data\workflow\states directory)

❖ Local Data:

- AuthorAddress
- dUser
- dWfName
- dWfStepID
- RemainingStepUsers

❖ Response Template: WORKFLOW_INFO (workflow_info.htm)

Used By

- Resource Includes:
  - setup_workflow_action_popups
  - wf_in_queue_display
  - workflow_action_popup
- Templates:
  - WORKFLOW_REVIEW_FORM (workflow_review_form.htm)
  - WORKFLOW_REVIEW_FRAMES (workflow_review_frames.htm)
- Other: SoapCustom:Wsdl:Workflow:Services

GET_WORKFLOW_SCRIPT

Service used to retrieve a workflow script.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- wfScriptName: Name of the script template to be used.
- wfScriptDescription: Description used for the script.

GET_WORKFLOWDOCREVISIONS

Service that returns a list of content item revisions that are in a specific workflow. This service is run in the browser environment by selecting **Active Workflows** in PNE links and clicking on the workflow name.

The most likely error is a workflow name that does not exist.

Access Level: Read, Global, Scriptable (49)

Queries Executed:

- Qworkflow
- QwfStates
- QworkflowDocuments
- QworkflowSteps

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Optional Service Parameters

- dClbraName: Filters the results to include only items from the specified Collaboration project.

Results

- Result Sets:
  - WorkflowSteps (All fields from all rows of WorkflowSteps DB table for specified workflow plus the additional fields, dUsers and dHasTokens)
  - WfDocuments (Rows correspond to each content item belonging to the specified workflow; each row contains all fields from WorkflowDocuments, Revisions, and DocMeta DB tables)
  - WorkflowStates (All fields from the WorkflowStates DB table for the specified workflow)
  - WF_INFO (All fields from the Workflows DB table for the specified workflow)
- Local Data:
  - clbraAccount
  - dClbraName
  - dUser
  - dWfName
  - isCollaboration
- Response Template:
  - WORKFLOW_DOCS (workflow_docs_list.htm)

Used By

- Resource Includes:
  - active_collaboration_workflows_table
  - checkin_multiuploadapplet_processing_functions
  - setup_active_standard_workflows_table_row
  - legacy_active_collaboration_workflows_table

- legacy_active_standard_workflows_table
- wf_in_queue_display
- workflow_action_popup
- workflow_revisions_href

❖ Templates:

- CONTRIBUTOR_MAIL (contributor_mail.htm)
- PROJECT_INFO (project_info.htm)
- WF_REJECT_MAIL (reject_mail.htm)
- WORKFLOW_INFO (workflow_info.htm)
- WORKFLOW_REJECT_FORM (reject_doc.htm)
- WORKFLOW_REVIEW_FORM (workflow_review_form.htm)

❖ Other: SoapCustom:Wsdl:Workflow:Services

- Redirect service for: WORKFLOW_APPROVE, WORKFLOW_EDIT_APPROVE, WORKFLOW_REJECT, WORKFLOW_CHECKIN

Example

```
IdcService=GET_WORKFLOW
dWfName=mktg_review
```

GET_WORKFLOWS

Service that retrieves a list of all workflows including their description, security group, status, and type.

Location: <*install_dir*>/shared/config/resources/workflow.htm

TEST_WORKFLOW_SCRIPT

Service used to check the validity of a workflow script.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

❖ wfScriptName: Name of the script template to be used.

- wfScriptDescription: Description used for the script.
- dDocName: Content item to be used in the test scenario.

WORKFLOW_APPROVE

Service that approves a content item revision in a workflow.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

```
IdcService=WORKFLOW_APPROVE
dID=81
```

WORKFLOW_CANCEL

Service that cancels a Basic workflow.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

dWfName: The workflow name.

Example

```
IdcService=WORKFLOW_CANCEL
dWfName=mktg_review
```

WORKFLOW_CHECKIN

Service that checks in a new revision of a content item that is in a workflow. The most likely error is when the specified content item or workflow does not exist.

This service executes the subservice WORKFLOW_CHECKIN_SUB. This subservice checks a content item revision into a workflow.

Access Level: Read (1)

Queries Executed: QdociInfo, Qrevisions

Calls Subservice: WORKFLOW_CHECKIN_SUB

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dDocName: The content item identifier (Content ID).
- dID: The generated content item revision ID.
- dSecurityGroup: The security group such as *Public* or *Secure*.
- dDocAccount: The account for the content item. Required only if accounts are enabled.
- primaryFile: The absolute path to the location of the file as seen from the content server. Use the forward slash as the file separator.

  A primary file must be specified unless checking in metadata only. If an alternate file is specified with the primary file, the content refinery will convert the alternate file. Otherwise, the primary file will be converted.

  - If a primary file is not specified, a metafile can be used in its place. Only one metafile can exist though for each content item (a primary AND alternate meta file cannot co-exist).
  - If both a primary and alternate file is specified, their extensions must be different.

Optional Service Parameters

- isFinished: Indicates that the editing is finished for a rejected content item.
  1 (*true*)—The revision will be checked in and approved. |
  0 (*false*)—The revision will be checked in but will still need to be approved.
- doFileCopy: 1 (*true*)—The file will not be deleted from the hard drive after checkin.
  0 (*false*)—The file will be removed from your hard drive after checkin.
- alternateFile
- RedirectUrl

Results

- Local Data:
  - Content Server:

- alternateFile
- changedSubjects
- CurRevID
- dAction
- dActionDate
- dCheckoutUser
- dClbraName
- dConversion
- dCreateDate
- dDocAccount
- dDocAuthor
- dDocID
- dDocName
- dDocTitle
- dDocType
- dExtension
- dFileSize
- dFormat
- dID
- dInDate
- dIsPrimary
- dIsWebFormat
- dlsCheckedOut
- doFileCopy
- dLocation
- dOriginalName
- dOutDate
- dpAction
- dpEvent
- dProcessingState
- dPublishState

- ❖ dPublishType
- ❖ dRawDocID
- ❖ dReleaseState
- ❖ dRevClassID
- ❖ dRevisionID
- ❖ dRevLabel
- ❖ dRevRank
- ❖ dSecurityGroup
- ❖ dStatus
- ❖ dUser
- ❖ dUserName
- ❖ isCheckin
- ❖ isCurRevEmpty
- ❖ isDocProfileUsed
- ❖ isEditMode
- ❖ isInfoOnly
- ❖ isNew
- ❖ IsNotLatestRev
- ❖ isStatusChanged
- ❖ IsUpdate
- ❖ IsWorkflow
- ❖ latestID
- ❖ oldName
- ❖ prevID
- ❖ prevReleaseState
- ❖ primaryFile
- ❖ VaultfilePath
- ❖ WebfilePath

• Workflow:

- ❖ dWfComputed
- ❖ dWfCurrentStepID

- dWfDirectory
- dWfDocState
- dWfEntry
- dWfID
- dWfName
- dWfStatus
- dWfStepID
- dWfStepDescription
- dWfStepIsAll
- dWfStepName
- dWfStepType
- dWfStepWeight
- dWfType
- dWorkflowState
- entryCount
- wfAction
- wfCurrentStepPrefix
- WfEditFinished
- wfMessage
- wfStepCheckinType

- Response Template:
  - REDIRECT_TEMPLATE (redirect_template.htm)
  - Default redirect service (branched): GET_WORKFLOWDOCREVISIONS or GET_PORTAL_PAGE (Page=WF_INQUEUE_LIST)

Example

```
IdcService=WORKFLOW_CHECKIN
dID=92
dDocName=test_00063
dWfID=4
dSecurityGroup=Public
dDocAccount=mainaccount
```

```
dWfType=Criteria
dWfName=mktg_review
primaryFile=c:/hello_hello.txt
dRevLabel=2
doFileCopy=1
isFinished=1
```

See Also

– *CACHE_WORKFLOW_CHECKIN (page 5-25)*
– *WORKFLOW_CHECKIN_SUB (page 6-11)*

WORKFLOW_EDIT_APPROVE

Service used for client applications (that is, not used by the core Content Server) which need the ability to approve a document without passing through a checkin/checkout process.

Location: <*install_dir*>/shared/config/resources/workflow.htm

WORKFLOW_REJECT

Service that rejects a content item revision in a workflow.

The system administrator cannot approve or reject a content item in a workflow unless they are defined as a reviewer for the current step. Therefore, if you are running this service in IdcCommand as the user *sysadmin*, you cannot reject the content item unless *sysadmin* is defined as a reviewer.

Access Level: Read (1)

Queries Executed:

- QdocInfo
- IworkflowDocHistory
- UrevisionStatus
- UworkflowDocStep
- UrevisionStatus
- DworkflowDocState
- QwfDocInformation

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.
- wfRejectMessage: The rejection message.

Optional Service Parameters

- RedirectUrl

Results

- Local Data:
  - Content Server:
    - changedSubjects
    - dActionDate
    - dClbraName
    - dDocAuthor
    - dDocName
    - dDocTitle
    - dExtension
    - dID
    - dPublishState
    - dReleaseState
    - dRevClassID
    - dStatus
    - dUser
    - isCurRevEmpty
    - IsWorkflow
    - prevReleaseState
  - Workflow:
    - dAction
    - dOriginalName

- dWfCompute
- dWfCurrentStepID
- dWfDirectory
- dWfDocState
- dWfID
- dWfName
- dWfStatus
- dWfStepDescription
- dWfStepID
- dWfStepIsAll
- dWfStepName
- dWfStepType
- dWfStepWeight
- dWfType
- dWorkflowState
- entryCount
- wfAction
- wfCurrentStepPrefix
- WfEditFinished
- wfMailSubject
- wfMailTemplate
- wfMessage
- wfRejectMessage
- wfStepCheckinType
- wfUsers

- Response Template:
  - REDIRECT_TEMPLATE (redirect_template.htm)
  - Default redirect service (branched): GET_WORKFLOWDOCREVISIONS or GET_PORTAL_PAGE (Page=WF_INQUEUE_LIST)

References

- Applets: Repostory Manager
- Templates: WORKFLOW_REJECT_FORM (reject_doc.htm)
- Other: SoapCustomer:Wsdl:Workflow:Services

Example

```
IdcService=WORKFLOW_REJECT
dID=95
wfRejectMessage=Please Revise
```

WORKFLOW_START

Service Class: Workflow Service

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.
- dSecurityGroup: The security group, such as *Public* or *Secure*.

Optional Service Parameters

- wfMessage: The message that will be included in the Workflow Started notification e-mail.

Example

```
IdcService=WORKFLOW_START
dWfName=mktg_review
dSecurityGroup=Public

wfMessage=Please check in this content item
```

Chapter

7

ARCHIVING SERVICES

OVERVIEW

This chapter describes the services available when using and customizing archives. The location for specific archive services are listed within each individual service.

This chapter covers the following topics:

- About Archiving Services (page 7-1)
- Archive Services (page 7-3)

ABOUT ARCHIVING SERVICES

Note: The most commonly used services have more extensive descriptions.

A typical service entry is similar to the following:

```
GET_ARCHIVED_FILE
FileService 34 null null null !csUnableToDownloadFile
3:computeDocID::0:!csUnableToDownload(dDocName)
5:QarchivedDoc:FILE_DOC_INFO:2:!csItemNoLongerInHistory
3:checkSecurity:FILE_DOC_INFO:0:!csUnableToDownload(dDocName)
3:createArchiveFileName::0:null
```

In this example:

- the first line indicates the service name: `GET_ARCHIVED_FILE`.

- ❖ The second line lists the service's attributes:
  - Service class: `FileService`. When using the Archiver, this is frequently `ArchiveService`.
  - Access level: `34`, indicating this is Scriptable and Writeable.
  - Template: `null`
  - Service type: `null`
  - Subjects Notified: `null`
  - Error message and parameter used for error message: `Unable to Download File`
- ❖ The remainder of the entry lists the service's actions, each line designating one action:
  - action type: for example, `5`, indicating a Select Cache Query, which specifies a query to retrieve information from the database and store it for later use. This example also shows a type of `3`, indicating a Java method.
  - action name: for example, `QarchivedDoc` or `computeDocID`
  - action parameters: for example, `FILE_DOC_INFO`. If no parameters are specified, double colons (`::`) are included.
  - action control mask: for example, `0`, indicating "Ignore error" or `2`, indicating "Must exist (at least one record must be returned by the query). If no control mask is specified, double colons (`::`) or 0 are included.
  - action error message, such as `Item No Longer In History`, or null if no message is available.

Each element of the service's action is separated by a colon (`:`).

Important: All services have at least one required parameter. The IdcService parameter takes the name of the service as its argument. If other parameters are required, they are noted in the description of the service.

See Chapter 4 (*Customizing Services)* for details about a service's attributes and actions.

Important: Most archive services use the underlying method used by EDIT_ARCHIVEDATA (page 7-14) and consequently require the EditItems parameter set to the appropriate value. The user interface usually controls what is put into this parameter. If necessary, the user should exercise archive services through the user interface with Filter Debug flags enabled to determine how the values should be set. By using the Filter Debug flags, the request as it is sent to the server can be captured.

ARCHIVE SERVICES

Unless noted otherwise, all services listed here are the ArchiveService Service Class. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- ADD_ARCHIVE (page 7-5)
- ADD_COLLECTION (page 7-5)
- ADD_PROXIEDCOLLECTION (page 7-6)
- CANCEL_ARCHIVE (page 7-6)
- *CHECKIN_ARCHIVE (page 7-6)
- COPY_ARCHIVE (page 7-11)
- DELETE_ARCHIVE (page 7-11)
- DELETE_BATCH_FILE (page 7-12)
- DELETE_BATCH_FILE_DOCUMENTS (page 7-12)
- DELETE_BATCH_FILE_TABLES (page 7-13)
- EDIT_ARCHIVE (page 7-13)
- EDIT_ARCHIVEDATA (page 7-14)
- EDIT_EXPORTERS (page 7-14)
- EDIT_TRANSFEROPTIONS (page 7-15)
- EXECUTE_BATCH (page 7-15)
- EXPORT_ARCHIVE (page 7-15)
- GET_ARCHIVECOLLECTIONS (page 7-17)
- GET_ARCHIVETABLECONTENT (page 7-17)
- GET_ARCHIVED_FILE (page 7-17)
- GET_ARCHIVES (page 7-19)
- GET_ARCHIVERELATIONQUERY (page 7-19)
- GET_BATCH_FILE_DOCUMENTS (page 7-19)
- GET_BATCH_PROPERTIES (page 7-20)
- GET_BATCH_SCHEMA (page 7-20)

- GET_BATCH_VALUES (page 7-21)
- GET_BATCHFILES (page 7-21)
- GET_PROXIED_ARCHIVECOLLECTIONS (page 7-22)
- GET_PROXIEDSERVERS (page 7-22)
- GET_REPLICATION_DATA (page 7-22)
- GET_TABLECOLUMNLIST (page 7-23)
- GET_TARGET_INFO (page 7-23)
- GET_TARGET_TRANSFER_STATUS (page 7-23)
- GET_TRANSFER_SOURCE_INFO (page 7-24)
- IMPORT_ARCHIVE (page 7-24)
- IMPORT_ARCHIVE_START_AUTOMATED (page 7-26)
- IMPORT_BATCHFILE (page 7-26)
- IMPORT_DOCUMENT (page 7-26)
- IMPORT_TABLE_ENTRY (page 7-27)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)
- REGISTER_IMPORTER (page 7-28)
- REMOVE_COLLECTION (page 7-29)
- REMOVE_EXPORTER (page 7-29)
- REMOVE_IMPORTER (page 7-30)
- REMOVE_PROXIEDTRANSFER (page 7-30)
- REMOVE_QUEUED_IMPORT (page 7-30)
- REMOVE_TRANSFER (page 7-31)
- REQUEST_TRANSFER (page 7-31)
- TRANSFER_ARCHIVE (page 7-31)
- UPDATE_TARGET_TOTALS (page 7-32)
- UPDATE_TRANSFER_STATUS (page 7-32)
- UPLOAD_ARCHIVE_TRANSFER (page 7-32)

ADD_ARCHIVE

Service that adds an archive to an archive collection. The most likely errors are mismatched parameters or when an archive name is not unique.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- aArchiveDescription: The menu label for the content server instance (used on the interface). For example, *Master_on_server01*.

Example

To add an archive to a master content server, the required parameters will be:

```
IdcService=ADD_ARCHIVE
IDC_Name=Master_on_server01
aArchiveName=archive_test
aArchiveDescription=this is an archive test
```

ADD_COLLECTION

Service that creates a new archive collection. The most likely error is an instance menu label that is not unique.

Caution: Using duplicate *IDC_Name* collection names will cause data corruption. The Archiver cannot be used to move or copy data between two collections that share the same *IDC_Name*. To do so will corrupt the data on the target system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection.
- aCollectionLocation: The absolute path to the collection location. Use the forward slash as the file separator.
- aVaultDir: The absolute path to the *vault* directory. Use the forward slash as the file separator.

- aWeblayoutDir: The absolute path to the *weblayout* directory. Use the forward slash as the file separator.

Example

```
IdcService=ADD_COLLECTION
IDC_Name=new_collection
aCollectionLocation=c:/stellent/archives
aVaultDir=c:/stellent/value
aWeblayoutDir=c:/stellent/weblayout
```

ADD_PROXIEDCOLLECTION

Service used to add an archive collection from another server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- psIDC_Name: The name of the provider (proxied connection).
- IDC_Name: The name of the archive collection.

CANCEL_ARCHIVE

Service that cancels the current archive request.

Location: <*install_dir*>/shared/config/resources/std_services.htm

CHECKIN_ARCHIVE

Service that checks a content item revision into an archive.

Access Level: Admin (8)

Location: <*install_dir*>/shared/config/resources/std_services.htm

- This service executes one of the following subservices, based on parameters and conditional variables:
  - DELETE_BYREV: If the Action parameter is *Delete* and the content item revision exists in the system.
  - DELETE_BYCLASS: If the Action parameter is *Delete* and the content item revision does not exist in the system.

- CHECKIN_NEW_SUB: If the content item does not exist in the system, the Action parameter is not *Delete*, and the conditional variable *IsPublish* is set to *true*.
- INSERT_NEW: If the content item does not exist in the system, the Action parameter is not *Delete*, and the conditional variable *IsPublish* is set to *false* or not specified.
- UPDATE_DOCINFO_BYREV: If the Action parameter is not *Delete* and the *primaryFile* parameter is not empty. Additionally, the conditional variable *IsPublish* must be *true* or the content item revision must exist in the system.
- UPDATE_BYREV: If the Action parameter is not *Delete* and the *primaryFile* parameter is empty. Additionally, the conditional variable *IsPublish* must be *true* or the content item revision must exist in the system.

Important: The *IsPublish* variable is set by the server when a workflow or project is unregistered. This value cannot be set as a parameter or as a configuration entry in this service.

❖ If the *IsNative* parameter is *true*, the service executes one of the following subservices:

- DELETE_BYREV: If the content item exists in the Revisions table and action parameter is *Delete*.
- UPDATE_BYREV: If the content item exists in the Revisions table, the content has not been deleted, and the action parameter is *Update*.
- INSERT_NATIVE: If the content item does not exist in the server and the action parameter is not *Delete*.

❖ If a content item already exists in the system and has not been deleted, the server cannot update or insert the entry; an error will be thrown.

❖ The most likely error is when the content item name is not unique or when the service is unable to check in the specified file.

❖ A primary file is required. If you do not want to check in a primary file and want to check in only metadata, an additional parameter must be included and a configuration entry added in the content server.

❖ Required additional parameter (metadata checkin):

```
createPrimaryMetaFile=true
```

❖ Required content server configuration entry (metadata checkin):

```
AllowPrimaryMetaFile=true
```

Example:

```
IdcService=CHECKIN_ARCHIVE
Action=insert
dDocTitle=my_test
dDocAuthor=sysadmin
dDocType=ADACCT
dSecurityGroup=Public
createPrimaryMetaFile=true
```

Additional Required Service Parameters

- Action: Must be set to a value as specified above.
- dDocName: The Content ID for the content item.
  - If Content ID auto generation is enabled, this parameter is not required. If *dDocName* is defined, it will override the auto generated Content ID.
  - The Content ID cannot contain spaces or invalid characters ;/ \?:@&=+"#%<>*~| [ ].
- dDocAuthor: The content item author (contributor).
- dDocTitle: The content item title.
- dSecurityGroup: The security group such as *Public* or *Secure*.
- dDocAccount: The account for the content item. Required only if accounts are enabled.
- primaryFile: The absolute path to the location of the file as seen from the server. Use the forward slash as the file separator.

  A primary file must be specified unless checking in metadata only. If an alternate file is specified with the primary file, the content refinery will convert the alternate file. Otherwise, the primary file will be converted.
  - If a primary file is not specified, a metafile can be used in its place. Only one metafile can exist though for each content item (a primary AND alternate meta file cannot co-exist).
  - If both a primary and alternate file is specified, their extensions must be different.
- doFileCopy: 1 (*true*)—The file will not be deleted from the hard drive after checkin. 0 (*false*)—The file will be removed from your hard drive after checkin.
- Required custom fields: Custom metadata fields that are required must also be specified.

Optional Service Parameters

- alternateFile: The alternate file for conversion.
  - Only one metafile can exist though for each content item (a primary AND alternate meta file cannot co-exist.)
  - If an alternate file is specified with the primary file, the content refinery will convert the alternate file. Otherwise, the primary file will be converted.
- dCreateDate: The date the content item was created. By default, this is the current date.
- dRevLabel: The revision label for the content item. If set, the label will be used to locate the specified revision.
- doDocSecurityCheck: Enables the document security check. By default this entry is *false*.
- dPublishState: The publish state. If the content item exists in the system, this parameter must be empty.
- dReleaseState: The release state (used to indicate the web state of the revision).
- IsNative: This is a configuration entry but can be specified as a parameter.
  - If the *IsNative* parameter is *true* and the content item exists in the system, the specified Content ID (*dDocName*) must coincide with the content name in the database. Otherwise, the command is not native and the service will be determined as if the *IsNative* parameter is *false*.
  - If the *IsNative* parameter is *true* and the Action parameter is *Delete*, the content item must exist in the system. Otherwise, the command is not native and the service will be determined as if the *IsNative* parameter is *false*.
- webViewableFile: If the content is marked as web-viewable the file format is determined by the parameter *webViewableFile:format* and the extension is determined by the parameter *dWebExtension*. Otherwise, the extension and file format are determined by the parameters *dExtension* and *dFormat*, respectively. The user can override the file format and extension by overriding these parameters.
- Optional custom fields: Custom metadata fields that are not required can also be specified.

Results

- Local Data:
  - Action

- dRevClassID
- dRevisionID,
- dRendition2
- dRendition1
- dDocAuthor
- DocExists
- isCheckin
- StatusMessage
- primaryFile
- dStatus
- dRevLabel
- dWorkflowState
- dDocTitle
- StatusCode
- dpEvent
- isNew
- dUser
- isEditMode
- dDocName

Used By

❖ Applets: Batch Loader

Example

```
IdcService=CHECKIN_ARCHIVE
Action=insert
dDocAuthor=user1
dDocName=test
dDocTitle=new content
dSecurityGroup=Public
primaryFile=c:/test.txt
doFileCopy=true
```

See Also

– *CHECKIN_NEW_SUB (page 5-38)*

– *DELETE_BYCLASS (page 5-60)*
– *DELETE_BYREV (page 5-61)*
– *INSERT_NATIVE (page 7-27)*
– *INSERT_NEW (page 7-28)*

COPY_ARCHIVE

Service that copies an archive into a collection.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IdcService: Must be set to COPY_ARCHIVE.
- InstanceMenuLabel: The name of the collection. If you are using the default collection, this is the name of the content server instance.

Example

```
IdcService=COPY_ARCHIVE
InstanceMenuLabel=Master_on_server01
```

DELETE_ARCHIVE

Service that deletes an existing archive from a collection. The most likely error is when the specified archive does not exist on the system.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IdcService: Must be set to DELETE_ARCHIVE.
- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.

Example

```
IdcService=DELETE_ARCHIVE
IDC_Name=Master_on_server01
```

```
aArchiveName=archive_test
```

DELETE_BATCH_FILE

Service that deletes a batch file from an archive.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- aBatchFile: The batch file subdirectory and HDA file name.

Example

```
IdcService=DELETE_BATCH_FILE
IDC_Name=Master_on_server01
aArchiveName=archive_test
aBatchFile=02-jan-16_12.02.06_184/0216120206~1.hda
```

DELETE_BATCH_FILE_DOCUMENTS

Service used during batch file editing. It is available when using the **Archiver—General—View Batch Files—Edit** dialog. Using this dialog, a user can delete selected entries in a batch file.

When the user finishes editing and clicks **OK**, the changes are sent to the server. The delete actions performed during the dialog operation are handled by this service.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- aBatchFile: The batch file subdirectory and HDA file name.
- DeletedRows: A resultset which contains the items to delete.

Example

```
IdcService=DELETE_BATCH_FILE_DOCUMENTS
IDC_Name=Master_on_server01
aArchiveName=archive_test
aBatchFile=02-jan-16_16.53.02_289/0216165302~1.hda
```

DELETE_BATCH_FILE_TABLES

Service similar to DELETE_BATCH_FILE_DOCUMENTS except it is used on exported table data.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.

– *See Also*

– *DELETE_BATCH_FILE_DOCUMENTS (page 7-12)*

EDIT_ARCHIVE

Service used to change the description of an archive after it has been created. To access the description, double click on the archive in the archive list.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.

Example

```
IdcService=EDIT_ARCHIVE
IDC_Name=Master_on_server01
```

EDIT_ARCHIVEDATA

Service used to edit the data as specified in the EditItems parameter.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Important: Most archive services use the underlying method used by EDIT_ARCHIVEDATA and consequently require the EditItems parameter set to the appropriate value. The user interface usually controls what is put into this parameter. If necessary, the user should exercise these services through the user interface with Filter Debug flags enabled to determine how the values should be set. By using the Filter Debug flags, the request as it is sent to the server can be captured.

Additional Required Service Parameters

- EditItems: A comma-separated list of keys that are used to define the archive. Any key used must also be specified as a parameter. For example, if EditItems=aValueMaps, then aValueMaps is a required parameter to this service.
- KeyValue: Value for key in the EditItem parameter.
- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.

Example

```
IdcService=EDIT_ARCHIVEDATA
IDC_Name=Master_on_server01
```

EDIT_EXPORTERS

Service that edits the archive data for the specified archive. If specified, it registers or unregisters the archive.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- EditItems: The only values accepted are *aIsAutomatedExport* and *aRegisteredExporters*.

By setting EditItems, these parameters are also required:

- aIsAutomatedExport

- aRegisteredExporters

See Also

– *EDIT_ARCHIVEDATA (page 7-14)*

EDIT_TRANSFEROPTIONS

Service used to edit transfer options for an archive process.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.

EXECUTE_BATCH

Service that executes a batch operation for an archive.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- aBatchFile: The path and name of the batch file to be executed.

EXPORT_ARCHIVE

Service that exports an archive.

- The export will read in the archive's definition file and will follow the export rules specified. This includes specifying the export query, whether or not users and/or document configuration information should be exported, and whether or not previous batch files should be deleted.
- The content server executes this service asynchronously. This means that the server needs to be running for the services to be even executed and that IdcCommand will exit announcing success, when actually all it has done is told the server to perform the action.

Note: The EXPORT_ARCHIVE service only starts the archive export and provides confirmation that the request to start the export has been made. No notification is sent regarding the status or completion of the archive export. A custom component could be created to provide notification of the status and completion of the archive export.

- The content server can only export archives one at a time. A batch file that has multiple exports will declare success on the first and failure on all subsequent commands until it has finished the first. This means that a batch file should have only one export in it, and some external process will need to determine that the action has completed before issuing another export command.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- dataSource: Must be set to *RevisionIDs*. This is the query stub which, along with the export query, will be used to create the list of revisions to export.

Optional Service Parameters

- aDoDelete: 1 (*true*)—Revisions are deleted after successful export.

  0 (*false*)—Revisions are not deleted after export.

Example

- IdcCommand command file format (exports the archive *archive_test* in the collection *Master_on_server01*):

```
# To export an archive
IdcService=EXPORT_ARCHIVE
aArchiveName=archive_test
IDC_Name=Master_on_server01
dataSource=RevisionIDs
```

- HDA format with optional parameter:

```
@Properties LocalData
IdcService=EXPORT_ARCHIVE
aArchiveName=archive_test
IDC_Name=Master_on_server01
```

```
dataSource=RevisionIDs
aDoDelete=1
@end
```

GET_ARCHIVECOLLECTIONS

Service that returns a list of all archive collections.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_ARCHIVETABLECONTENT

Service used to preview the table data to be exported. It is used in the **Archiver—Export Data —Table—Preview** dialog.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- resultName: The result set name for the table content.

GET_ARCHIVED_FILE

Service that returns a specific rendition of an archived content item revision from an archive.

Given a *dID* or a *dDocName* and *RevisionSelectionMethod* parameter, the service determines the file name of a particular rendition of the revision and returns that file to the client.

The most likely errors are mismatched parameters or a request for a revision or rendition that does not exist.

Note: It is recommended that *dDocName* be included in all requests for content items where the requester knows the *dDocName*. Error messages in the content server assume that it is present, as do other features such as forms.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

Important: Either the content item revision ID (*dID*) must be specified or a Content ID (*dDocName*) along with a *RevisionSelectionMethod* parameter must be defined.

Optional Service Parameters

- dID: The generated content item revision ID.
  - If *dID* is not specified, *dDocName* and *RevisionSelectionMethod* must specified.
  - A rendition of the revision of the content item with this ID will be returned, if it exists, and the *RevisionSelectionMethod* parameter does not exist or has the value *Specific*.
- dDocName: The content item identifier (Content ID).
  - If *dDocName* is not present, *dID* must be present and *RevisionSelectionMethod* must not be present.
  - If *RevisionSelectionMethod* is present, a rendition of a revision of the content item with this name will be returned, if it exists. If *RevisionSelectionMethod* is not present, *dDocName* will be used in error messages.
- RevisionSelectionMethod: The revision selection method.
  - If present, *dDocName* must be present. The value of this variable is the method used to compute a *dID* from the specified *dDocName*. Its value can be *Specific*, *Latest*, or *LatestReleased*.
  - If the value is *Specific*, the *dDocName* is ignored, and *dID* is required and is used to get a rendition. If the value is *Latest*, the latest revision of the content item is used to compute the *dID*. If the value is *LatestReleased*, the latest released revision of the content item is used to compute the *dID*.
- Rendition: The content item rendition. This parameter specifies the rendition of the content item and can be set to *Primary*, *Web*, or *Alternate*. If *Rendition* is not present, it defaults to *Primary.*
  - If the value is *Primary*, the primary rendition of the selected revision is returned.
  - If the value is *Web*, the web-viewable rendition of the selected revision is returned.
  - If the value is *Alternate*, the alternate rendition of the selected revision is returned.

Example

```
IdcService=GET_ARCHIVED_FILE
dDocName=notice
RevisionSelectionMethod=LatestReleased
Rendition=web
```

See Also

– *GET_FILE (page 5-129)*

GET_ARCHIVES

Service that returns a list of all archives in a collection.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

❖ IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.

Example

```
IdcService=GET_ARCHIVES
IDC_Name=Master_on_server01
```

GET_ARCHIVERELATIONQUERY

Service used to query for the established relationships between exported tables. All parameters are provided by the user interface and come from the definition of the archive. Depending on the archive export definition, this service must be called before GET_ARCHIVETABLECONTENT.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *GET_ARCHIVETABLECONTENT (page 7-17)*

GET_BATCH_FILE_DOCUMENTS

Service that returns all batch file content items.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- aBatchFile: The batch file subdirectory and HDA file name.

Example

```
IdcService=GET_BATCH_FILE_DOCUMENTS
IDC_Name=Master_on_server01
aArchiveName=latest_archive
aBatchFile=02-jan-18_09.34.41_430/0218093441~1.hda
```

GET_BATCH_PROPERTIES

Service that returns the properties of the specified batchfile.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- aBatchFile: The batch file subdirectory and HDA file name.

GET_BATCH_SCHEMA

Service that returns the batch schema. Schema information describes the classes of objects that are stored in the database.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.

- aArchiveName: The archive name.
- aBatchFile: The batch file subdirectory and HDA file name.

Example

```
IdcService=GET_BATCH_SCHEMA
IDC_Name=Master_on_server01
aArchiveName=latest_archive
aBatchFile=02-jan-18_09.34.41_430/0218093441~1.hda
```

GET_BATCH_VALUES

Service that returns the metadata field values for the batch.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- aBatchFile: The batch file subdirectory and HDA file name.

Example

```
IdcService=GET_BATCH_VALUES
IDC_Name=Master_on_server01
aArchiveName=latest_archive
aBatchFile=02-jan-18_09.34.41_430/0218093441~1.hda
```

GET_BATCHFILES

Service that returns batch files for a specified archive.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.

Example

```
IdcService=GET_BATCHFILES
IDC_Name=Master_on_server01
aArchiveName=latest_archive
```

GET_PROXIED_ARCHIVECOLLECTIONS

Service that returns a list of all archive collections on a proxied content server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- psIDC_Name: The name of the proxied content server.

Example

```
IdcService=GET_PROXIED_ARCHIVECOLLECTIONS
psIDC_Name=Proxied_2_on_test13
```

GET_PROXIEDSERVERS

Service that returns a list of outgoing providers.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_REPLICATION_DATA

Service that returns replication data. The most likely error is an incorrect archive location.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.

Example

```
IdcService=GET_REPLICATION_DATA
IDC_Name=Master_on_server01
```

GET_TABLECOLUMNLIST

Service that retrieves the column information about specified tables. It returns the name, type, and length of the columns. In order to execute this service, the caller must have ADMIN rights.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- tableName: A list of comma-separated table names.

GET_TARGET_INFO

Service that performs a status check of a target. It does an initial status check to determine if the target is still targetable and if it exists.

This service is not intended for external use.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- TargetCollection: The name of the collection to be targeted.
- TargetArchive: The name of the archive within the target collection.
- aTransferOwner: The name of the user who is initiating the transfer.

GET_TARGET_TRANSFER_STATUS

Service used during transfer to determine transfer status. The TransferMonitor uses it to determine if the transfer has completed or is still in progress.

Not intended for external use.

Location: <*install_dir*>/shared/config/resources/std_services.htm

GET_TRANSFER_SOURCE_INFO

Service that is used during a pull transfer to establish the work that needs to be performed. Used when the source Archiver for the transfer resides on a proxied server and the transfer owner needs to determine what work, if any, must be done.

Not intended for external use.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- SourceArchive: The archive used as source for the target.
- aTransferOwner: The name of the user who is initiating the transfer.
- aArchiveName: The archive name.

IMPORT_ARCHIVE

Service that imports an archive.

- The import will read in the archive's definition file and will follow the import rules specified.
- The content server executes this service asynchronously. This means that the content server needs to be running for the service to be executed and that IdcCommand will announce success as soon as the service has been requested, not when it has been executed successfully.

Note: The IMPORT_ARCHIVE service only starts the archive import and provides confirmation that the request to start the import has been made. No notification is sent regarding the status or completion of the archive import. A custom component could be created to provide notification of the status and completion of the archive import.

- The server can only import archives one at a time. A batch file that has multiple imports will declare success on the first and failure on all subsequent commands until it has finished the first. This means that a batch file should have only one import in it

and some external process will need to determine that the action has completed before issuing another import command.

- The most likely error is an archive name that does not exist.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.

Note: At least one of the Additional Service Parameters (page 7-25) must be set for this service to execute successfully.

Additional Service Parameters

- aImportDocuments: When set to *true*, the service will import content item revisions.
- aImportDocConfig: When set to *true*, the service will import document configuration information.
- aImportUsers: When set to *true*, the service will import user information.

Important: If the users have not been exported and aImportUsers is set to *true*, the service fails. If the document configuration has not been exported and this parameter is set to *true*, the service fails.

Example

- IdcCommand command file format (import the revisions in the *archive_test* archive from the *Master_on_server01* collection):

```
# To import an archive
IdcService=IMPORT_ARCHIVE
aArchiveName=archive_test
IDC_Name=Master_on_server01
aImportDocuments=true
```

- HDA format:

```
@Properties LocalData
IdcService=IMPORT_ARCHIVE
aArchiveName=archive_test
IDC_Name=Master_on_server01
```

```
aImportDocuments=true
@end
```

IMPORT_ARCHIVE_START_AUTOMATED

Service that registers a queued archive for import.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.

IMPORT_BATCHFILE

Service that imports the content items in the specified batch files, as selected from the **View Batch File** dialog, into the Content Server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.

IMPORT_DOCUMENT

Service that imports a specified document, usually selected from the **View Batch File** dialog, into the Content Server.

The most likely error is a content item name that does not exist.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.

- aArchiveName: The archive name.
- dDocName: The Content ID for the content item.

Example

```
IdcService=IMPORT_DOCUMENT
IDC_Name=Master_on_server01
aArchiveName=JAN_22_02
dDocName=billing_00004
```

See Also

– *IMPORT_TABLE_ENTRY (page 7-27)*

IMPORT_TABLE_ENTRY

Service used to add entries to a table in an archive. This service is equivalent to IMPORT_DOCUMENT. It works on exported table items instead of exported content items. It is accessible from the **View Batch File** dialog and is used to import the specified table item.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- aBatchFile: The path and name of the batch file where the table is stored.

See Also

– *IMPORT_DOCUMENT (page 7-26)*

INSERT_NATIVE

Subservice used by the Archiver to recover IDs created during INSERT_NEW. This service does not generate new IDs. It reuses the IDs stored during the export. This subservice is only used when the administrator is archiving back into the original system that created the export. The Archiver uses INSERT_NATIVE when the exporter and

importer have the same instance name (IDC_name) and the system is trying to recover the content item completely.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_ARCHIVE (page 7-6)*

INSERT_NEW

Subservice used during a CHECKIN_ARCHIVE when the system has discovered no prior item with the specified dDocName. This subservice is used during a batch load as well as an archive import operation. It adds a new content item into the system and creates new IDs (such as dID, dRevClassID) for it.

Location: <*install_dir*>/shared/config/resources/std_services.htm

See Also

– *CHECKIN_ARCHIVE (page 7-6)*

REGISTER_IMPORTER

Service that registers or unregisters the importer for an archive. The most likely errors are mismatched parameters or an incorrect instance menu label or archive name.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.
- EditItems: Must be set to *aRegisteredImporter,aImportLogonUser*.
- IsRegister: Defines the registration setting:
  - 1 (*true*)—Register Self
  - 0 (*false*)—Unregister

Example

Self register the content server as importer:

```
IdcService=REGISTER_IMPORTER
IDC_Name=Master_on_server01
aArchiveName=JAN_22_02
IsRegister=1
EditItems=aRegisteredImporter,aImportLogonUser
```

REMOVE_COLLECTION

Service that removes an archive collection. You cannot remove the default archive collection.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection.

Example

```
IdcService=REMOVE_COLLECTION
IDC_Name=new_collection
```

REMOVE_EXPORTER

Service that removes an exporter from an archive. The most likely error is an incorrect archive location.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- aArchiveName: The archive name.
- IDC_Name: The name of the exporter.

Example

```
IdcService=REMOVE_EXPORTER
IDC_Name=Master_on_server02
aArchiveName=archive_02
```

REMOVE_IMPORTER

Service that removes an importer from an archive. The most likely error is an incorrect archive location.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the importer. If you are using the default collection, this is the name of the content server instance.
- aArchiveName: The archive name.

Example

```
IdcService=REMOVE_IMPORTER
IDC_Name=Master_on_server01
aArchiveName=my_archive_test
```

REMOVE_PROXIEDTRANSFER

Service that is called when removing the transfer settings for an archive but the Content Server performing the activity is not the Content Server that owns the archive (that is, the transfer is being proxied). To remove or stop the transfer, the request must be submitted to this proxied server.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- aArchiveLocation: Location of the archive.
- IDC_Name: The name of the owner of the transfer.

REMOVE_QUEUED_IMPORT

Service that allows a user to delete a queued import. It is used from the Automation dialog, accessible from the **Archiver Options—View Automation—Queued Automated Imports** menu.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the importer. If you are using the default collection, this is the name of the content server instance.

REMOVE_TRANSFER

Service that allows a user to delete or stop an automated transfer. It is used from the Automation dialog, accessible from the **Archiver Options—View Automation** menu.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- aArchiveLocation: Location of the archive.
- IDC_Name: The name of the importer. If you are using the default collection, this is the name of the content server instance.

REQUEST_TRANSFER

Service that initiates a transfer request.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- aArchiveName: The archive name.
- IDC_Name: The name of the exporter. If you are using the default collection, this is the name of the content server instance.

TRANSFER_ARCHIVE

Service that initiates a specified manual transfer.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- aArchiveName: The archive name.
- IDC_Name: The name of the exporter. If you are using the default collection, this is the name of the content server instance.

UPDATE_TARGET_TOTALS

Service that updates the amount of work the archiver has completed in the archive definition file. This service is not intended for external use. It is part of the Transfer engine that is responsible for monitoring the automated transfer as well as updating the data on completion of any manual or automated transfers.

Location: <*install_dir*>/shared/config/resources/std_services.htm

UPDATE_TRANSFER_STATUS

Service used by the TransferMonitor to send update information about the progress of the transfer. Not intended for use by client applications.

Location: <*install_dir*>/shared/config/resources/std_services.htm

UPLOAD_ARCHIVE_TRANSFER

Service used to upload the export batch file to the target. This service is not extended for external use and is part of the service that make up the Transfer engine of the Archiver.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection. If you are using the default collection, this is the name of the content server instance.
- TargetCollection: The name of the collection to be targeted.
- TargetArchive: The name of the archive within the target collection.
- SourceCollection: The name of the collection used for the transfer.
- SourceArchive: The archive used as source for the target.
- aBatchFile: The name of the batch file used in the transfer.
- ZipFile:path: The pathname to the zip file of the transfer to be uploaded.

Chapter

8

FOLDERS SERVICES

OVERVIEW

This chapter describes the services available when using and customizing archives. The location for specific archive services are listed within each individual service.

This chapter covers the following topics:

- About Folder Services (page 8-1)
- Folder Services (page 8-2)

ABOUT FOLDER SERVICES

A typical service entry is similar to the following:

```
COLLECTION_ADMIN_SELECT_COLLECTION
DocService 56 COLLECTION_PROFILE_SELECT_COLLECTION null null
    !csCollectionUnableToRetrieveCollections
3:evaluate:<$isAdmin="true"$>::null
3:loadAdminQueryConfig:::null
```

In this example:

- the first line indicates the service name: `COLLECTION_ADMIN_SELECT_COLLECTION`.
- The second line and third lines list the service's attributes:
  - Service class: `DocService`.
  - Access level: `54`, indicating this is Scriptable, Global, and Admin permissions.
  - Template: `COLLECTION_PROFILE_SELECT_COLLECTION`

- Service type: `null`
- Subjects Notified: `null`
- Error message and parameter used for error message: `Unable to Retrieve Collections`

❖ The remainder of the entry lists the service's actions, each line designating one action:

- action type: for example, `3`, indicating a Java method.
- action name: for example, `evaluate`
- action parameters: for example, `<$isAdmin="true"$>`. If no parameters are specified, double colons (`::`) are included.
- action control mask: for example, `0`, indicating "Ignore error" or `2`, indicating "Must exist (at least one record must be returned by the query). If no control mask is specified, double colons (`::`) or 0 is included.
- action error message, such as `Item No Longer In History`, or null if no message is available.

Each element of the service's action is separated by a colon (`:`).

Important: All services have at least one required parameter. The IdcService parameter takes the name of the service as its argument. If other parameters are required, they are noted in the description of the service.

See Chapter 4 (*Customizing Services)* for details about a service's attributes and actions.

FOLDER SERVICES

Individual service types are noted in each service description. The following services are described in this section:

❖ COLLECTION_ADD (page 8-5)

❖ COLLECTION_ADD_LINK (page 8-6)

❖ COLLECTION_ADD_MOUNT (page 8-6)

❖ COLLECTION_BROWSE (page 8-7)

❖ COLLECTION_CHECKIN_NEW (page 8-7)

❖ COLLECTION_CHECKIN_REVISION (page 8-7)

❖ COLLECTION_CHECKIN_SEL_SUB (page 8-8)

- COLLECTION_COPY_ALL (page 8-8)
- COLLECTION_COPY_COLLECTION (page 8-8)
- COLLECTION_COPY_ITEM (page 8-9)
- COLLECTION_COPY_LOT (page 8-9)
- COLLECTION_DELETE_ALL (page 8-9)
- COLLECTION_DELETE_COLLECTION (page 8-9)
- COLLECTION_DELETE_ITEM (page 8-9)
- COLLECTION_DELETE_LOT (page 8-10)
- COLLECTION_DISPLAY (page 8-10)
- COLLECTION_EDIT (page 8-11)
- COLLECTION_GET_ADMIN_CONFIG (page 8-11)
- COLLECTION_GET_ADMIN_INHERIT_CONFIG (page 8-11)
- COLLECTION_GET_ADMIN_MARKED_CONFIG (page 8-11)
- COLLECTION_GET_ADMIN_META_CONFIG (page 8-12)
- COLLECTION_GET_ADMIN_METADATA_DEFAULTS (page 8-12)
- COLLECTION_GET_ADMIN_MOUNTED_CONFIG (page 8-12)
- COLLECTION_GET_ARCHIVE (page 8-12)
- COLLECTION_GET_BRANCH (page 8-13)
- COLLECTION_GET_COLLECTION (page 8-13)
- COLLECTION_GET_COLLECTIONS (page 8-13)
- COLLECTION_GET_CONTENT_FILE (page 8-13)
- COLLECTION_GET_CONTENTS (page 8-14)
- COLLECTION_GET_FILE (page 8-14)
- COLLECTION_GET_INFO (page 8-14)
- COLLECTION_GET_LINKS (page 8-14)
- COLLECTION_GET_META_MAPPING (page 8-14)
- COLLECTION_GET_PROFILE_METADATA_DEFAULTS (page 8-15)
- COLLECTION_GET_PROFILE_METADATA_REVISION_DEFAULTS (page 8-15)

- COLLECTION_GET_REFERENCE (page 8-15)
- COLLECTION_GET_SEARCH_FORM (page 8-15)
- COLLECTION_GET_SYSTEM_FILE (page 8-16)
- COLLECTION_GET_USER_CONFIG (page 8-16)
- COLLECTION_INDEX_MOUNTS (page 8-16)
- COLLECTION_INFO (page 8-16)
- COLLECTION_ISVALID_META (page 8-17)
- COLLECTION_LOCK (page 8-17)
- COLLECTION_MOVE_ALL (page 8-17)
- COLLECTION_MOVE_COLLECTION (page 8-17)
- COLLECTION_MOVE_ITEM (page 8-17)
- COLLECTION_MOVE_LOT (page 8-18)
- COLLECTION_NEW (page 8-18)
- COLLECTION_PROFILE_UPDATE_COLUMNS (page 8-19)
- COLLECTION_REBUILD_MOUNT (page 8-19)
- COLLECTION_REMOVE_MOUNT (page 8-19)
- COLLECTION_RESTORE_COLLECTION (page 8-19)
- COLLECTION_RESTORE_ITEM (page 8-20)
- COLLECTION_SEARCH_CONTENT (page 8-20)
- COLLECTION_SEARCH_RESULTS (page 8-20)
- COLLECTION_SET_ARCHIVE (page 8-20)
- COLLECTION_SET_USER_CONFIG (page 8-20)
- COLLECTION_UNLOCK (page 8-21)
- COLLECTION_UPDATE (page 8-21)
- COLLECTION_UPDATE_ADMIN_CONFIG (page 8-22)
- COLLECTION_UPDATE_ADMIN_INHERIT_CONFIG (page 8-22)
- COLLECTION_UPDATE_ADMIN_METADATA_DEFAULTS (page 8-22)
- COLLECTION_UPDATE_ALL (page 8-22)

- COLLECTION_UPDATE_ITEM (page 8-22)
- COLLECTION_UPDATE_META (page 8-23)
- COLLECTION_UPDATE_META_TABLE (page 8-23)
- COLLECTION_UPDATE_MOUNTS (page 8-23)
- COLLECTION_UPDATE_PROFILE_METADATA_DEFAULTS (page 8-23)
- COLLECTION_UPDATE_PROFILE_METADATA_REVISION_DEFAULTS (page 8-24)
- COLLECTION_UPDATE_STRUCTURE (page 8-24)
- GOTO_COLLECTION (page 8-24)
- GOTO_ROOT_COLLECTION (page 8-24)
- GET_OPTION_LISTS (page 8-24)
- GET_ENVIRONMENT (page 8-25)

COLLECTION_ADD

Service that creates a content server folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Additional Required Service Parameters

- hasParentCollectionID: Defines (true/false) whether the folder being added as a parent folder. If set to *true*, dParentCollectionID must be defined.
- dCollectionName: The folder name.
- dParentCollectionID: The folder ID for the parent folder. Used when hasParentCollectionID is set to *true*.
- dCollectionOwner: The folder owner (defined when the folder is created).

Example

```
IdcService=COLLECTION_ADD
hasParentCollectionID=true
dCollectionName=Products
```

```
dParentCollectionID=1
dCollectionOwner=bsmith
```

COLLECTION_ADD_LINK

Service that adds a shortcut to a content server folder. The shortcut references the folder using either the folder ID or the actual path to the folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Additional Required Service Parameters

- dCollectionID: The folder ID of the parent folder referenced by the shortcut.
- dDocID: The content ID. Used when making a shortcut to a content item rather than a folder.
- dLinkName: The shortcut destination.
- dLinkType: The shortcut type. Must be either COLLECTION or CONTENT.
- tohasCollectionID: Defines (true/false) whether the folder is referenced using the folder ID. If set to *true*, todCollectionID nust be defined.
- todCollectionID: The folder ID that the shortcut references.

Example

```
IdcService=COLLECTION_ADD_LINK
dCollectionID=194
dLinkType=COLLECTION
tohasCollectionID=false
todCollectionID=" "
```

COLLECTION_ADD_MOUNT

Service that maps a folder to a local file system.Whenever the folder changes, the mapped directory is updated.

There is no security on this directory.

The most likely errors are mismatched parameters or the folder does not exist.

Service Class: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_mounted_service.htm

COLLECTION_BROWSE

Service that loads the defined folder metadata and the folder path. The most likely errors are mismatched parameters or the folder does not exist.

Service Class: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_interface_service.htm

Additional Required Service Parameters

- dCollectionID: The folder ID of the parent folder referenced by the shortcut.

  For example, ="+id+"
- hasCollectionID: Defines (true/false) whether the folder has an assigned collection ID.
- changeToUser: Reassigns the folder to a different user.

Example

```
IdcService=COLLECTION_BROWSE
dCollectionID="+id+"
hasCollectionID=true
changeToUser=bsmith
```

COLLECTION_CHECKIN_NEW

Service that checks new content into a folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Class: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_CHECKIN_REVISION

Service that checks in a revision to an existing folder.

This service sends the checkin request to one of the following subservices, which are the same subservices called during checkin through the browser or Repository Manager application. (These subservices are not called during a Batch Loader or Archive import.)

- COLLECTION_CHECKIN_SEL_SUB
- CHECKIN_SEL_FORM
- COLLECTION_SEARCH_CONTENT
- CHECKOUT_BY_NAMENULL
- COLLECTION_LOCKNULL
- DELETE_REVNULL

This service checks security to determine if the user has sufficient permission to check in the folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Class: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_CHECKIN_SEL_SUB

Service that checks in a revision to an existing folder under certain conditions. Used when the folder exists on the system but no valid revision was specified or when the content item is checked out (but not in a workflow).

The most likely errors are mismatched parameters or the folder does not exist.

Service Class: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_COPY_ALL

Service that copies all content residing in the specified folder to the destination folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_COPY_COLLECTION

Service that copies a specified content server folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_COPY_ITEM

Service that copies a single content item in a folder to a destination folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: File Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_COPY_LOT

Service that copies a group of folders. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_DELETE_ALL

Service that deletes all content residing in a folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_DELETE_COLLECTION

Service that deletes a specified content server folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_DELETE_ITEM

Service that deletes a single content item in a folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_DELETE_LOT

Service that deletes a group of folders. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Additional Required Service Parameters

- tohasCollectionID: Defines (true/false) if the collection is referenced using the collection ID. If set to *true*, todCollectionID nust be defined.
- todCollectionID: The folder ID that the shortcut references.
- todCollectionPath: The URL path to the folder.

Example

```
IdcService=COLLECTION_DELETE_LOT
tohasCollectionID=true
todCollectionID=194
todCollectionPath=""
```

COLLECTION_DISPLAY

Service that loads folder metadata and displays the defined folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_interface_service.htm

Additional Required Service Parameters

- hasCollectionID: Defines (true/false) if the folder has an assigned collection ID.
- dCollectionPath: The folder containing the folder.
- orderby: Defines the column to sort on.
- direction: Defines the direction of the order as descending (*DESC*) or ascending (ASC).

Example

```
IdcService=COLLECTION_DISPLAY
dCollectionPath=%2fContent%20Server%20Folders%2f
orderby=dName
direction=ASC
```

COLLECTION_EDIT

Service that edits a content server folder. The most likely errors are mismatched parameters or the folder does not exist.

Service type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Additional Required Service Parameters

- hasCollectionID: Defines (true/false) if the folder has an assigned collection ID.
- dCollectionID: The folder ID of the parent folder referenced by the shortcut.

COLLECTION_GET_ADMIN_CONFIG

Service that retrieves the current global values defined for the folders hierarchy. No specific collection is passed. The most likely errors are mismatched parameters or the folder does not exist.

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_ADMIN_INHERIT_CONFIG

Service that retrieves the current metadata fields that are propagated when it is requested. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_ADMIN_MARKED_CONFIG

Service that retrieves “special” folders and their disabled/enabled status.

A disabled folder does not show up in the hierarchy. An enabled folder is accessible.

The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_ADMIN_META_CONFIG

Service that retrieves the fields from the administration metadata ResultSet for the defined folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_ADMIN_METADATA_DEFAULTS

Service that retrieves the option lists for custom fields, default metadata, and the administration metadata of the folder configuration (retrieves field defaults).

The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_ADMIN_MOUNTED_CONFIG

Service that retrieves the folders system configuration for the local folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_mounted_service.htm

COLLECTION_GET_ARCHIVE

Service that downloads the archived folder structure of the specified folder. This service does not download the contents of a folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: File Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_BRANCH

Service that retrieves the structure of the specified folder and its subfolders. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_COLLECTION

Service that retrieves all the subfolders of the specified folder. Compare COLLECTION_GET_COLLECTIONS. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_COLLECTIONS

Service that retrieves the content server folder. Compare COLLECTION_GET_COLLECTION The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Example

To retrieve a listing of hierarchical collections, the required parameters are:

```
hasCollectionID=1
dCollectionID=collectionnumber
```

COLLECTION_GET_CONTENT_FILE

Service that downloads the first content item of the specified collection that matches the metadata of the item passed in (excluding revisions). The most likely errors are mismatched parameters or the folder does not exist.

Service Type: File Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_CONTENTS

Service that retrieves all the sub-content of the specified folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_FILE

Service that determines the type of item specified and then downloads its content. The item may or may not be a content item. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: File Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_INFO

Service that retrieves content server folder information for the local folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_LINKS

Service that retrieves the specified shortcut (dLinkID). The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_META_MAPPING

Service used to retrieve the WebDAV equivalent properties of the specified folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_PROFILE_METADATA_DEFAULTS

Service used to retrieve the metadata replacements for content that is checked in as a revision. If a field is not defined, the old value is retained. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_PROFILE_METADATA_REVISION_DEFAULTS

Service that retrieves field defaults for a specified revision (retrieves option lists for custom fields, revision defaults, and metadata defaults). The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_REFERENCE

Service used to determine what type of item the path being passed in is referencing (Link, Content, or Folder). The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_SEARCH_FORM

Service that retrieves the folder specific search form. The most likely errors are mismatched parameters or the folder does not exist.

Search Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_SYSTEM_FILE

Service that retrieves the files that are usually dynamically built at the time of the request and have no counterpart on the system. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: File Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_GET_USER_CONFIG

Service that retrieves user specific configuration for the display and behavior of their folder structure. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location; <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_INDEX_MOUNTS

Service that updates content that has been demoted by the content server indexer to the local folder by the new version. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_mounted_service.htm

COLLECTION_INFO

Service that retrieves the metadata associated with this particular folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Additional Required Service Parameters

- hasCollectionID: Defines (true/false) if the folder has an assigned collection ID.
- dCollectionID: The folder ID of the parent folder referenced by the shortcut.

COLLECTION_ISVALID_META

Service that verifies that the folder can be added by the item of the specified metadata. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_LOCK

Service that checks out a folder by name (dDocName) and locks the folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_MOVE_ALL

Service that moves the content in a specified folder to the destination folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_MOVE_COLLECTION

Service that moves a folder. Compare COLLECTION_MOVE_ALL. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_MOVE_ITEM

Service that moves a single content item from the specified folder to the destination folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_MOVE_LOT

Service that is capable of moving multiple references to folders and content items to a destination folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Additional Required Service Parameters

- tohasCollectionID: Defines (true/false) if the folder is referenced using the folder ID.
- If set to true, todCollectionID must be defined.
- todCollectionID: The folder ID that the shortcut references.
- todCollectionPath: The URL path to the folder.

Example

```
IdcService=COLLECTION_MOVE_LOT
tohasCollectionID=true
todCollectionID=194
todCollectionPath=""
```

COLLECTION_NEW

Service used to create a new folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Additional Required Service Parameters

- hasParentCollectionID: Defines (true/false) if the folder being added as a parent folder. If set to *true*, dParentCollectionID must be defined.
- dParentCollectionID: The folder ID for the parent folder. Used when hasParentCollectionID is set to *true*.
- dCollectionInherit: Used to inherit folder metadata.

Example

```
IdcService=COLLECTION_NEW
hasParentCollectionID=true
dParentCollectionID=1
dCollectionInherit=0
```

COLLECTION_PROFILE_UPDATE_COLUMNS

Service that updates user hierarchy columns. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_REBUILD_MOUNT

Service that rewrites the files of the specified local folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_mounted_service.htm

COLLECTION_REMOVE_MOUNT

Service that removes the specified local folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_mounted_service.htm

COLLECTION_RESTORE_COLLECTION

Service that restores a folder that is currently in the “Trash” folder to its original location. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_RESTORE_ITEM

Service that restores an item that is currently in the “Trash” folder to its original location. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_SEARCH_CONTENT

Service that retrieves all content that matches all of the metadata that is passed in for a content item. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_SEARCH_RESULTS

Service that displays the search results of a folder search. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_SET_ARCHIVE

Service that takes the uploaded archive and applies it to the system. This service deletes any folders that are in conflict with the archive. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_SET_USER_CONFIG

Service that sets the user configuration for the display and behavior of their folder structure. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UNLOCK

Service that undoes a checkout of a content item and makes it available for use. This service frees content after a COLLECTION LOCK. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE

Service that updates information for a collection. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Additional Required Service Parameters

- dCollectionName: The folder name.
- hasParentCollectionID: Defines (true/false) whether the folder being added as a parent folder. If set to *true*, dParentCollectionID must be defined.
- dParentCollectionID: The folder ID for the parent folder. Used when hasParentCollectionID is set to *true*.
- dCollectionOwner: The folder owner (defined when the folder is created).

Example

IdcService=COLLECTION_UPDATE

dCollectionName=Products

dParentCollectionID=1

dCollectionOwner=bsmith

COLLECTION_UPDATE_ADMIN_CONFIG

Service that updates the folders system configuration settings. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_ADMIN_INHERIT_CONFIG

Service that updates the folders system configuration setting by inheriting the metadata from an existing folder (from which metadata is propagated). The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_ADMIN_METADATA_DEFAULTS

Service that updates the folders system metadata field default system settings. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_ALL

Service that updates all items in a collection and changes the defined metadata. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_ITEM

Service that updates a specific item in a collection and changes the defined metadata. Folders related service (table Folders_Services). The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_META

Service that updates folder metadata. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Meta Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_META_TABLE

Service that updates a folder metadata table. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Meta Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_MOUNTS

Service that updates the files of the specified local folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Search Service

Location: <*install_dir*>/custom/Folders/resources/folders_mounted_service.htm

COLLECTION_UPDATE_PROFILE_METADATA_DEFAULTS

Service that updates the profile metadata field defaults. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_PROFILE_METADATA_REVISION_DEFAULTS

Service that updates the profile metadata field defaults of a revision. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

COLLECTION_UPDATE_STRUCTURE

Service that refreshes the folder cache. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Meta Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

GOTO_COLLECTION

Service that displays a folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

GOTO_ROOT_COLLECTION

Service that displays the root folder. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

GET_OPTION_LISTS

Folders-related service that retrieves the option lists. The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Service (general service)

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

GET_ENVIRONMENT

Service that loads the environment (takes the content server settings and retrieves them). The most likely errors are mismatched parameters or the folder does not exist.

Service Type: Doc Service

Location: <*install_dir*>/custom/Folders/resources/folders_service.htm

Chapter

9

CONTENT PUBLISHER SERVICES

OVERVIEW

This chapter covers the following topics:

- About Content Publisher Services (page 9-1)
- Content Publisher Services (page 9-2)

ABOUT CONTENT PUBLISHER SERVICES

A typical Content Publisher service entry is similar to the following:

```
GET_FILE
FileService 3 null null null !csUnableToDownloadFile
3:computeDocID::0:!csUnableToDownload(dDocName)
5:QdocInfo:FILE_DOC_INFO:2:!csItemNoLongerInSystem
3:checkSecurity:FILE_DOC_INFO:0:!csUnableToDownload(dDocName)
3:createFileName:FILE_DOC_INFO:0:null</td><ADD EXAMPLE
```

- The first line indicates the service name: `GET_FILE`
- The second line lists the service's attributes:
  - Service class: `FileService`
  - Access level: `3`

- Template: `null` (none used)
- Service type: `null` (not a subservice)
- Subjects Notified: `null`
- Error message: `Unable to Download File`

❖ The remainder of the entry lists the service's actions, each line designating one action:

- action type: for example, `3`, indicating a Java method.
- action name: for example, `checkSecurity`
- action parameters: for example, `FILE_DOC_INFO`. If no parameters are specified, double-colons (`::`) are included.
- action control mask: for example, `0`, indicating no control mask. The double-colon indicator (`::`) can also be included.
- action error message: for example, `Unable to Download (dDocName)`, or null if no message is available.

Each element of the service's action is separated by a colon (`:`).

Important: All services have at least one required parameter. The IdcService parameter takes the name of the service as its argument. If other parameters are required, they are noted in the description of the service.

See Chapter 4 (*Customizing Services)* for details about a service's attributes and actions.

CONTENT PUBLISHER SERVICES

The following services are used with Content Publisher and are described elsewhere in this documentation. Any specific parameters necessary for Content Publisher are noted in those descriptions.:

❖ ASSIGN_DOCINFO_FORM (page 5-24)

❖ CHECKIN_ARCHIVE (page 7-6)

❖ CHECKOUT_BY_NAME (page 5-55)

❖ DELETE_DOC (page 5-61)

❖ DOC_INFO (page 5-64)

❖ DOC_INFO_BY_NAME (page 5-68)

❖ DOC_INFO_SIMPLE (page 5-71)

- DOC_INFO_SIMPLE_BYREV (page 5-72)
- GET_ACTIVE_WORKFLOWS (page 6-37)
- GET_DOC_CONFIG_INFO (page 5-77)
- GET_DOC_PAGE (page 5-146)
- GET_FILE (page 5-129)
- GET_PREVIEW_LIST (page 9-5)
- GET_PREVIEW_LIST_EX (page 9-5)
- GET_PROBLEMREPORTS_SEARCH_RESULTS (page 6-6)
- GET_PROJECT_INFO (page 9-6)
- GET_RESULT_OPTIONS (page 5-12)
- GET_SEARCH_RESULTS (page 5-174)
- GET_SECURE_PAGE (page 5-151)
- GET_TABLE (page 5-13)
- GET_WORKFLOW_INFO (page 6-39)
- GET_WORKFLOWDOCREVISIONS (page 6-42)
- NOTIFY_INDEXER (page 5-81)
- PING_SERVER (page 5-19)
- PREVIEW_DOCUMENT (page 9-7)
- PROBLEMREPORT_INFO (page 6-9)
- UNDO_CHECKOUT_BY_NAME (page 5-95)

The following services are described in this chapter:

- CHECKIN_PUBLISH (page 9-4)
- END_PUBLISHING (page 9-4)
- GET_PROJECT_ACTIONS (page 9-5)
- GET_PROJECT_REGISTRATION (page 9-6)
- GET_REGISTEREDPROJECTS (page 9-7)
- PREVIEW_DOCUMENT (page 9-7)
- REGISTER_PROJECT (page 9-7)

- START_PUBLISHING (page 9-7)
- UNREGISTER_PROJECT (page 9-8)

CHECKIN_PUBLISH

Service used to check in a project for Publisher.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dProjectID: The Content Publisher project ID generated by Content Publisher.
- dPrjTopParent: The publication ID of the topmost publication in the project to which the document belongs.
- dPrjMiddleParent: The path of publication ID values leading from the dPrjTopParent to the dPrjIntermediateParent.
- dPrjImmediateParent: The publication ID of the immediate parent publication to which the project belongs.

Optional Service Parameters

- agentID: An identifier for the software that is calling the service. Content Publisher uses the reserved agentID `ContentPublisher`.
- Action: Always set to the value `update`.
- isStaging: set to `1` if staging, `0` if publishing.
- docPublishType: A comma-separated list containing the type of document being published. Acceptable values include: `entrypoint`, `toc`, `lof`, `lot`, `index`, `query`, `results`, `frameset`, `page`, `splitpage`, `firstpage`, `graphic`, `gallery`, `assoc`, `mapfile`, `passthru`, `unknown`.
- `isPendingContributorRelease`: An indicator of whether the document producing this output is in a released state.

END_PUBLISHING

A service that signals the end of a publishing operation.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dProjectID: The Content Publisher project ID generated by Content Publisher.
- agentID: An identifier for the software that is calling the service. Content Publisher uses the reserved agentID `ContentPublisher`.

GET_PREVIEW_LIST

Service that retrieves the list of Content Publisher projects the specified item falls into or matches. From a page generated by this service, the user can decide to preview the content item in the context of the selected Content Publisher project.

This service queries the checkin cache for information to create the preview list. The cached data determines the metadata for the content item and is used to discover which projects the content item may be previewed in. If the content item is no longer in the cache, the service fails.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The document ID for the content item.

GET_PREVIEW_LIST_EX

Service similar to GET_PREVIEW_LIST except it does not use cached data to compute the potential Content Publisher projects. The content information or metadata is retrieved from the database. See GET_PREVIEW_LIST (page 9-5) for details.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The document ID for the content item.
- dID: The content ID for the content item.

GET_PROJECT_ACTIONS

Service that lists actions associated with a project.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dProjectID: The Content Publisher project ID generated by Content Publisher.
- agentID: An identifier for the software that is calling the service. Content Publisher uses the reserved agentID `ContentPublisher`.

GET_PROJECT_INFO

Service that returns information about all content items associated with a Content Publisher project.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- dProjectID: The Content Publisher project ID generated by Content Publisher.

Example

```
IdcService=GET_PROJECT_INFO
dProjectID=servG0SGXDW5YD55T80NHPR57MWW40
```

GET_PROJECT_REGISTRATION

Service that returns workflow registration information for a Content Publisher project.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- projectID: The Content Publisher project ID generated by Content Publisher.

Important: Use *projectID* for this service (not *dProjectID*).

Example

```
IdcService=GET_PROJECT_REGISTRATION
projectID=saviZN0LN18G9D6THX91KD4O9RTMKG
```

GET_REGISTEREDPROJECTS

Service that returns information about all registered Content Publisher projects.

Location: <*install_dir*>/shared/config/resources/workflow.htm

PREVIEW_DOCUMENT

Service that submits the designated content item to the Content Publisher preview engine via HTMLPreview.

Location: <*install_dir*>/shared/config/resources/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
- projectID: The unique identifier for the Content Publisher project.
- pagePathID: The Content Publisher node.

Both projectID and pagePathID are computed from the information retrieved from GET_PREVIEW_LIST.

See Also

– *GET_PREVIEW_LIST (page 9-5)*

REGISTER_PROJECT

Service used to register a Content Publisher project.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- projectID: The Content Publisher project ID generated by Content Publisher.
- functions: An indicator of the intended usage of the project. Acceptable values include: `registered`, `preview`, `stagingworkflow`.

START_PUBLISHING

A service that signals the start of a publishing operation.

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- projectID: The Content Publisher project ID generated by Content Publisher.

UNREGISTER_PROJECT

Location: <*install_dir*>/shared/config/resources/workflow.htm

Additional Required Service Parameters

- projectID: The Content Publisher project ID generated by Content Publisher.
- functions: An indicator of the intended usage of the project. Acceptable values include: `registered`, `preview`, `stagingworkflow`.

Chapter

10

RECORD MANAGER SERVICES

OVERVIEW

The following sections list the services available for Record Management.

- About Record Manager Services (page 10-1)
- Record Manager Services (page 10-2)

ABOUT RECORD MANAGER SERVICES

A typical service entry is similar to the following:

```
LIST_DISPOSITION_RECORDS
Service 33 PENDING_REVIEW_LIST null null !csUnableToGetRecordsList
3:hasRmaRights:admin.performactions:0:null
3:checkRecordsFoldersMap::0:null
3:setLocalValuesAfterEval:dRequestedUser,<$dUser$>::null
3:setLocalValues:IsNavigation,1::null
```

In this example:

- the first line indicates the service name: `LIST_DISPOSITION_RECORDS`
- The second line lists the service's attributes:
  - Service class: `service` (generic service)
  - Access level: `33`
  - Template: `PENDING_REVIEW_LIST`

- Service type: `null`
- Subjects Notified: `null`
- Error message: `Unable to get records list`

❖ The remainder of the entry lists the service's actions, each line designating one action:

- action type: for example, `3`, indicating a Java method.
- action name: for example, `hasRmaRights`
- action parameters: for example, `dRequestUser`. If no parameters are specified, `::` are included.
- action control mask: for example, `1`, indicating "Ignore error". If no control mask is specified, `::` is included.
- action error message, or null if no message is available.

Each element of the service's action is separated by a colon (`:`).

See Chapter 4 (*Customizing Services)* for details about a service's attributes and actions.

RECORD MANAGER SERVICES

The following services are used in Record Management functions:

❖ ACTIVATE_SERVICE (page 10-4)

❖ APPROVE_DELETE_SERVICE (page 10-5)

❖ BROWSE_CATEGORY_FORM (page 10-5)

❖ BROWSE_FOLDER_FORM (page 10-5)

❖ BROWSE_SERIES_FORM (page 10-5)

❖ CHECKIN_NEW_REVISION_SERVICE (page 10-5)

❖ CHECKIN_SIMILAR_FORM (page 10-6)

❖ CLEAR_FOLDER_CANCELLED_DATE (page 10-6)

❖ CLEAR_FOLDER_EXPIRATION_DATE (page 10-6)

❖ CLEAR_FOLDER_OSBOLETE_DATE (page 10-6)

❖ CLEAR_FOLDER_RESCINDED_DATE (page 10-6)

❖ CLEAR_FOLDER_REVIEW_DATE (page 10-7)

❖ CLEAR_RECORD_CANCELLED_DATE (page 10-7)

- CLEAR_RECORD_EXPIRATION_DATE (page 10-7)
- CLEAR_RECORD_OBSOLETE_DATE (page 10-7)
- CLOSE_FOLDER (page 10-7)
- CREATE_FOLDER (page 10-8)
- CREATE_FOLDER_FORM (page 10-8)
- DELETE_ALL_BUT_LAST_N_REVISIONS_SERVICE (page 10-8)
- DELETE_FOLDER (page 10-9)
- DELETE_REVISION_SERVICE (page 10-9)
- EDIT_FOLDER (page 10-9)
- EDIT_FOLDER_FORM (page 10-9)
- FREEZE_FOLDER (page 10-10)
- FREEZE_RECORD (page 10-10)
- INFO_CATEGORY_FORM (page 10-10)
- INFO_FOLDER_FORM (page 10-11)
- INFO_FOLDER_LIFECYCLE (page 10-11)
- INFO_FOLDER_METADATA_HISTORY (page 10-11)
- INFO_FOLDER_REVIEW_HISTORY (page 10-11)
- INFO_RECORD_LIFECYCLE (page 10-11)
- INFO_RECORD_METADATA_HISTORY (page 10-12)
- INFO_RECORD_REVIEW_HISTORY (page 10-12)
- INFO_SERIES_FORM (page 10-12)
- MARK_FOLDER_ACTIVATION_DATE (page 10-12)
- MARK_FOLDER_CANCELLED_DATE (page 10-12)
- MARK_FOLDER_EXPIRATION_DATE (page 10-13)
- MARK_FOLDER_OBSOLETE_DATE (page 10-13)
- MARK_FOLDER_RESCINDED_DATE (page 10-13)
- MARK_FOLDER_REVIEW_DATE (page 10-13)
- MARK_FOLDER_REVIEW_DATE_RECURSIVE (page 10-14)

- MARK_RECORD_CANCELLED_DATE (page 10-14)
- MARK_RECORD_EXPIRATION_DATE (page 10-14)
- MARK_RECORD_OBSOLETE_DATE (page 10-14)
- MARK_RECORD_RESCINDED_DATE (page 10-15)
- MARK_RECORD_REVIEW_DATE (page 10-15)
- MOVE_FOLDER (page 10-15)
- PREVIEW_RECORD_LIFECYCLE (page 10-15)
- RMA_CLOSE_SERVICE (page 10-16)
- RMA_CUTOFF_SERVICE (page 10-16)
- RMA_DESTROY_SERVICE (page 10-16)
- RMA_EXPORT_ARCHIVE_SERVICE (page 10-16)
- RMA_EXPORT_SERVICE (page 10-17)
- RMA_MARK_COMPLETED (page 10-17)
- RMA_NO_ACTION_SERVICE (page 10-17)
- RMA_OBSOLETE_SERVICE (page 10-17)
- RMA_SCRUB_SERVICE (page 10-18)
- RMA_SUPERSEDE_SERVICE (page 10-18)
- UNCLOSE_FOLDER (page 10-18)
- UNFREEZE_FOLDER (page 10-18)
- UNFREEZE_RECORD (page 10-19)

ACTIVATE_SERVICE

Service used to set an Activate disposition action date.

Additional Required Service Parameters

- dDispositionID: The disposition action to be processed.

APPROVE_DELETE_SERVICE

Service used to an Approve Delete disposition action date.

Additional Required Service Parameters

- dDispositionID: The disposition action to be processed.

BROWSE_CATEGORY_FORM

Service that retrieves a page used to browse he contents of a records category in the records file plan.

Additional Required Service Parameters

- dCategoryID: The records category to be retrieved.

BROWSE_FOLDER_FORM

Service that retrieves a page used to browse the contents of a records folder in the records file plan.

Additional Required Service Parameters

- dFolderID: The records folder to be retrieved.

BROWSE_SERIES_FORM

Service that retrieves a page used to browse the contents of a records series in the records file plan.

Additional Required Service Parameters

- dSeriesID: The records series to be retrieved.

CHECKIN_NEW_REVISION_SERVICE

Service used to check in the latest revision of a record as a new revision. This is usually used to trigger a workflow.

CHECKIN_SIMILAR_FORM

Service used to retrieve the Check In Similar page, a checkin page with metadata fields already filled in based on the previous content item checked in.

Additional Required Service Parameters

- dID: The content ID for the record.

CLEAR_FOLDER_CANCELLED_DATE

Service that clears the date for a cancelled records folder.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.

CLEAR_FOLDER_EXPIRATION_DATE

Service that clears the expiration date for a cancelled records folder.

Additional Required Service Parameters

- dFolderID: The folder to be used.

CLEAR_FOLDER_OSBOLETE_DATE

Service that clears the date at which a folder becomes obsolete.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.

CLEAR_FOLDER_RESCINDED_DATE

Service that clears the date at which a folder is rescinded.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.

CLEAR_FOLDER_REVIEW_DATE

Service that clears the date at which a folder was last reviewed.

Additional Required Service Parameters

❖ dFolderID: The folder ID to be used.

CLEAR_RECORD_CANCELLED_DATE

Service that clears the date at which a record was cancelled.

Additional Required Service Parameters

❖ dID: The record ID to be used.

CLEAR_RECORD_EXPIRATION_DATE

Service that clears the date at which a record was expired.

Additional Required Service Parameters

❖ dID: The record ID to be used.

CLEAR_RECORD_OBSOLETE_DATE

Service that clears the date at which a record becomes obsolete.

Additional Required Service Parameters

❖ dID: The record ID to be used.

CLOSE_FOLDER

Service that closes an existing records folder in the records file plan. No further records can be checked into the closed records folder or its subfolders unless the user has the Folder Open/Close right or is the author of the closed folder.

Additional Required Service Parameters

❖ dFolderID: The folder ID to be used.

See Also

– *RMA_CLOSE_SERVICE (page 10-16)*

CREATE_FOLDER

Service that creates a new records folder in the records file plan. This service is usually called from the CREATE_FOLDER_FORM page.

Additional Required Service Parameters

❖ dFolderID: The ID for the new folder.

❖ dFolderName: A name for the new folder.

❖ dSecurityGroup: The security group to be associated with the folder.

❖ dDocAuthor: The person creating the folder.

❖ dCategoryID: The folder category.

See Also

– *CREATE_FOLDER_FORM (page 10-8)*

CREATE_FOLDER_FORM

Service that calls CREATE_FOLDER to create a new records folder in the records file plan.

See Also

– *CREATE_FOLDER (page 10-8)*

DELETE_ALL_BUT_LAST_N_REVISIONS_SERVICE

Service used to process a Delete All But Last *N* Revisions disposition action.The records administrator sets *n,* which is a variable.

Additional Required Service Parameters

❖ dDispositionID: The disposition action to be processed.

DELETE_FOLDER

Service that deletes an existing records folder from the records file plan.

Additional Required Service Parameters

❖ dFolderID: The ID for the new folder.

DELETE_REVISION_SERVICE

Service that deletes the latest revision of a record.

Additional Required Service Parameters

❖ dDispositionID: The disposition action to be processed.

EDIT_FOLDER

Service called by EDIT_FOLDER_FORM to update the (modified) properties of an existing records folder in the records file plan.

Additional Required Service Parameters

❖ dFolderID: The folder to be edited.

❖ dFolderName: The name of the folder to be edited.

❖ dSecurityGroup: The security group associated with the folder.

❖ dDocAuthor: The folder owner.

See Also

– *EDIT_FOLDER_FORM (page 10-9)*

EDIT_FOLDER_FORM

Service that calls EDIT_FOLDER to edit a folder in the records file plan.

Additional Required Service Parameters

- dFolderID: The folder to be edited.

See Also

– *EDIT_FOLDER (page 10-9)*

FREEZE_FOLDER

Service that freezes an existing records folder in the records file plan. Freezing a folder pauses any processing of disposition rules associated with the folder until an UNFREEZE service is called.

Additional Required Service Parameters

- dFolderID: The folder to be frozen.
- dCategoryID: The folder category.
- auditComments: The freeze name.
- auditComments2: The freeze reason.

FREEZE_RECORD

Service that freezes an existing record. Freezing a record pauses any processing of disposition rules associated with the record until an UNFREEZE service is called.

Additional Required Service Parameters

- dID: The content ID of the record to be frozen.
- auditComments: The freeze name.
- auditComments2: The freeze reason.

INFO_CATEGORY_FORM

Service that retrieves the information page of a records category.

Additional Required Service Parameters

- dCategoryID: The records category.

INFO_FOLDER_FORM

Service that retrieves the metadata information page and related content for a records folder in the records file plan.

Additional Required Service Parameters

- dFolderID: The ID for the folder to be retrieved.

INFO_FOLDER_LIFECYCLE

Service that retrieves the life cycle information page for a records folder.

Additional Required Service Parameters

- dFolderID: The ID for the folder to be retrieved.

INFO_FOLDER_METADATA_HISTORY

Service that retrieves the metadata history of a folder.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.

INFO_FOLDER_REVIEW_HISTORY

Service that retrieves the review history of a records folder.

Additional Required Service Parameters

- dFolderID: The folder to be reviewed.

INFO_RECORD_LIFECYCLE

Service that retrieves the life cycle information page for a record.

Additional Required Service Parameters

- dID: The content ID of the record.

- dCategoryID: The category ID for the record.

INFO_RECORD_METADATA_HISTORY

Service that retrieves the metadata history of a record.

Additional Required Service Parameters

- dID: The content ID of the record.

INFO_RECORD_REVIEW_HISTORY

Service that retrieves the review history of a record.

Additional Required Service Parameters

- dID: The content ID of the record.

INFO_SERIES_FORM

Service that retrieves the information page for a records series in the records file plan.

Additional Required Service Parameters

- dSeriesID: The series ID of the record series.

MARK_FOLDER_ACTIVATION_DATE

Service that sets the current date for when a records folder will be activated.

Additional Required Service Parameters

- dFolderID: The folder ID to be activated.
- currentDate: The date to be used for activation.

MARK_FOLDER_CANCELLED_DATE

Service that sets the current date for when a records folder will be cancelled.

Additional Required Service Parameters

- dFolderID: The folder ID to be cancelled.
- currentDate: The date to be used for cancellation.

MARK_FOLDER_EXPIRATION_DATE

Service that sets the current date for the expiration date for a records folder.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.
- currentDate: The date to be used for expiration.

MARK_FOLDER_OBSOLETE_DATE

Service that sets the current date for when a records folder becomes obsolete.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.
- currentDate: The date to be used as the obsolete date.

MARK_FOLDER_RESCINDED_DATE

Service that sets the current date for when a records folder will be rescinded.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.
- currentDate: The date to be used as the rescinded date.

MARK_FOLDER_REVIEW_DATE

Service that sets the current date for when a records folder will be reviewed.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.
- currentDate: The date to be used as the review date.

MARK_FOLDER_REVIEW_DATE_RECURSIVE

Service that sets the current date for when a records folder, including all of its child folders, will be reviewed.

Additional Required Service Parameters

- dFolderID: The folder ID to be used.
- currentDate: The date to be used as the review date.

MARK_RECORD_CANCELLED_DATE

Service that sets the current date for when a record will be cancelled.

Additional Required Service Parameters

- dID: The content ID of the record to be cancelled.
- currentDate: The date to be used as the cancellation date.

MARK_RECORD_EXPIRATION_DATE

Service that sets the current date for when a record will expire.

Additional Required Service Parameters

- dID: The content ID of the record to be used.
- currentDate: The date to be used as the expiration date.

MARK_RECORD_OBSOLETE_DATE

Service that sets the current date to be used to make a record obsolete.

Additional Required Service Parameters

- dID: The content ID of the record to be used.
- currentDate: The date to be used as the obsolete date.

MARK_RECORD_RESCINDED_DATE

Service that sets the current date for when a record will be rescinded.

Additional Required Service Parameters

- dID: The content ID of the record to be rescinded.
- currentDate: The date to be used as the rescinded date.

MARK_RECORD_REVIEW_DATE

Service that sets the current date for when a record will be reviewed.

Additional Required Service Parameters

- dID: The content ID of the record to be reviewed.
- currentDate: The date to be used as the review date.

MOVE_FOLDER

Service that moves a records folder from one location in the records file plan to another.

Additional Required Service Parameters

- dFolderID: The folder ID of the folder to be moved.
- dCategoryID: The folder category.

PREVIEW_RECORD_LIFECYCLE

Service that retrieves the life cycle preview of a record.

RMA_CLOSE_SERVICE

Service used to process a Close disposition action on a folder. This service calls the CLOSE_FOLDER service.

Additional Required Service Parameters

❖ dDispositionID: The disposition ID.

See Also

– *CLOSE_FOLDER (page 10-7)*

RMA_CUTOFF_SERVICE

Service used to process a Cutoff disposition action.

Additional Required Service Parameters

❖ dDispositionID: The disposition ID.

RMA_DESTROY_SERVICE

Service used to process a Destroy disposition action. All revisions of records or folders are destroyed unless other disposition rules take precedence.

Additional Required Service Parameters

❖ dDispositionID: The disposition ID.

RMA_EXPORT_ARCHIVE_SERVICE

Service used to add a Records Manager archive to a Content Server archive.

Additional Required Service Parameters

❖ dDispositionID: The disposition ID.

RMA_EXPORT_SERVICE

Service used to create a zip archive of records and folders.

Additional Required Service Parameters

❖ dDispositionID: The disposition ID.

RMA_MARK_COMPLETED

Service used to mark the completion of an action.

Additional Required Service Parameters

❖ dDispositionID: The disposition ID.

RMA_NO_ACTION_SERVICE

Service used to process a No Action disposition action.

Additional Required Service Parameters

❖ dDispositionID: The disposition ID.

RMA_OBSOLETE_SERVICE

Service used to process an Obsolete disposition action. This service calls either MARK_FOLDER_OBSOLETE_DATE or MARK_RECORD_OBSOLETE_DATE.

Additional Required Service Parameters

❖ dDispositionID: The disposition ID.

See Also

– *MARK_FOLDER_OBSOLETE_DATE (page 10-13)*

– *MARK_RECORD_OBSOLETE_DATE (page 10-14)*

RMA_SCRUB_SERVICE

Service used to overwrite folders and records so they cannot be recovered.

Additional Required Service Parameters

- dDispositionID: The disposition ID.

RMA_SUPERSEDE_SERVICE

Service used to process a Supersede disposition action.

Additional Required Service Parameters

- dDispositionID: The disposition ID.

UNCLOSE_FOLDER

Service that revokes the closed status of an existing records folder in the records file plan. Unclosing a folder allows records to be checked into the closed records folder or its subfolders.

Additional Required Service Parameters

- dFolderID: The ID of the folder to be unclosed.

UNFREEZE_FOLDER

Service that unfreezes an existing records folder in the records file plan. Unfreezing a folder releases the folder for processing of its associated disposition rules.

Additional Required Service Parameters

- dFolderID: The folder to be unfrozen.
- auditComments: The unfreeze name.
- auditComments2: The unfreeze reason.

UNFREEZE_RECORD

Service that unfreezes an existing record. Unfreezing a record releases the record for processing of its associated disposition rules.

Additional Required Service Parameters

- dFolderID: The record to be unfrozen.
- auditComments: The unfreeze name.
- auditComments2: The unfreeze reason.

Appendix

A

ACTIONS

OVERVIEW

This section describes the actions used by services. An *action* is an operation to be performed as part of a service script. Actions can execute SQL statements, perform a query, run code, cache the results of a query, or load an option list.

ABOUT SERVICE ACTIONS

An action is defined as a list of colon-separated segments, using the following format:

```
type:name:parameters:control mask:error message
```

The action type can be designated by a number or a descriptive name:

- 1: Select query. This executes a database query to retrieve information (read-only action) then discards the results.
- 2: Execute query. This executes a database query to delete, add, or update information in the database.
- 3: Java method. This specifies a code module that is a part of the Java class implementing the service.
- 4: Load option list. This loads an option list stored in the system.
- 5: Select cache query. This executes a database query to retrieve information (read-only action) and stores the results for later use.

See Actions (page 4-12) for details about actions, their parameters, and control masks.

A

addAliases

Adds the aliases and passes *IworkflowAlias* as a parameter. Called as a Java method by ADD_WORKFLOWALIASES (page 6-24).

addCollaboration

Called as a Java method by ADD_COLLABORATION (page 2-3).

addContext

Called as a Java method by ADD_WEB_APP (page 5-118).

addFiles:

Checks in the content item by name. Called as a Java method by these services:

- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)
- REPLACE_METAFILE_SUB (page 5-82)
- UPDATE_BYREV (page 5-101)
- WORKFLOW_CHECKIN_SUB (page 6-11)

addOrEditDocMetaData

Called as a Java method by ADD_METADEF (page 5-136) and EDIT_METADEF (page 5-138).

addOrEditProvider

Adds the new provider. Called as a Java method by ADD_EDIT_PROVIDER (page 5-157)

addOrEditSchemaTable

Called as a Java method by ADDOREDIT_SCHEMA_TABLE (page 5-164).

addProblemReport

Adds the problem report to the database. Called as a Java method by ADD_PROBLEMREPORT (page 6-3).

addRegisteredUserAttribute

Retrieves the user database profile information. Called as a Java method by REGISTER_USER (page 5-204).

addSchemaRelation

Called as a Java method by ADD_SCHEMA_RELATION (page 5-164).

addSchemaView

Called as a Java method by ADD_SCHEMA_VIEW (page 5-164).

addSubscription

Adds the subscription and passes *Isubscription* as a parameter. Called as a Java method by SUBSCRIBE (page 5-86).

addTemplate

Called as a Java method by ADD_WF_TEMPLATE (page 6-13).

addUserAttributes

Adds/updates the user security attributes and passes *IuserSecurityAttribute* as a parameter. Called as a Java method by ADD_USER (page 5-182) and EDIT_USER (page 5-195).

addWfDocuments

Adds the workflow content items and passes *ADD_WORKFLOWDOCUMENT_SUB* as a parameter. The action throws a data exception if the service is unable to add content item revisions. Called as a Java method by ADD_WORKFLOWDOCUMENTS (page 6-25).

addWorkflow

Adds the workflow. Called as a Java method by ADD_WORKFLOW (page 6-22).

addWorkflowScript

Called as a Java method by ADD_WORKFLOW_SCRIPT (page 6-23).

addWorkflowStep

Adds the workflow steps and passes *IworkflowStep* as a parameter The action throws a data exception if the service is unable to add the step to the workflow. Called as a Java method by ADD_WORKFLOWSTEP (page 6-26).

addWorkflowStepScript

Called as a Java method by ADD_WORKFLOWSTEP (page 6-26).

addWorkflowToken

Called as a Java method by ADD_WORKFLOW_TOKEN (page 6-23).

Alias

Retrieves alias information. The result of this query is assigned to the parameter *Alias* and stored for later use. Called as a Select Cache Query action by GET_ALIASES (page 5-200).

AliasUserMap

Retrieves the user alias map. The result of this query is assigned to the parameter *AliasUserMap* and stored for later use. Called as a Select Cache Query action by GET_ALIASES (page 5-200).

allowProblemReportAction

Allows the problem report action to execute and passes *update* as a parameter. Called as a Java method by DELETE_PROBLEMREPORT (page 6-4) and UPDATE_PROBLEMREPORT (page 6-10).

appendCommonSystemInfo

Called as a Java method by GET_SYSTEM_AUDIT_INFO (page 5-13).

appendDatabaseAuditMessage

Called as a Java method by APPEND_DATABASE_AUDIT_INFO (page 5-158).

appendFileCachingMessage

Called as a Java method by APPEND_FILE_CACHING_INFO (page 5-118).

approveDo

Approves the content item for the workflow. Called as a Java method by WORKFLOW_APPROVE (page 6-45).

B

buildAllWebStringFiles

Called as a Java method by LM_BUILD_WEB_STRING_FILES (page 5-14).

buildExpiredContentQuery

Called as a Java method by GET_EXPIRED (page 5-78).

buildPreviewList

Called as a Java method by these services:

- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- GET_PREVIEW_LIST (page 9-5)
- GET_PREVIEW_LIST_EX (page 9-5)

buildSearchIndex

Called as a Java method by START_SEARCH_INDEX (page 5-132).

buildSourceInfo

Builds the source information. Called as a Java method by PROBLEMREPORT_INFO (page 6-9).

C

cacheCheckin

Called as a Java method by these services:

- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)

cacheTemplates

Caches templates. Called as a Java method by these services:

- ADD_WF_TEMPLATE (page 6-13)
- DELETE_WF_TEMPLATE (page 6-13)
- EDIT_WF_TEMPLATE (page 6-14)

cancelComponentInstall

Cancels the installation of a component. Called as a Java method by CANCEL_COMPONENT_INSTALL (page 5-134).

cancelCriteriaWorkflow

Cancels the criteria workflow and passes *WfDocuments* and *Qdocuments* as parameters. Called as a Java method by these services:

- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)
- DELETE_WORKFLOWCRITERIA (page 6-30)

cancelSearchIndex

Cancels the search indexing session. Called as a Java method by CANCEL_SEARCH_INDEX (page 5-124).

cancelWorkflow

Cancels the workflow and passes *WfDocuments* and *Qdocuments* as parameters. Called as a Java method by DELETE_WORKFLOW (page 6-29) and WORKFLOW_CANCEL (page 6-45).

canDeleteStep

Evaluates whether a workflow step can be deleted. Called as a Java method by DELETE_WORKFLOWSTEP (page 6-32).

checkCanCreateDocSecurity

Evaluates the assigned security level to verify that the user is authorized to perform this action. Called as a Java method by these services:

- CHECKIN_SEL_FORM (page 5-45)
- CHECKOUT_SUB (page 5-57)
- UNDO_CHECKOUT (page 5-92)
- UNDO_CHECKOUT_BY_NAME (page 5-95)

checkCollaborationAccess

Called as a Java method by these services:

- ADD_COLLABORATION (page 2-3)
- ADD_COLLABORATION_FORM (page 2-3)
- DELETE_COLLABORATION (page 2-4)
- EDIT_CLBRA_ACCESS_LIST (page 2-4)
- EDIT_CLBRA_ACCESS_LIST_FORM (page 2-4)
- EDIT_COLLABORATION (page 2-4)
- EDIT_COLLABORATION_FORM (page 2-5)
- GET_CLBRA_DOCUMENTS (page 2-5)
- GET_CLBRA_INFO (page 2-5)

checkConversionCache

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB (page 5-120).

checkCounters

Called as a Java method by INSERT_NATIVE (page 7-27).

checkCriteriaWorkflow

Evaluates the criteria workflow. Called as a Java method by these services:

- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)

checkDocRules

Evaluates the defined content item rules and passes *checkout* and *isNotPublished* as parameters. Called as a Java method by these services:

- CACHE_CHECKIN_NEW (page 5-24)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_FORM (page 5-45)
- CHECKOUT_BY_NAME (page 5-55)
- CHECKOUT_SUB (page 5-57)
- GET_UPDATE_FORM (page 5-80)
- UNDO_CHECKOUT (page 5-92)

- UNDO_CHECKOUT_BY_NAME (page 5-95)
- UPDATE_DOCINFO_SUB (page 5-110)

checkDocState

Called as a Java method by ADD_WORKFLOWDOCUMENT_SUB (page 6-25).

checkForceLogin

Forces a log in. Called as a Java method by these services:

- GET_CLBRA_DOCUMENTS (page 2-5)
- LOGIN (page 5-15)
- PING_SERVER (page 5-19)
- UPDATE_SUBSCRIPTION_NOTIFY (page 5-110)
- UPDATE_SUBSCRIPTION_USED (page 5-111)

checkForPublish

Called as a Java method by UPDATE_BYREV (page 5-101).

checkForRefreshingCachedResources

Called as a Java method by LOAD_RESOURCE_FILE (page 5-122).

checkInByID

Called as a Java method by CACHE_CHECKIN_SEL (page 5-24) and CHECKIN_SEL_SUB (page 5-47).

checkInByName

Checks in the content item by name. Called as a Java method by CHECKIN_BYNAME (page 5-26).

checkIsLatestRev

Evaluates for the latest revision. Called as a Java method by CHECKOUT_BY_NAME (page 5-55) and RESUBMIT_FOR_CONVERSION (page 5-82).

checkIsSelf

Checks whether the user is editing self. Called as a Java method byEDIT_USER_PROFILE (page 5-198).

checkParametersAgainstResultSet

Called as a Java method by DELETE_DOC (page 5-61).

checkProblemReportSecurity

Retrieves the data assigned to the parameter *DOC_INFO* and evaluates the problem report security information. Called as a Java method by these services:

- DELETE_PROBLEMREPORT (page 6-4)
- GET_UPDATE_PROBLEMREPORT_FORM (page 6-7)
- PROBLEMREPORT_INFO (page 6-9)
- RESEND_PROBLEMREPORT (page 6-9)
- UPDATE_PROBLEMREPORT (page 6-10)

checkRefreshUsers

Updates the user information. Called as a Java method by these services:

- ADD_USER (page 5-182)
- CHANGE_USER_AUTH_TYPE (page 5-184)
- DELETE_USER (page 5-191)
- EDIT_USER (page 5-195)
- EDIT_USER_PROFILE (page 5-198)
- REGISTER_USER (page 5-204)

checkRevisionProperties

Retrieves the latest revision information and passes *Qrevisions* as a parameter. Called as a Java method by these services:

- CHECKIN_SEL_FORM (page 5-45)
- CHECKOUT_OK (page 5-56)
- CHECKOUT_SUB (page 5-57)
- GET_UPDATE_FORM (page 5-80)

checkSecurity

Retrieves the data assigned to the parameter *DOC_INFO* and evaluates the assigned security level to verify that the user is authorized to perform this action. The parameter varies by service. Called as a Java method by these services:

- ADD_PROBLEMREPORT (page 6-3)

- ADD_WORKFLOW (page 6-22)
- ADD_WORKFLOWALIASES (page 6-24)
- ADD_WORKFLOWDOCUMENT (page 6-24)
- ADD_WORKFLOWDOCUMENTS (page 6-25)
- APPLET_DOCINFO (page 5-6)
- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW (page 5-31)
- CHECKIN_SEL (page 5-40)
- CHECKIN_UNIVERSAL (page 5-48)
- CONTINUE_CHECKIN (page 5-59)
- CONTINUE_SUBMIT_HTML_FORM (page 5-59)
- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_ENABLE (page 6-28)
- DELETE_CHECKIN_CACHE (page 5-61)
- DELETE_DOC (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)
- DELETE_WFCONTRIBUTORS (page 6-28)
- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOWCRITERIA (page 6-30)
- DELETE_WORKFLOWDOCUMENTS (page 6-30)
- DOC_INFO (page 5-64)
- EDIT_WORKFLOW (page 6-32)
- EDIT_WORKFLOWCRITERIA (page 6-34)
- FORM_PROCESS (page 5-76)
- GET_ARCHIVED_FILE (page 7-17)
- GET_DOCUMENT_PROBLEMREPORTS (page 6-5)

- GET_DYNAMIC_CONVERSION (page 5-118)
- GET_EXTERNAL_DOC_INFO (page 5-170)
- GET_EXTERNAL_HIGHLIGHT_INFO (page 5-171)
- GET_EXTERNAL_XML_HIGHLIGHT_INFO (page 5-171)
- GET_FILE (page 5-129)
- GET_HIGHLIGHT_INFO (page 5-172)
- GET_PREVIEW_LIST (page 9-5)
- GET_PREVIEW_LIST_EX (page 9-5)
- GET_UPDATE_FORM (page 5-80)
- GET_WF_COMPANION_INFO (page 6-38)
- GET_WORKFLOW_INFO (page 6-39)
- GET_WORKFLOW_INFO_BYNAME (page 6-40)
- GET_WORKFLOWDOCREVISIONS (page 6-42)
- GET_WORKFLOWDOCUMENTS (page 6-7)
- GET_XML_HIGHLIGHT_INFO (page 5-175)
- LOAD_RESOURCE_FILE (page 5-122)
- RESUBMIT_FOR_CONVERSION (page 5-82)
- REV_HISTORY (page 5-84)
- REVIEW_WORKFLOW_DOC (page 6-10)
- SELECTDOC (page 5-84)
- SUBMIT_HTML_FORM (page 5-85)
- SUBSCRIBE (page 5-86)
- UNSUBSCRIBE (page 5-95)
- UPDATE_DOCINFO_SUB (page 5-110)
- VALIDATE_DOCINFO (page 5-112)
- VIEW_DOC (page 5-177)
- WORKFLOW_APPROVE (page 6-45)
- WORKFLOW_CANCEL (page 6-45)
- WORKFLOW_CHECKIN (page 6-45)
- WORKFLOW_REJECT (page 6-50)
- WORKFLOW_REJECT_FORM (page 6-12)

- WORKFLOW_START (page 6-53)

checkSubAdmin

Checks if the user has the sub administrator role and passes a parameter, which varies depending which service is used. Called as a Java method by the following services:

- ADD_USER (page 5-182)
- ADD_WORKFLOW (page 6-22)
- ADD_WORKFLOW_SCRIPT (page 6-23)
- ADD_WORKFLOW_TOKEN (page 6-23)
- ADD_WORKFLOWALIASES (page 6-24)
- ADD_WORKFLOWDOCUMENT (page 6-24)
- ADD_WORKFLOWDOCUMENTS (page 6-25)
- ADD_WORKFLOWSTEP (page 6-26)
- CHANGE_USER_AUTH_TYPE (page 5-184)
- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_ENABLE (page 6-28)
- DELETE_USER (page 5-191)
- DELETE_WFCONTRIBUTORS (page 6-28)
- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOW_SCRIPT (page 6-29)
- DELETE_WORKFLOW_TOKEN (page 6-30)
- DELETE_WORKFLOWCRITERIA (page 6-30)
- DELETE_WORKFLOWDOCUMENTS (page 6-30)
- DELETE_WORKFLOWSTEP (page 6-32)
- EDIT_USER (page 5-195)
- EDIT_WORKFLOW (page 6-32)
- EDIT_WORKFLOW_SCRIPT (page 6-33)
- EDIT_WORKFLOW_TOKEN (page 6-34)
- EDIT_WORKFLOWCRITERIA (page 6-34)
- EDIT_WORKFLOWSTEP (page 6-35)
- GET_ADMIN_PAGE (page 5-145)
- GET_CRITERIA_WORKFLOWS_FOR_GROUP (page 6-5)

- GET_PROJECT_REGISTRATION (page 9-6)
- GET_PROJECT_REGISTRATION (page 9-6)
- GET_REGISTEREDPROJECTS (page 9-7)
- GET_USERS (page 5-202)
- GET_WF_COMPANION_INFO (page 6-38)
- GET_WORKFLOW_SCRIPT (page 6-42)
- GET_WORKFLOWS_FOR_ALL (page 6-8)
- PAGE_HANDLER (page 5-153)
- QUERY_USER_ATTRIBUTES (page 5-203)
- REGISTER_PROJECT (page 9-7)
- UNREGISTER_PROJECT (page 9-8)
- UPDATE_USEROPTION_LIST (page 5-209)
- WORKFLOW_CANCEL (page 6-45)
- WORKFLOW_START (page 6-53)

checkUserAuthType

Called as a Java method by CHANGE_USER_AUTH_TYPE (page 5-184).

checkWorkflow

Checks workflow information by referencing *WF_INFO* and passes *isNotActiveBasic* as a parameter. Called as a Java method by these services:

- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_BYNAME (page 5-26)
- CHECKIN_SEL_SUB (page 5-47)
- CHECKOUT_SUB (page 5-57)
- DELETE_DOC (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)
- WORKFLOW_CHECKIN_SUB (page 6-11)
- WORKFLOW_START (page 6-53)

checkWorkflowAdminAccess

Called as a Java method by GET_WORKFLOWS (page 6-44).

computeCompanionDirectory

Called as a Java method by ADD_WORKFLOWDOCUMENT_SUB (page 6-25).

computeDeleteSecurity

Called as a Java method by DELETE_REV (page 5-62).

computeDocID

Computes the generated content item revision ID. Called as a Java method by these services:

- GET_ARCHIVED_FILE (page 7-17)
- GET_DYNAMIC_CONVERSION (page 5-118)
- GET_FILE (page 5-129)
- LOAD_RESOURCE_FILE (page 5-122)

computeDocInfoInHtmlPage

Evaluates the content information for the HTML page. Parameters vary by service. Called as a Java method by these services:

- CHECKIN_LIST (page 5-30)
- CONTINUE_CHECKIN (page 5-59)
- DELETE_CHECKIN_CACHE (page 5-61)
- DOC_SUBS_LIST (page 5-73)
- GET_EXPIRED (page 5-78)
- SUBSCRIPTION_LIST (page 5-91)
- UNSUBSCRIBE_FROM_LIST (page 5-98)
- WORK_IN_PROGRESS (page 5-112)

computeDocSubscribers

Evaluates the content item for subscribers. Called as a Java method by GET_DOC_SUBSCRIBERS (page 5-78).

computeProjectInfo

Computes project information. Called as a Java method by GET_PROJECT_REGISTRATION (page 9-6).

computeRejectTargetStep

Computes the reject target step. Called as a Java method by WORKFLOW_REJECT (page 6-50).

computeRemainingUsers

Called as a Java method by these services:

- GET_WORKFLOW_INFO (page 6-39)
- GET_WORKFLOW_INFO_BYNAME (page 6-40)
- WORKFLOW_REJECT (page 6-50)

computeURLs

Evaluates URLS and passes *REVISIONS* as a parameter. Called as a Java method by REV_HISTORY (page 5-84).

computeWfDocumentsInfo

Evaluates the workflow content item information and passes *WfDocuments* as a parameter. Called as a Java method by GET_WORKFLOWDOCUMENTS (page 6-7).

conditionalDetermineCheckin

Prepares the form for check in. Called as a Java method by FORM_PROCESS (page 5-76).

continueCheckin

Called as a Java method by CONTINUE_CHECKIN (page 5-59).

controlIndexing

Executes the search indexing control. Called as a Java method by CONTROL_SEARCH_INDEX (page 5-125).

createAddEditProviderForm

Creates an Add/Edit provider form. Called as a Java method by GET_ADD_EDIT_PROVIDER_FORM (page 5-160).

createArchiveFileName

Creates the archive file name. Called as a Java method by GET_ARCHIVED_FILE (page 7-17).

createFileName

Creates a new file name. Called as a Java method by GET_FILE (page 5-129) and LOAD_RESOURCE_FILE (page 5-122).

createNewRev

Called as a Java method by ADD_WORKFLOWDOCUMENT_SUB (page 6-25).

createResultSetSQL

Executes a query with parameters taken from the Data Binder (*dataSource* and *whereClause* local data) rather than from given parameters. It places the results in the local data using the ResultSet name found in the Data Binder (*resultName*). Called as a Java method by these services:

- CHECKIN_LIST (page 5-30)
- CONTINUE_CHECKIN (page 5-59)
- DELETE_CHECKIN_CACHE (page 5-61)
- GET_ACTIVE_WORKFLOWS (page 6-37)
- DOC_SUBS_LIST (page 5-73)
- GET_CLBRA_DOCUMENTS (page 2-5)
- GET_DATARESULTSET (page 5-10)
- GET_EXPIRED (page 5-78)
- GET_WORKFLOWS (page 6-44)
- SUBSCRIBE_EX (page 5-89)
- UNSUBSCRIBE_FROM_LIST_EX (page 5-100)
- WORK_IN_PROGRESS (page 5-112)

createSubscriptionType

Creates the subscription type. Called as a Java method by CREATE_SUBSCRIPTION_TYPE (page 5-60).

createWebFileNameFromRelativeUrl

Creates a web file name from the provided relative URL and passes *TEMPLATE_URL_INFO* as a parameter. Called as a Java method by GET_DYNAMIC_URL (page 5-121).

createWorkflowID

Creates the workflow label. Called as a Java method by ADD_WORKFLOW (page 6-22).

D

Dalias

Deletes the alias. Called as an Execute Query action by DELETE_ALIAS (page 5-189).

DaliasUsers

Deletes the alias user. Called as an Execute Query action by DELETE_ALIAS (page 5-189) and EDIT_ALIAS (page 5-191).

Dcollaboration

Called as an Execute Query action by DELETE_COLLABORATION (page 2-4).

DdocAccount

Deletes the content item account. Called as an Execute Query action by DELETE_DOC_ACCOUNT (page 5-7).

DdocFormat

Deletes the content item format. Called as an Execute Query action by DELETE_DOCFORMAT (page 5-8).

DdocType

Deletes the content item type. Called as an Execute Query action by DELETE_DOCTYPE (page 5-8).

Ddocument

Called as an Execute Query action by these services:

- REMOVE_METAFILE_SUB (page 5-81)

- UPDATE_BYREV (page 5-101)
- WORKFLOW_CHECKIN_SUB (page 6-11)

decodeTopicValues

Decodes the topic values. Called as a Java method by PNE_SAVE_QUERY (page 5-154).

deleteAliases

Called as a Java method by DELETE_WFCONTRIBUTORS (page 6-28).

deleteCollaboration

Called as a Java method by DELETE_COLLABORATION (page 2-4).

deleteDoc

Retrieves the data assigned to the parameter *REVISIONS* and deletes the content item. Called as a Java method by DELETE_BYCLASS (page 5-60) and DELETE_BYNAME (page 5-60).

deleteDocumentSubscription

Deletes the content item subscription and passes *deleteRev* as a parameter. Called as a Java method by these services:

- DELETE_BYCLASS (page 5-60)
- DELETE_BYNAME (page 5-60)
- DELETE_BYREV (page 5-61)
- DELETE_DOC (page 5-61)
- DELETE_REV_EX (page 5-63)

deleteProblemReport

Deletes the problem report information. Called as a Java method by DELETE_PROBLEMREPORT (page 6-4).

deleteProvider

Deletes the provider. Called as a Java method by DELETE_PROVIDER (page 5-158).

deleteResultTemplate

Deletes the result template. Called as a Java method by DELETE_RESULT_TEMPLATE (page 5-145).

deleteRev

Deletes a previous revision of a content item and passes *DOC_INFO* as a parameter. Called as a Java method by DELETE_REV (page 5-62) and DELETE_REV_EX (page 5-63).

deleteRevFiles

Called as a Java method by these services:

- REMOVE_METAFILE_SUB (page 5-81)
- UPDATE_BYREV (page 5-101)
- WORKFLOW_CHECKIN_SUB (page 6-11)

deleteSchemaRelation

Called as a Java method by DELETE_SCHEMA_RELATION (page 5-165).

deleteSchemaTable

Called as a Java method by DELETE_SCHEMA_TABLE (page 5-165).

deleteSchemaView

Called as a Java method by DELETE_SCHEMA_VIEW (page 5-165).

deleteSubscriptionType

Action that deletes the subscription type. Called as a Java method by DELETE_SUBSCRIPTION_TYPE (page 5-64).

deleteTemplate

Deletes the workflow template. Called as a Java method by ADD_WF_TEMPLATE (page 6-13).

deleteWfDesign

Called as a Java method by DELETE_WORKFLOW (page 6-29) and DELETE_WORKFLOWCRITERIA (page 6-30).

deleteWfDocuments

Deletes workflow content items. Called as a Java method by DELETE_WORKFLOWDOCUMENTS (page 6-30).

deleteWorkflowScript

Deletes a workflow script. Called as a Java method by DELETE_WORKFLOW_SCRIPT (page 6-29).

deleteWorkflowStepScript

Deletes the script associated with the workflow step. Called as a Java method by DELETE_WORKFLOWSTEP (page 6-32).

deleteWorkflowToken

Deletes the workflow token. Called as a Java method by DELETE_WORKFLOW_TOKEN (page 6-30).

determineCheckin

Prepares the form for check in. Called as a Java method by these services:

- CHECKIN_UNIVERSAL (page 5-48)
- CONTINUE_SUBMIT_HTML_FORM (page 5-59)
- SUBMIT_HTML_FORM (page 5-85)

DextensionMap

Deletes the extension map. Called as an Execute Query action by DELETE_DOCEXTENSION (page 5-7).

DgroupRole

Deletes the role. Called as an Execute Query action by DELETE_GROUP (page 5-190).

disableSendFile

Called as a Java method by GET_DYNAMIC_CONVERSION (page 5-118) and GET_EXTERNAL_DYNAMIC_CONVERSION (page 5-122).

Dmetadef

Deletes the meta definition information. Called as an Execute Query action by DEL_METADEF (page 5-138).

doCachedCheckinCleanup

Called as a Java method by CONTINUE_CHECKIN (page 5-59) and DELETE_CHECKIN_CACHE (page 5-61).

doCachedCleanup

Called as a Java method by CONTINUE_SUBMIT_HTML_FORM (page 5-59).

DocFormats

Retrieves content item formats. The result of this query is assigned to the *DocFormats* parameter and stored for later use. Called as a Select Cache Query action by GET_DOCFORMATS (page 5-11).

docHistoryInfo

Evaluates content item history information and passes *Checkout* and *IdocHistory* as parameters. Called as a Java method by these services:

- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- CHECKOUT_SUB (page 5-57)
- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)
- CRITERIAWORKFLOW_ENABLE (page 6-28)
- DELETE_BYCLASS (page 5-60)
- DELETE_BYNAME (page 5-60)
- DELETE_BYREV (page 5-61)
- DELETE_DOC (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)
- UNDO_CHECKOUT (page 5-92)
- UNDO_CHECKOUT_BY_NAME (page 5-95)
- UPDATE_BYREV (page 5-101)
- UPDATE_DOCINFO_BYREV (page 5-109)
- UPDATE_DOCINFO_SUB (page 5-110)
- WORKFLOW_APPROVE (page 6-45)
- WORKFLOW_CANCEL (page 6-45)

- WORKFLOW_CHECKIN_SUB (page 6-11)
- WORKFLOW_REJECT (page 6-50)
- WORKFLOW_START (page 6-53)

docRefinery

Initiates the refinery check-in process. Called as a Java method by these services:

- CHECKIN_BYNAME (page 5-26)
- CHECKIN_SEL_SUB (page 5-47)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)
- REPLACE_METAFILE_SUB (page 5-82)
- UPDATE_BYREV (page 5-101)
- UPDATE_DOCINFO_SUB (page 5-110)
- WORKFLOW_CHECKIN_SUB (page 6-11)

doSubService

Executes the specified subservice. Called as a Java method by these services:

- ADD_WORKFLOWDOCUMENT (page 6-24)
- CHECKIN_NEW (page 5-31)
- CHECKIN_SEL (page 5-40)
- CHECKOUT (page 5-52)
- CHECKOUT_BY_NAME (page 5-55)
- GET_DOC_PAGE (page 5-146)
- GET_DYNAMIC_CONVERSION (page 5-118)
- GET_EXTERNAL_DYNAMIC_CONVERSION (page 5-122)
- GET_SECURE_PAGE (page 5-151)
- GET_SYSTEM_AUDIT_INFO (page 5-13)
- LM_LOAD_LAYOUTS (page 5-14)
- UPDATE_DOCINFO (page 5-102)
- UPDATE_DOCINFO_BYFORM (page 5-105)

- WORKFLOW_CHECKIN (page 6-45)

doSubserviceIfMetafile

Called as a Java method byUPDATE_DOCINFO_SUB (page 5-110).

doUpload

Called as a Java method by CHUNKED_UPLOAD (page 5-143).

downloadComponent

Called as a Java method by DOWNLOAD_COMPONENT (page 5-134).

downloadItems

Called as a Java method by DOWNLOAD_LISTBOX_ITEMS (page 5-144).

doWorkflowAction

Performs the workflow action. Called as a Java method by these services:

- DELETE_BYCLASS (page 5-60)
- DELETE_BYNAME (page 5-60)
- DELETE_BYREV (page 5-61)
- DELETE_DOC (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)

DprjDocumentsDeleted

Called as an Execute Query action by END_PUBLISHING (page 9-4).

DproblemReport

Deletes the problem report information. Called as an Execute Query action by DELETE_PROBLEMREPORT (page 6-4).

Drole

Deletes the role. Called as an Execute Query action by DELETE_ROLE (page 5-190).

DsecurityGroup

Deletes the security group. Called as an Execute Query action by DELETE_GROUP (page 5-190).

DsubscriptionType

Deletes the subscription type. Called as an Execute Query action by DELETE_SUBSCRIPTION_TYPE (page 5-64).

Duser

Called as an Execute Query action by DELETE_USER (page 5-191).

DuserSecurityAttributes

Deletes the user security attributes. Called as an Execute Query action by these services:

- ADD_USER (page 5-182)
- DELETE_ROLE (page 5-190)
- EDIT_USER (page 5-195)

DuserSubscription

Called as an Execute Query action by DELETE_USER (page 5-191).

Dworkflow

Deletes the workflow. Called as an Execute Query method by DELETE_WORKFLOW (page 6-29) and DELETE_WORKFLOWCRITERIA (page 6-30).

DworkflowAliasAll

Deletes the workflow aliases. Called as an Execute Query action by DELETE_WORKFLOW (page 6-29) and DELETE_WORKFLOWCRITERIA (page 6-30).

DworkflowCriteria

Deletes the workflow criteria. Called as an Execute Query action by DELETE_WORKFLOWCRITERIA (page 6-30).

DworkflowDocState

Deletes the workflow content item state. Called as an Execute Query action by WORKFLOW_REJECT (page 6-50).

DworkflowDocumentAll

Deletes the workflow content items. Called as an Execute Query by these services:

- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)

- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOWCRITERIA (page 6-30)

DworkflowStateAll

Deletes the workflow states. Called as an Execute Query action by these services:

- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)
- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOWCRITERIA (page 6-30)
- WORKFLOW_CANCEL (page 6-45)

DworkflowStep

Deletes the workflow steps. Called as an Execute Query action by DELETE_WORKFLOWSTEP (page 6-32).

DworkflowStepAliases

Deletes the workflow step aliases. Called as an Execute Query action by DELETE_WORKFLOWSTEP (page 6-32).

DworkflowStepsAll

Deletes the workflow steps. Called as an Execute Query action by DELETE_WORKFLOW (page 6-29) and DELETE_WORKFLOWCRITERIA (page 6-30).

DworkflowUserAttributeAll

Deletes the workflow user attributes. Called as an Execute Query action by these services:

- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)
- DELETE_WORKFLOWCRITERIA (page 6-30)

E

editCollaboration

Called as a Java method by EDIT_CLBRA_ACCESS_LIST (page 2-4) and EDIT_COLLABORATION (page 2-4).

editCriteria

Prepares the criteria for edit. Called as a Java method by EDIT_WORKFLOWCRITERIA (page 6-34).

editRole

Edits role information. Called as a Java method by EDIT_ROLE (page 5-194).

editSchemaRelation

Called as a Java method by EDIT_SCHEMA_RELATION (page 5-166).

editSchemaView

Called as a Java method by EDIT_SCHEMA_VIEW (page 5-166).

editSchemaViewValues

Called as a Java method by GET_SCHEMA_VIEW_VALUES (page 5-169).

editTemplate

Edits the workflow template. Called as a Java method by EDIT_WF_TEMPLATE (page 6-14).

editWorkflowScript

Called as a Java method by EDIT_WORKFLOW_SCRIPT (page 6-33).

editWorkflowStep

Edits the workflow step and passes *IworkflowStep* as a parameter. Called as a Java method by EDIT_WORKFLOWSTEP (page 6-35).

editWorkflowStepScript

Edits the script associated with the workflow step. Called as a Java method by EDIT_WORKFLOWSTEP (page 6-35).

editWorkflowToken

Called as a Java method by EDIT_WORKFLOW_TOKEN (page 6-34).

enableDisableProvider

Changes the provider enable state. Called as a Java method by ENABLE_DISABLE_PROVIDER (page 5-159).

executeArchiveMethod

Performs a specified action on an archive collection: add, delete, etc. Called as a Java method by these services:

- ADD_ARCHIVE (page 7-5)
- ADD_COLLECTION (page 7-5)
- ADD_PROXIEDCOLLECTION (page 7-6)
- CANCEL_ARCHIVE (page 7-6)
- COPY_ARCHIVE (page 7-11)
- DELETE_ARCHIVE (page 7-11)
- DELETE_BATCH_FILE (page 7-12)
- DELETE_BATCH_FILE_DOCUMENTS (page 7-12)
- EDIT_ARCHIVE (page 7-13)
- EDIT_ARCHIVEDATA (page 7-14)
- EDIT_EXPORTERS (page 7-14)
- EDIT_TRANSFEROPTIONS (page 7-15)
- EXPORT_ARCHIVE (page 7-15)
- GET_ARCHIVES (page 7-19)
- GET_BATCH_SCHEMA (page 7-20)
- GET_BATCH_VALUES (page 7-21)
- GET_BATCHFILES (page 7-21)
- GET_BATCH_FILE_DOCUMENTS (page 7-19)
- GET_REPLICATION_DATA (page 7-22)
- GET_TARGET_INFO (page 7-23)
- GET_TARGET_TRANSFER_STATUS (page 7-23)
- GET_TRANSFER_SOURCE_INFO (page 7-24)
- IMPORT_ARCHIVE (page 7-24)
- IMPORT_DOCUMENT (page 7-26)
- REGISTER_IMPORTER (page 7-28)
- REMOVE_COLLECTION (page 7-29)
- REMOVE_EXPORTER (page 7-29)
- REMOVE_IMPORTER (page 7-30)
- REMOVE_PROXIEDTRANSFER (page 7-30)

- REMOVE_TRANSFER (page 7-31)
- REQUEST_TRANSFER (page 7-31)
- TRANSFER_ARCHIVE (page 7-31)
- UPDATE_TARGET_TOTALS (page 7-32)
- UPDATE_TRANSFER_STATUS (page 7-32)
- UPLOAD_ARCHIVE_TRANSFER (page 7-32)

executeCommands

Called as a Java method by EXECUTE_BATCH (page 7-15).

executeManifest

Executes the manifest. Called as a Java method by UPLOAD_NEW_COMPONENT (page 5-135).

executePageService

Executes the page service. Called as a Java method by PAGE_HANDLER (page 5-153) and SAVE_GLOBALINCLUDES (page 5-155).

ExtensionFormatMap

Retrieves the extension format map. The result of this query is assigned to the *ExtensionFormatMap* parameter and stored for later use. Called as a Select Cache Query method by GET_DOCEXTENSIONS (page 5-11).

F

filterUpdateData

Filters the update information. Called as a Java method by EDIT_USER_PROFILE (page 5-198).

G

getCollaborationData

Called as a Java method by these services:

- EDIT_CLBRA_ACCESS_LIST_FORM (page 2-4)

- EDIT_COLLABORATION_FORM (page 2-5)
- GET_CLBRA_INFO (page 2-5)

getCollaborationDocs

Called as a Java method by GET_CLBRA_DOCUMENTS (page 2-5).

getCollaborations

Called as a Java method by GET_COLLABORATION_LIST (page 2-5).

getCompanionHistory

Called as a Java method by GET_WORKFLOW_INFO (page 6-39) and GET_WORKFLOW_INFO_BYNAME (page 6-40).

getComponentConfig

Called as a Java method by GET_COMPONENT_CONFIG (page 5-134).

getDefaultDocFormats

Sets the default content item format. Called as a Java method by DOC_FORMATS_WIZARD (page 5-64).

getDocFormats

Retrieves the file formats for the content item. Called as a Java method by these services:

- DOC_INFO (page 5-64)
- REVIEW_WORKFLOW_DOC (page 6-10)
- WORKFLOW_REJECT_FORM (page 6-12)

getDocSubscriptionInfo

Evaluates if the current user has subscribed to the content item and modifies the *DOC_INFO* page. Called as a Java method by these services:

- DOC_INFO (page 5-64)
- REVIEW_WORKFLOW_DOC (page 6-10)
- SUBSCRIBE_DOC_USER (page 5-88)
- SUBSCRIBE_FORM (page 5-90)
- UNSUBSCRIBE_FORM (page 5-97)

getDocumentList

Retrieves the content item list. Called as a Java method by DOC_SUBS_LIST (page 5-73).

getDynamicPage

Retrieves the dynamic page. Called as a Java method by GET_DYNAMIC_PAGE (page 5-149).

getExternalDocInfo

Retrieves the external content item information. Called as a Java method by GET_EXTERNAL_DOC_INFO (page 5-170).

getExternalSecurityInfo

Evaluates the assigned security level to verify that the user is authorized to perform this action. The action passes *SearchCollectionDocInfo* as a parameter. Called as a Java method by these services:

- GET_EXTERNAL_DOC_INFO (page 5-170)
- GET_EXTERNAL_HIGHLIGHT_INFO (page 5-171)
- GET_EXTERNAL_XML_HIGHLIGHT_INFO (page 5-171)
- VIEW_DOC (page 5-177)

getFilesInAppDir

Retrieves the files listed in the application directory. Called as a Java method by GET_FILELIST (page 5-12).

getHighlightInfo

Presents the content item highlight information. Called as a Java method by these services:

- GET_EXTERNAL_HIGHLIGHT_INFO (page 5-171)
- GET_EXTERNAL_XML_HIGHLIGHT_INFO (page 5-171)
- GET_HIGHLIGHT_INFO (page 5-172)
- GET_XML_HIGHLIGHT_INFO (page 5-175)

getLatestID

Passes *QlatestID* as a parameter. Called as a Java method by DELETE_REV (page 5-62).

getLatestIDRevInfo

Retrieves the latest revision information and passes *Qrevisions* as a parameter. Called as a Java method by these services:

- CHECKIN_SEL_FORM (page 5-45)
- CHECKOUT (page 5-52)
- CHECKOUT_BY_NAME (page 5-55)
- UNDO_CHECKOUT (page 5-92)
- UNDO_CHECKOUT_BY_NAME (page 5-95)

getOptionList

Retrieves the latest revision information and passes *Qrevisions* as a parameter. Called as a Java method by GET_OPTION_LIST (page 5-140) and UPDATE_OPTION_LIST (page 5-142).

getOutgoingProviders

Called as a Java method by GET_PROXIEDSERVERS (page 7-22).

getProxiedArchiveCollections

Called as a Java method by GET_PROXIED_ARCHIVECOLLECTIONS (page 7-22).

getSchemaRelations

Called as a Java method by GET_SCHEMA_TABLE_INFO (page 5-168).

getSchemaTableInfo

Called as a Java method by GET_SCHEMA_TABLE_INFO (page 5-168) and GET_SCHEMA_VIEW_INFO (page 5-169).

getSchemaTables

Called as a Java method by GET_SCHEMA_TABLES (page 5-168).

getSchemaViewInfo

Called as a Java method by GET_SCHEMA_VIEW_INFO (page 5-169).

getSchemaViews

Called as a Java method by GET_SCHEMA_VIEWS (page 5-169).

getSearchResults

Retrieves the search results. Called as a Java method by GET_SEARCH_RESULTS (page 5-174) and PNE_GET_SEARCH_RESULTS (page 5-176).

getSubscriptionList

Retrieves the subscription list and passes *QdocNameSubscription* *QnotDocNameSubscriptions* as parameters. Called as a Java method by SUBSCRIPTION_LIST (page 5-91) and UNSUBSCRIBE_FROM_LIST (page 5-98).

getTable

Called as a Java method by GET_TABLE (page 5-13).

getTemplate

Retrieves template data. Called as a Java method by GET_WF_TEMPLATE (page 6-20).

getTemplateConversions

Called as a Java method by GET_TEMPLATE_CONVERSIONS (page 5-122).

getTemplates

Retrieves the workflow template data. Called as a Java method by GET_WF_TEMPLATES (page 6-20).

getURLAbsolute

Resolves the URL of the content item. The URL is passed to the **Web Location:** entry of the DOC_INFO template. Called as a Java method by DOC_INFO (page 5-64) and REVIEW_WORKFLOW_DOC (page 6-10).

getUserAttributes

Retrieves user attributes. Called as a Java method by QUERY_USER_ATTRIBUTES (page 5-203).

getUserCollaborationList

Called as a Java method by GET_USER_CLBRA_LIST (page 2-6).

getUserMailAddress

Resolves the email address of the content item author and the user who has checked out the content item. The action passes *dDocAuthor* and *AuthorAddress* as parameters. Called as a Java method by these services:

- DOC_INFO (page 5-64)
- DOC_INFO_LATESTRELEASE (page 5-70)
- DOC_INFO_SIMPLE (page 5-71)
- DOC_INFO_SIMPLE_BYREV (page 5-72)
- GET_WORKFLOW_INFO (page 6-39)
- GET_WORKFLOW_INFO_BYNAME (page 6-40)
- PROBLEMREPORT_INFO (page 6-9)
- REVIEW_WORKFLOW_DOC (page 6-10)

getUserProfile

Retrieves user profile information. Called as a Java method by GET_USER_INFO (page 5-200).

getUsers

Retrieves user list. Called as a Java method by GET_USERS (page 5-202).

getUserUnique

Retrieves the unique user name. Called as a Java method by REGISTER_USER (page 5-204).

getViewEditInfo

Called as a Java method by GET_SCHEMA_VIEW_EDIT_INFO (page 5-168).

getViewValues

Called as a Java method by GET_SCHEMA_VIEW_VALUES (page 5-169).

getWebAppStatus

Called as a Java method by GET_WEB_APP_STATUS (page 5-122).

getWfDocuments

Retrieves workflow content items and passes *WfDocuments* as a parameter. Called as a Java method by these services:

- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOWCRITERIA (page 6-30)
- WORKFLOW_CANCEL (page 6-45)

- WORKFLOW_START (page 6-53)

getWorkflowDesignInfo

Called as a Java method by GET_WORKFLOW (page 6-39).

getWorkflowDocumentInfo

Called as a Java method by GET_WORKFLOW (page 6-39).

getWorkflowInfo

Evaluates whether the content item is part of a workflow. The action passes *WF_INFO* as a parameter. The DOC_INFO template is referenced and if *WF_INFO* exists the workflow information is included in the DOC_INFO template. Called as a Java method by DOC_INFO (page 5-64) and REVIEW_WORKFLOW_DOC (page 6-10).

getWorkflowScript

Called as a Java method by GET_WORKFLOW_SCRIPT (page 6-42).

getWorkflowStepAliasesInfo

Retrieves the workflow step alias information. Called as a Java method by GET_WORKFLOW (page 6-39).

H

hidePassword

Hides the password and passes *USER_INFO* as a parameter. Called as a Java method by GET_USER_INFO (page 5-200).

I

Ialias

Inserts the alias user information. Called as an Execute Query action by ADD_ALIAS (page 5-179).

Icollaboration

Called as an Execute Query action by ADD_COLLABORATION (page 2-3).

IdocAccount

Inserts a new account. Called as an Execute Query action by ADD_DOC_ACCOUNT (page 5-4).

IdocFormat

Inserts the content item format. Called as an Execute Query action by ADD_DOCFORMAT (page 5-5).

IdocType

Inserts the content item type. Called as an Execute Query action by ADD_DOCTYPE (page 5-6).

IextensionMap

Inserts the extension map. Called as an Execute Query action by ADD_DOCEXTENSION (page 5-4).

Imeta

Inserts the meta data information. Called as an Execute Query action by these services:

- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)

Imetadef

Inserts the meta data definition information. Called as an Execute Query action by ADD_METADEF (page 5-136).

insertAliasUsers

Adds the alias information to the database. Called as a Java method by ADD_ALIAS (page 5-179) and EDIT_ALIAS (page 5-191).

insertGroupRow

Called as a Java method by ADD_GROUP (page 5-181).

IproblemReport

Updates the problem report. Called as an Execute Query action by ADD_PROBLEMREPORT (page 6-3).

Irevision

Updates the revision ID. Called as an Execute Query action by these services:

- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)

Irole

Inserts the role. Called as an Execute Query by ADD_ROLE (page 5-181).

isDocCheckedOut

Called as a Java method by these services:

- CACHE_WORKFLOW_CHECKIN (page 5-25)
- WORKFLOW_APPROVE (page 6-45)
- WORKFLOW_CHECKIN_SUB (page 6-11)

IsecurityGroup

Inserts the security group. Called as an Execute Query action by ADD_GROUP (page 5-181).

Isubscription

Inserts the subscription. Called as an Execute Query by SUBSCRIBE_EX (page 5-89).

Iworkflow

Inserts the workflow information in the database. Called as an Execute Query action by ADD_WORKFLOW (page 6-22).

IworkflowDocument

Called as an Execute Query action by ADD_WORKFLOWDOCUMENT_SUB (page 6-25).

IworkflowState

Provides an internal status table that stores information about content items in active workflows. Called as an Execute Query action by WORKFLOW_APPROVE (page 6-45).

L

loadActiveComponentData

Loads configuration information and passes *ACTIVE_COMPONENTS* as a parameter. Called as a Java method by CONFIG_INFO (page 5-7).

loadAndValidateValues

Evaluates the specified values. Called as a Java method by these services:

- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)
- CRITERIAWORKFLOW_ENABLE (page 6-28)
- WORKFLOW_CANCEL (page 6-45)
- WORKFLOW_START (page 6-53)

loadCollaborationList

Called as a Java method by GET_ACTIVE_WORKFLOWS (page 6-37).

loadComponentInstallInfo

Called as a Java method by GET_COMPONENT_INSTALL_FORM (page 5-134).

loadComponentInstallSettings

Called as a Java method by GET_COMPONENT_INSTALL_SETTINGS' (page 5-134).

loadConfigurationInfo

Loads the content item specific configuration information. Called as a Java method by GET_DOC_CONFIG_INFO (page 5-77).

loadDefaultInfo

Loads the default configuration information. Called as a Java method by these services:

- ASSIGN_DOCINFO_FORM (page 5-24)
- CHECKIN_NEW_FORM (page 5-37)

- CHECKIN_SEL_FORM (page 5-45)
- CHECKIN_SIMILAR_FORM (page 5-47)
- GET_DOC_CONFIG_INFO (page 5-77)
- GET_PORTAL_PAGE (page 5-150)
- GET_UPDATE_FORM (page 5-80)
- LOAD_DOC_ENVIRONMENT (page 5-14)

loadDocConfig

Loads the content item configuration information. Called as a Java method by DOC_FORMATS_WIZARD (page 5-64) and EDIT_DOC_FORMATS (page 5-75).

loadDocDefaults

Called as a Java method by these services:

- ASSIGN_DOCINFO_FORM (page 5-24)
- CHECKIN_NEW_FORM (page 5-37)
- CHECKIN_SIMILAR_FORM (page 5-47)

loadFilterConfig

Called as a Java method by these services:

- ASSIGN_DOCINFO_FORM (page 5-24)
- CHECKIN_NEW_FORM (page 5-37)
- CHECKIN_SIMILAR_FORM (page 5-47)

loadGlobalIncludes

Loads the global includes. Called as a Java method by LOAD_GLOBALINCLUDES (page 5-153).

loadMetaDefaults

Loads the default configuration information. Called as a Java method these services:

- ASSIGN_DOCINFO_FORM (page 5-24)
- CHECKIN_NEW_FORM (page 5-37)
- CHECKIN_SIMILAR_FORM (page 5-47)
- GET_UPDATE_FORM (page 5-80)

loadMetaOptionsLists

Loads the meta data options list. Called as a Java method by these services:

- ASSIGN_DOCINFO_FORM (page 5-24)
- CHECKIN_NEW_FORM (page 5-37)
- CHECKIN_SEL_FORM (page 5-45)
- CHECKIN_SIMILAR_FORM (page 5-47)
- GET_DOC_CONFIG_INFO (page 5-77)
- GET_PORTAL_PAGE (page 5-150)
- GET_UPDATE_FORM (page 5-80)
- LOAD_DOC_ENVIRONMENT (page 5-14)

loadPRStateLists

Loads the problem report state lists. Called as a Java method by GET_PROBLEMREPORTS_SEARCH_FORM (page 6-6) and GET_UPDATE_PROBLEMREPORT_FORM (page 6-7).

loadRegisteredProjects

Retrieves problem report search information. Called as a Java method by GET_PROBLEMREPORTS_SEARCH_FORM (page 6-6).

loadServerOutput

Loads the server output. Called as a Java method by CLEAR_SERVER_OUTPUT (page 5-134) and GET_SERVER_OUTPUT (page 5-135).

loadSharedTable

Loads the shared table information and passes *ArchiveCollections* as a parameter. Parameters vary by service. Called as a Java method by these services:

- CREATE_SUBSCRIPTION_TYPE (page 5-60)
- DELETE_SUBSCRIPTION_TYPE (page 5-64)
- GET_ARCHIVECOLLECTIONS (page 7-17)
- GET_DOC_CONFIG_INFO (page 5-77)
- GET_PACKAGE_ENVIRONMENT_PAGE (page 5-79)
- GET_PORTAL_PAGE (page 5-150)
- GET_RESULT_OPTIONS (page 5-12)
- GET_SYSTEM_AUDIT_INFO (page 5-13)

- GET_USER_INFO (page 5-200)
- LOAD_DOC_ENVIRONMENT (page 5-14)
- UPDATE_SUBSCRIPTION_TYPE (page 5-110)

loadTopic

Called as a Java method by these services:

- LOAD_PNE_PORTAL (page 5-202)
- LOAD_USER_TOPIC (page 5-203)
- LOAD_WORKFLOW_QUEUE (page 6-8)

loadTraceFlags

Called as a Java method by GET_SYSTEM_AUDIT_INFO (page 5-13).

loadUserAndCheckEditAllowed

Loads the user information and evaluates the allowed edit privilege for the user. Called as a Java method by these services:

- ADD_USER (page 5-182)
- CHANGE_USER_AUTH_TYPE (page 5-184)
- DELETE_USER (page 5-191)
- EDIT_USER (page 5-195)
- QUERY_USER_ATTRIBUTES (page 5-203)

loadUserMetaData

Loads the user meta data information. Called as a Java method by these services:

- GET_FILTER_ADMIN_PAGE (page 5-200)
- GET_SELF_REGISTER_PAGE (page 5-200)
- GET_USER_INFO (page 5-200)

loadWfCompanionInfo

Retrieves information for companion workflows. Called as a Java method by GET_WF_COMPANION_INFO (page 6-38).

M

makeNewRevClass

Called as a Java method by CACHE_CHECKIN_NEW (page 5-24) and CHECKIN_NEW_SUB (page 5-38).

mapDocNamedResultSetValuesCheckMetaChange

Called as a Java method by CACHE_WORKFLOW_CHECKIN (page 5-25).

mapDocResultSetCheckMetaChange

Called as a Java method by CACHE_CHECKIN_SEL (page 5-24) and CHECKIN_SEL_SUB (page 5-47).

mapNamedResultSetValues

Retrieves the data assigned to the parameter *DOC_INFO* and maps the result set values for *dStatus*, *dReleaseState*, and *dProcessingState*. The parameter is the same among services while the result set map values vary by service. Called as a Java method by these services:

- DELETE_REV (page 5-62)
- DOC_INFO (page 5-64)
- RESUBMIT_FOR_CONVERSION (page 5-82)
- REVIEW_WORKFLOW_DOC (page 6-10)

mapResultSet

Maps the result set and passes *QdocInfo, dRevClassID, dDocName, dSecurityGroup, dCheckoutUser, dDocAccount*, and *dPublishState* as parameters. Parameters vary based on service. Called as a Java method by these services:

- ADD_PROBLEMREPORT (page 6-3)
- APPLET_DOCINFO (page 5-6)
- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_FORM (page 5-45)

- CHECKIN_SEL_SUB (page 5-47)
- DOC_INFO_LATESTRELEASE (page 5-70)
- DOC_INFO_SIMPLE_BYREV (page 5-72)
- INSERT_NEW (page 7-28)
- RESUBMIT_FOR_CONVERSION (page 5-82)
- UNDO_CHECKOUT (page 5-92)
- UNDO_CHECKOUT_BY_NAME (page 5-95)
- UNSUBSCRIBE (page 5-95)
- WORKFLOW_APPROVE (page 6-45)
- WORKFLOW_CHECKIN_SUB (page 6-11)
- WORKFLOW_REJECT (page 6-50)

markDocDeleted

Sets the status message and passes *delete_doc* as a parameter. Called as a Java method by these services:

- DELETE_BYCLASS (page 5-60)
- DELETE_BYNAME (page 5-60)
- DELETE_DOC (page 5-61)

markRevDeleted

Updates the revision status as deleted. Called as a Java method by these services:

- DELETE_BYREV (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)

mergeTable

Merges the named table. Called as a Java method by MERGE_TABLE (page 5-16).

N

notifyChange

Notifies the provider of changes to the content item. Called as a Java method by NOTIFY_CHANGE (page 5-161).

notifyCollaborationUsers

Called as a Java method by ADD_COLLABORATION (page 2-3) and EDIT_CLBRA_ACCESS_LIST (page 2-4).

notifyContributor

Called as a Java method by ADD_PROBLEMREPORT (page 6-3) and RESEND_PROBLEMREPORT (page 6-9).

notifyEndPublish

Called as a Java method by END_PUBLISHING (page 9-4).

notifyPublisher

Called as a Java method by ADD_PROBLEMREPORT (page 6-3) and RESEND_PROBLEMREPORT (page 6-9).

notifyStartPublish

Called as a Java method by START_PUBLISHING (page 9-7).

P

packageEnvironment

Called as a Java method by PACKAGE_ENVIRONMENT (page 5-81).

packageLocalization

Called as a Java method by LOAD_USER_LOCALIZATION (page 5-144).

pageRequest

Executes an HTML page request. Called as a Java method by these services:

- GET_ADMIN_PAGE (page 5-145)
- GET_DOC_PAGE (page 5-146)
- GET_PORTAL_PAGE (page 5-150)
- GET_SECURE_PAGE (page 5-151)
- LOGIN (page 5-15)
- PNE_SAVE_QUERY (page 5-154)

- PNE_UPDATE_PERSONAL_URLS (page 5-154)
- PNE_UPDATE_PORTAL_INFO (page 5-155)

performPreview

Called as a Java method by PREVIEW_DOCUMENT (page 9-7).

postCheckinFilter

Processes the post checkin filter. Called as a Java method by FORM_PROCESS (page 5-76).

postHtmlFormCheckin

Submits the HTML form. Called as a Java method by CONTINUE_SUBMIT_HTML_FORM (page 5-59) and SUBMIT_HTML_FORM (page 5-85).

prepareCheckinSecurity

Evaluates the assigned security level to verify that the user is authorized to perform this action. Called as a Java method by these services:

- CHECKIN_NEW (page 5-31)
- CHECKIN_SEL (page 5-40)
- UPDATE_DOCINFO_SUB (page 5-110)

prepareDocInfoValidate

Called as a Java method by VALIDATE_DOCINFO (page 5-112).

prepareFormContinue

Called as a Java method by CONTINUE_SUBMIT_HTML_FORM (page 5-59).

prepareForPreview

Called as a Java method by CACHE_SUBMIT_HTML_FORM (page 5-24).

prepareInsertNew

Called as a Java method by INSERT_NEW (page 7-28).

prepareMailForStepUsers

Sends email to the workflow step users. Called as a Java method by WORKFLOW_REJECT (page 6-50) and WORKFLOW_START (page 6-53).

prepareRedirect

Prepares the redirect template for the specified service. Called as a Java method by these services:

- ADD_COLLABORATION (page 2-3)
- ADD_EDIT_PROVIDER (page 5-157)
- ADD_PROBLEMREPORT (page 6-3)
- CHECKIN_NEW (page 5-31)
- CHECKIN_SEL (page 5-40)
- CHECKOUT (page 5-52)
- CLEAR_SERVER_OUTPUT (page 5-134)
- CONTINUE_SUBMIT_HTML_FORM (page 5-59)
- DELETE_COLLABORATION (page 2-4)
- DELETE_PROBLEMREPORT (page 6-4)
- DELETE_PROVIDER (page 5-158)
- DELETE_REV (page 5-62)
- EDIT_CLBRA_ACCESS_LIST (page 2-4)
- EDIT_COLLABORATION (page 2-4)
- EDIT_DOC_FORMATS (page 5-75)
- EDIT_TRACE_OPTIONS (page 5-10)
- EDIT_USER_PROFILE (page 5-198)
- ENABLE_DISABLE_PROVIDER (page 5-159)
- RESUBMIT_FOR_CONVERSION (page 5-82)
- SAVE_TEMPLATE_CONVERSIONS (page 5-124)
- SAVE_USER_TOPICS (page 5-205)
- SUBMIT_HTML_FORM (page 5-85)
- SUBSCRIBE (page 5-86)
- TEST_PROVIDER (page 5-162)
- UNDO_CHECKOUT (page 5-92)
- UNREGISTER_PROJECT (page 9-8)
- UNSUBSCRIBE (page 5-95)
- UPDATE_DOCINFO_BYFORM (page 5-105)
- UPDATE_FILTER_INFO (page 5-209)

- UPDATE_PROBLEMREPORT (page 6-10)
- UPLOAD_NEW_COMPONENT (page 5-135)
- WORKFLOW_APPROVE (page 6-45)
- WORKFLOW_CHECKIN (page 6-45)
- WORKFLOW_REJECT (page 6-50)

prepareTopicEdits

Prepares the topic edits. Called as a Java method by these services:

- EDIT_USER_PROFILE (page 5-198)
- GET_CLBRA_DOCUMENTS (page 2-5)
- PNE_GET_SEARCH_RESULTS (page 5-176)
- PNE_SAVE_QUERY (page 5-154)
- PNE_UPDATE_PERSONAL_URLS (page 5-154)
- PNE_UPDATE_PORTAL_INFO (page 5-155)
- SAVE_USER_TOPICS (page 5-205)

prepareWebViewableDelivery

Called as a Java method by GET_DYNAMIC_CONVERSION (page 5-118) and GET_EXTERNAL_DYNAMIC_CONVERSION (page 5-122).

prepSubscription

Prepares the subscription. Called as a Java method by DOC_SUBS_LIST (page 5-73).

prepSubscriptionDateUpdate

Prepares the subscription date for update and passes the specified parameter. Called as a Java method by UPDATE_SUBSCRIPTION_NOTIFY (page 5-110) and UPDATE_SUBSCRIPTION_USED (page 5-111).

processCheckinArchive

Checks the content item into the archive. Called as a Java method by these services:

- CHECKIN_ARCHIVE (page 7-6)
- CHECKIN_ARCHIVE_NO_NOTIFY (page 5-25)
- CHECKIN_PUBLISH (page 9-4)

processForm

Processes the form submission and passes *checkSecurity* as a parameter. Called as a Java method by FORM_PROCESS (page 5-76).

Q

QactiveWorkflowProjects

Retrieves registered project information. The result of this query is assigned to the parameter *ProjectInfo* and stored for later use. The action passes *mustExist* as a parameter. Called as a Select Cache Query action by GET_PROJECT_REGISTRATION (page 9-6).

Qalias

Queries the user alias information. The result of this query is assigned to the parameter *AliasInfo*. Called as a Select Query action by ADD_ALIAS (page 5-179) and EDIT_ALIAS (page 5-191).

QaliasesForUser

Queries if the user has been assigned an alias. The result of this query is assigned to the parameter *Alias*. Called as a Select Query action by DELETE_USER (page 5-191).

QarchivedDoc

Retrieves archived content item information from the database. The result of this query is assigned to the parameter *FILE_DOC_INFO* and stored for later use. This action should not throw any exceptions. The control mask setting specifies that the query must return a record or the action fails with the given error message. Called as a Select Cache Query action by GET_ARCHIVED_FILE (page 7-17).

Qcache

Called as a Select Cache Query action by CONTINUE_SUBMIT_HTML_FORM (page 5-59).

QcheckinCache

Called as a Select Cache Query by these services:

- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)

- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CONTINUE_CHECKIN (page 5-59)
- DELETE_CHECKIN_CACHE (page 5-61)
- GET_CACHED_CHECKIN_INFO (page 5-77)
- GET_PREVIEW_LIST (page 9-5)

QcheckinCachesForUser

Called as a Select Cache Query action by these services:

- CHECKIN_LIST (page 5-30)
- CONTINUE_CHECKIN (page 5-59)
- DELETE_BYREV (page 5-61)

Qcollaboration

Called as a Select Query action by ADD_COLLABORATION (page 2-3).

Called as a Select Cache Query action by these services:

- DELETE_COLLABORATION (page 2-4)
- EDIT_CLBRA_ACCESS_LIST (page 2-4)
- EDIT_CLBRA_ACCESS_LIST_FORM (page 2-4)
- EDIT_COLLABORATION (page 2-4)
- EDIT_COLLABORATION_FORM (page 2-5)
- GET_CLBRA_DOCUMENTS (page 2-5)
- GET_CLBRA_INFO (page 2-5)

QdocAccount

Queries the content item account information. The result of this query is assigned to the parameter *DOCACCOUNT_INFO*. Called as a Select Query action by ADD_DOC_ACCOUNT (page 5-4).

QdocAccounts

Retrieves content item account information. The result of this query is assigned to the parameter *DOCACCOUNT_INFO* and stored for later use. Called as a Select Cache Query action by QUERY_DOC_ACCOUNTS (page 5-20).

QdocInfo

Retrieves content item information. The result of this query is assigned to the parameter *DOC_INFO*. Called as a Select Cache Query action by these services:

- ADD_PROBLEMREPORT (page 6-3)
- APPLET_DOCINFO (page 5-6)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_SEL_FORM (page 5-45)
- CHECKOUT (page 5-52)
- CHECKOUT_BY_NAME (page 5-55)
- CHECKOUT_OK (page 5-56)
- DELETE_DOC (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)
- DOC_INFO (page 5-64)
- DOC_INFO_LATESTRELEASE (page 5-70)
- DOC_INFO_SIMPLE (page 5-71)
- DOC_INFO_SIMPLE_BYREV (page 5-72)
- GET_DOC_CONFIG_INFO (page 5-77)
- GET_DYNAMIC_CONVERSION (page 5-118)
- GET_FILE (page 5-129)
- GET_PREVIEW_LIST_EX (page 9-5)
- GET_UPDATE_FORM (page 5-80)
- GET_WORKFLOW_INFO (page 6-39)
- NOTIFY_CONTRIBUTOR (page 6-9)
- RESUBMIT_FOR_CONVERSION (page 5-82)
- REV_HISTORY (page 5-84)
- REVIEW_WORKFLOW_DOC (page 6-10)
- SELECTDOC (page 5-84)
- SUBSCRIBE (page 5-86)
- SUBSCRIBE_DOC_USER (page 5-88)
- SUBSCRIBE_FORM (page 5-90)
- UNSUBSCRIBE_FORM (page 5-97)

- UPDATE_DOCINFO_BYREV (page 5-109)
- UPDATE_DOCINFO_SUB (page 5-110)
- WORKFLOW_CHECKIN (page 6-45)

QdocInfoSimilarCheckin

Called as a Select Cache Query by CHECKIN_SEL_SUB (page 5-47).

QdocName

Called as a Select Cache Query by these services:

- DELETE_PROBLEMREPORT (page 6-4)
- DOC_INFO_SIMPLE (page 5-71)
- GET_UPDATE_PROBLEMREPORT_FORM (page 6-7)
- GET_WF_COMPANION_INFO (page 6-38)
- PROBLEMREPORT_INFO (page 6-9)
- RESEND_PROBLEMREPORT (page 6-9)
- UPDATE_PROBLEMREPORT (page 6-10)

QdocNameMeta

Retrieves the content item name. The result of this query is assigned to the parameter *DOC_INFO* and stored for later use. Called as a Select Cache Query by these services:

- DOC_INFO_SIMPLE (page 5-71)
- GET_DOCUMENT_PROBLEMREPORTS (page 6-5)
- GET_WORKFLOW_INFO_BYNAME (page 6-40)

QdocType

Queries the content item's file type information. The result of this query is assigned to the parameter *DOCTYPE_INFO*. Called as a Select Query action by ADD_DOCTYPE (page 5-6).

QdocTypeDocuments

Queries a specific content item's file type information. The result of this query is assigned to the parameter *DOCUMENTS*. Called as a Select Query action by DELETE_DOCTYPE (page 5-8).

QdocTypes

Retrieves content item types. The result of this query is assigned to the parameter *DocTypes* and stored for later use. Called as a Select Cache Query action by GET_DOCTYPES (page 5-11).

QdocumentProblemReports

Called as a Select Cache Query action by GET_DOCUMENT_PROBLEMREPORTS (page 6-5).

Qdocuments

Called as a Select Cache Query action by UPDATE_DOCINFO_BYREV (page 5-109) and UPDATE_DOCINFO_SUB (page 5-110).

QextensionFormatMap

Queries the content item's format map extension. The result of this query is assigned to the parameter *FormatMap*. Called as a Select Query action by DELETE_DOCFORMAT (page 5-8).

QextensionMap

Queries the mapping of the file extension. The result of this query is assigned to the parameter *ExtensionMap*. Called as a Select Query action by ADD_DOCEXTENSION (page 5-4).

QformatMap

Queries the format map information. The result of this query is assigned to the parameter *FormatMap*. Called as a Select Query action by ADD_DOCFORMAT (page 5-5).

Qgroup

Queries group information. The result of this query is assigned to the parameter *GROUP_INFO*. Called as a Select Query action by ADD_GROUP (page 5-181).

QgroupRevisions

Queries group revision information. The result of this query is assigned to the parameter *GROUP_REVS*. Called as a Select Query action by DELETE_GROUP (page 5-190).

QgroupRole

Queries group role information. The result of this query is assigned to the parameter *GroupRole*. Called as a Select Cache Query action by EDIT_GROUP (page 5-194).

QgroupWF

Queries group workflow information. The result of this query is assigned to the parameter *GROUP_WF*. Called as a Select Query action by DELETE_GROUP (page 5-190).

QisAliasSubscribed

Queries if the alias is subscribed. The result of this query is assigned to the parameter *SUBSCRIIPTION_INFO*. Called as a Select Query action by SUBSCRIBE_EX (page 5-89).

QisDocWFLocked

Called as a Select Query action by ADD_WORKFLOWDOCUMENT_SUB (page 6-25).

QisWfTokenUsed

Called as a Select Query action by DELETE_WORKFLOW_TOKEN (page 6-30).

QlatestIDByName

Retrieves the latest content item name information. The result of this query is assigned to the parameter *DOC_LATEST_ID* and stored for later use. The control mask setting specifies that the query must return a record or the action fails with the given error message. Called as a Select Cache Query action by CHECKOUT_BY_NAME (page 5-55) and UNDO_CHECKOUT_BY_NAME (page 5-95).

Qmetadef

Called as a Select Query action by these services:

- ADD_METADEF (page 5-136)
- EDIT_METADEF (page 5-138)
- DEL_METADEF (page 5-138)

QmetaFieldInfo

Retrieves meta data field information and passes *MetaFieldInfo* as a parameter. Called as a Select Cache Query action by GET_METADEFS (page 5-12) and UPDATE_META_TABLE (page 5-142).

QODMAdocInfo

Called as a Select Cache Query action by ODMA_DOC_INFO_SIMPLE (page 5-81).

QprjDocumentsDeleted

Called as a Select Cache Query action by GET_PROJECT_ACTIONS (page 9-5).

QprjDocumentsForAgent

Called as a Select Cache Query action by GET_PROJECT_ACTIONS (page 9-5).

QproblemReport

Retrieves problem report information from the database using a query. The result of this query is assigned to the parameter *ProblemReport* and stored for later use. Called as a Select Cache Query by these services:

- ADD_PROBLEMREPORT (page 6-3)
- DELETE_PROBLEMREPORT (page 6-4)
- GET_UPDATE_PROBLEMREPORT_FORM (page 6-7)
- PROBLEMREPORT_INFO (page 6-9)
- RESEND_PROBLEMREPORT (page 6-9)
- UPDATE_PROBLEMREPORT (page 6-10)

QprojectDocument

Called as a Select Cache Query by these services:

- ADD_PROBLEMREPORT (page 6-3)
- GET_DOCUMENT_PROBLEMREPORTS (page 6-5)
- PROBLEMREPORT_INFO (page 6-9)
- RESEND_PROBLEMREPORT (page 6-9)

QregisteredProject

Retrieves registered project information. The result of this query is assigned to the specified parameter. Called as a Select Cache Query by these services:

- ADD_PROBLEMREPORT (page 6-3)
- GET_DOCUMENT_PROBLEMREPORTS (page 6-5)
- GET_PROJECT_REGISTRATION (page 9-6)
- GET_PROJECT_REGISTRATION (page 9-6)
- PROBLEMREPORT_INFO (page 6-9)
- REGISTER_PROJECT (page 9-7)

- RESEND_PROBLEMREPORT (page 6-9)

QregisteredProjects

Called as a Select Cache Query action by GET_REGISTEREDPROJECTS (page 9-7).

QreleasedDocInfo

Retrieves content item security information. The result of this query is assigned to the parameter *SEC_DOC_INFO* and stored for later use. The control mask setting specifies that the query must return a record or the action fails with the given error message. Called as a Select Cache Query action by GET_HIGHLIGHT_INFO (page 5-172) and GET_XML_HIGHLIGHT_INFO (page 5-175).

QrevHistory

Retrieves revision history information. The result of this query is assigned to the parameter *REVISION_HISTORY*. The DOC_INFO template uses REVISION_HISTORY in a loop to present information about each revision in the DOC_INF page. Called as a Select Cache Query action by DOC_INFO (page 5-64) and REVIEW_WORKFLOW_DOC (page 6-10).

QrevHistoryReleased

Retrieves revision history information. The result of this query is assigned to the parameter *REVISION_HISTORY*. The DOC_INFO template uses REVISION_HISTORY in a loop to present information about each revision in the DOC_INFO page. Called as a Select Cache Query action by REV_HISTORY (page 5-84).

QrevisionsByClass

Retrieves revision information by class. The result of this query is assigned to the parameter *REVISIONS* and stored for later use. Called as a Select Cache Query by DELETE_BYCLASS (page 5-60) and DELETE_DOC (page 5-61).

QrevisionsByName

Called as a Select Cache Query action by DELETE_BYNAME (page 5-60).

Qrole

Queries role information. The result of this query is assigned to the parameter *ROLE_INFO*. Called as a Select Query action by ADD_ROLE (page 5-181).

Qroles

Queries role information. The result of this query is assigned to the parameter *ROLES* and stored for later use. The action throws a data exception if it is unable to query for list of roles. Called as a Select Cache Query action by ADD_GROUP (page 5-181).

QsecurityGroup

Retrieves security group information from the database. The result of this query is assigned to the parameter *SECURITY_GROUPS*. The control mask setting specifies that the query must return a record or the action fails with the given error message. The action throws a data exception if the security group is not in the database.

Called as a Select Query action by EDIT_GROUP (page 5-194).

Called as a Select Cache Query action by QUERY_GROUP (page 5-203).

Quser

Retrieves user information. The result of this query is assigned to the parameter *USER_INFO* and stored for later use. The control mask setting specifies that the query must return a record or the action fails with the given error message. The action throws a data exception if the system is unable to retrieve information for the specified user. Called as a Select Cache Query action by GET_USER_INFO (page 5-200).

QuserMetaFieldInfo

Called as a Select Cache Query action by GET_USER_METADEFS (page 5-14) and UPDATE_USER_META_TABLE (page 5-141).

QuserSecurityAttributeByType

Retrieves the security attributes by type. The result of this query is assigned to the parameter *RoleUsers*. Called as a Select Query action by DELETE_ROLE (page 5-190).

QuserSubscription

Retrieves the user subscription information. The result of this query is assigned to the parameter *USER_SUBSCRIPTION* and stored for later use. Called as a Select Cache Query action by DOC_SUBS_LIST (page 5-73).

QwfCriteriaDocRevs

Retrieves the workflow revision criteria. The result of this query is assigned to the parameter *WfDocuments* and stored for later use. This action should not throw any exceptions. Called as a Select Cache Query by CRITERIAWORKFLOW_DISABLE (page 6-27) and CRITERIAWORKFLOW_DISABLE_SUB (page 6-28).

QwfDocInformation

References the active workflow content item revision ID. The result of this query is assigned to the parameter *DOC_INFO* and stored for later use. Called as a Select Cache Query action by WORKFLOW_REJECT (page 6-50).

QwfDocName

Called as a Select Cache Query action by ADD_WORKFLOWDOCUMENT_SUB (page 6-25).

QwfDocState

Called as a Select Cache Query action by GET_WORKFLOW_INFO (page 6-39) and GET_WORKFLOW_INFO_BYNAME (page 6-40).

QwfStates

Retrieves the workflow state information and provides an internal status table that stores information about content items in active workflows. The result of this query is assigned to the parameter *WorkflowState* and stored for later use. This action should not throw any exceptions. Called as a Select Cache Query action by GET_WORKFLOWDOCREVISIONS (page 6-42).

Qworkflow

Retrieves workflow information. The result of this query is assigned to the specified parameter. Called as a Select Query by action by ADD_WORKFLOW (page 6-22).

Called as a Select Cache Query action by these services:

- ADD_WORKFLOWALIASES (page 6-24)
- ADD_WORKFLOWDOCUMENT (page 6-24)
- ADD_WORKFLOWDOCUMENTS (page 6-25)
- ADD_WORKFLOWSTEP (page 6-26)
- CRITERIAWORKFLOW_ENABLE (page 6-28)
- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)
- DELETE_WFCONTRIBUTORS (page 6-28)
- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOWCRITERIA (page 6-30)
- DELETE_WORKFLOWSTEP (page 6-32)

- DELETE_WORKFLOWDOCUMENTS (page 6-30)
- DELETE_WORKFLOWSTEP (page 6-32)
- EDIT_WORKFLOW (page 6-32)
- EDIT_WORKFLOWCRITERIA (page 6-34)
- EDIT_WORKFLOWSTEP (page 6-35)
- GET_WORKFLOW (page 6-39)
- GET_WORKFLOWDOCUMENTS (page 6-7)
- GET_WORKFLOWDOCREVISIONS (page 6-42)
- WORKFLOW_START (page 6-53)
- WORKFLOW_CANCEL (page 6-45)

QworkflowAlias

Retrieves workflow alias information. The result of this query is assigned to the parameter *WFAlias*. Called as a Select Query action by DELETE_ALIAS (page 5-189).

QworkflowAliases

Retrieves workflow alias information. Called as a Select Cache Query action by DELETE_WFCONTRIBUTORS (page 6-28)

QworkflowCriteriaForGroup

Called as a Select Cache Query action by GET_CRITERIA_WORKFLOWS_FOR_GROUP (page 6-5).

QworkflowCriteriaStepsForGroup

Called as a Select Cache Query action by GET_CRITERIA_WORKFLOWS_FOR_GROUP (page 6-5).

QworkflowDocument

Retrieves workflow content item information. The result of this query is assigned to the parameter *WfDocuments* and stored for later use. Called as a Select Cache Query action by these services:

- GET_WF_COMPANION_INFO (page 6-38)
- GET_WORKFLOW_INFO (page 6-39)
- GET_WORKFLOW_INFO_BYNAME (page 6-40)

QworkflowDocuments

Retrieves workflow content item information. The result of this query is assigned to the parameter *WfDocuments* and stored for later use. Called as a Select Cache Query action by GET_WORKFLOW (page 6-39) and GET_WORKFLOWDOCUMENTS (page 6-7).

QworkflowForID

Called as a Select Cache Query action by GET_WORKFLOW_INFO (page 6-39) and GET_WORKFLOW_INFO_BYNAME (page 6-40).

Qworkflows

Called as a Select Cache Query by GET_WORKFLOWS_FOR_ALL (page 6-8).

QworkflowStep

Retrieves workflow step information. The result of this query is assigned to the parameter *STEP_INFO*. The action throws a data exception if the step name is not unique. Called as a Select Query action by ADD_WORKFLOWSTEP (page 6-26). Called as a Select Cache Query action by EDIT_WORKFLOWSTEP (page 6-35).

QworkflowSteps

Retrieves workflow step information including step description, type, and number of reviewers required to pass each step. The result of this query is assigned to the specified parameter. Called as a Select Cache Query action by these services:

- CRITERIAWORKFLOW_ENABLE (page 6-28)
- GET_WORKFLOW_INFO (page 6-39)
- GET_WORKFLOW_INFO_BYNAME (page 6-40)
- WORKFLOW_START (page 6-53)

QworkflowStepsAll

Called as a Select Cache Query action by GET_WORKFLOWS_FOR_ALL (page 6-8).

R

refreshCache

Refreshes the cached specified information. The specified information varies by service. Called as a Java method by these services:

- ADD_ALIAS (page 5-179)

- ADD_DOC_ACCOUNT (page 5-4)
- ADD_DOCEXTENSION (page 5-4)
- ADD_DOCFORMAT (page 5-5)
- ADD_DOCTYPE (page 5-6)
- ADD_METADEF (page 5-136)
- ADD_WORKFLOW (page 6-22)
- ADD_WORKFLOWALIASES (page 6-24)
- ADD_WORKFLOWDOCUMENT (page 6-24)
- ADD_WORKFLOWDOCUMENTS (page 6-25)
- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_ENABLE (page 6-28)
- DEL_METADEF (page 5-138)
- DELETE_ALIAS (page 5-189)
- DELETE_DOC_ACCOUNT (page 5-7)
- DELETE_DOCEXTENSION (page 5-7)
- DELETE_DOCFORMAT (page 5-8)
- DELETE_DOCTYPE (page 5-8)
- DELETE_WFCONTRIBUTORS (page 6-28)
- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOWCRITERIA (page 6-30)
- DELETE_WORKFLOWDOCUMENTS (page 6-30)
- EDIT_ALIAS (page 5-191)
- EDIT_DOCEXTENSION (page 5-8)
- EDIT_DOCFORMAT (page 5-9)
- EDIT_DOC_FORMATS (page 5-75)
- EDIT_DOCTYPE (page 5-10)
- EDIT_METADEF (page 5-138)
- EDIT_WORKFLOW (page 6-32)
- EDIT_WORKFLOWCRITERIA (page 6-34)
- REGISTER_PROJECT (page 9-7)
- UNREGISTER_PROJECT (page 9-8)

- UPDATE_META_TABLE (page 5-142)
- UPDATE_USER_META (page 5-141)
- UPDATE_USER_META_TABLE (page 5-141)
- WORKFLOW_CANCEL (page 6-45)
- WORKFLOW_START (page 6-53)

refreshLayoutLists

Called as a Java method by LM_LOAD_LAYOUTS_SUB (page 5-14).

refreshRoles

Refreshes the user roles. Called as a Java method by these services:

- ADD_GROUP (page 5-181)
- ADD_ROLE (page 5-181)
- DELETE_GROUP (page 5-190)
- DELETE_ROLE (page 5-190)
- EDIT_ROLE (page 5-194)

registerProject

Called as a Java method by REGISTER_PROJECT (page 9-7).

rejectDoc

Rejects the content item and passes *UrevisionStatus* as a parameter. Called as a Java method by WORKFLOW_REJECT (page 6-50).

remoteCredentialsCheck

Called as a Java method by CHECK_USER_CREDENTIALS (page 5-185).

removeCachedUser

Called as a Java method by these services:

- ADD_USER (page 5-182)
- DELETE_USER (page 5-191)
- EDIT_USER (page 5-195)
- REGISTER_USER (page 5-204)

removeContext

Called as a Java method by REMOVE_WEB_APP (page 5-124).

removeSubscription

Deletes the subscription and passes *Dsubscription* as a parameter. Called as a Java method by these services:

- UNSUBSCRIBE (page 5-95)
- UNSUBSCRIBE_FROM_LIST (page 5-98)
- UNSUBSCRIBE_FROM_LIST_EX (page 5-100)

renameValues

Renames the associated values and passes specified parameters. Called as a Java method by these services:

- ADD_PROBLEMREPORT (page 6-3)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_PUBLISH (page 9-4)
- CHECKIN_SEL_SUB (page 5-47)
- CHECKIN_SIMILAR_FORM (page 5-47)
- DELETE_PROBLEMREPORT (page 6-4)
- DELETE_REV (page 5-62)
- DELETE_ROLE (page 5-190)
- DELETE_USER (page 5-191)
- END_PUBLISHING (page 9-4)
- GET_PROJECT_ACTIONS (page 9-5)
- GET_PROJECT_REGISTRATION (page 9-6)
- GET_UPDATE_FORM (page 5-80)
- GET_WORKFLOW_INFO (page 6-39)
- GET_WORKFLOW_INFO_BYNAME (page 6-40)
- NOTIFY_CONTRIBUTOR (page 6-9)
- REGISTER_PROJECT (page 9-7)
- START_PUBLISHING (page 9-7)
- SUBSCRIBE (page 5-86)

- UNREGISTER_PROJECT (page 9-8)
- WORKFLOW_APPROVE (page 6-45)
- WORKFLOW_CHECKIN_SUB (page 6-11)
- WORKFLOW_REJECT (page 6-50)
- WORKFLOW_START (page 6-53)

requestSecurityInfo

Requests the security information. Called as a Java method by REQUEST_SECURITYINFO (page 5-162).

resubmitDocToConversion

Resubmits the content item for conversion. Called as a Java method by RESUBMIT_FOR_CONVERSION (page 5-82).

retrieveAllProviderInfo

Retrieves the provider list. Called as a Java method by GET_ALL_PROVIDERS (page 5-161)

retrieveCachedInfo

Called as a Java method by these services:

- CONTINUE_CHECKIN (page 5-59)
- CONTINUE_SUBMIT_HTML_FORM (page 5-59)
- GET_CACHED_CHECKIN_INFO (page 5-77)
- GET_PREVIEW_LIST (page 9-5)

retrieveProblemReportInfo

Retrieves the problem report information. Called as a Java method by these services:

- GET_UPDATE_PROBLEMREPORT_FORM (page 6-7)
- PROBLEMREPORT_INFO (page 6-9)
- RESEND_PROBLEMREPORT (page 6-9)

retrieveProviderInfo

Retrieves registered provider information. Called as a Java method by GET_PROVIDER_INFO (page 5-161).

retrieveUserDatabaseProfileData

Retrieves the user database profile information. Called as a Java method by REGISTER_USER (page 5-204).

runHtmlConversion

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB (page 5-120).

S

saveGlobalIncludes

Saves the global includes. Called as a Java method by SAVE_GLOBALINCLUDES (page 5-155).

saveTemplateConversions

Called as a Java method by SAVE_TEMPLATE_CONVERSIONS (page 5-124)

saveUserOptionList

Saves the user option list. Called as a Java method by UPDATE_USEROPTION_LIST (page 5-209).

searchCacheReport

Called as a Java method by APPEND_SEARCH_AUDIT_INFO (page 5-170).

searchProblemReports

Retrieves problem report search information. Called as a Java method by GET_PROBLEMREPORTS_SEARCH_RESULTS (page 6-6).

sendMailTo

Sends an email to the problem report contributor. Called as a Java method by NOTIFY_CONTRIBUTOR (page 6-9).

setConditionVars

Called as a Java method by these services:

- ADD_COLLABORATION (page 2-3)
- ADD_COLLABORATION_FORM (page 2-3)
- ADD_WORKFLOW (page 6-22)

- ADD_WORKFLOWALIASES (page 6-24)
- ADD_WORKFLOWDOCUMENT (page 6-24)
- ADD_WORKFLOWDOCUMENTS (page 6-25)
- ASSIGN_DOCINFO_FORM (page 5-24)
- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_PUBLISH (page 9-4)
- CRITERIAWORKFLOW_DISABLE (page 6-27)
- CRITERIAWORKFLOW_DISABLE_SUB (page 6-28)
- CRITERIAWORKFLOW_ENABLE (page 6-28)
- DELETE_COLLABORATION (page 2-4)
- DELETE_WFCONTRIBUTORS (page 6-28)
- DELETE_WORKFLOW (page 6-29)
- DELETE_WORKFLOWCRITERIA (page 6-30)
- DELETE_WORKFLOWDOCUMENTS (page 6-30)
- EDIT_COLLABORATION (page 2-4)
- EDIT_COLLABORATION_FORM (page 2-5)
- EDIT_WORKFLOW (page 6-32)
- EDIT_WORKFLOWCRITERIA (page 6-34)
- GET_CLBRA_INFO (page 2-5)
- GET_WORKFLOWDOCREVISIONS (page 6-42)
- GET_WORKFLOWDOCUMENTS (page 6-7)
- WORKFLOW_APPROVE (page 6-45)
- WORKFLOW_CANCEL (page 6-45)
- WORKFLOW_REJECT (page 6-50)
- WORKFLOW_REJECT_FORM (page 6-12)
- WORKFLOW_START (page 6-53)

setDateToPresent

Sets the current date and passes the specified parameter. Called as a Java method by these services:

- SUBSCRIBE_EX (page 5-89)
- UPDATE_SUBSCRIPTION_NOTIFY (page 5-110)
- UPDATE_SUBSCRIPTION_USED (page 5-111)

setDeleteRevReleaseState

Updates the revision status as deleted. Called as a Java method by these services:

- DELETE_BYREV (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)

setExternalDocInfoFields

Sets the external content item information fields. Called as a Java method by GET_EXTERNAL_DOC_INFO (page 5-170).

setFileConversionInfo

Called as a Java method by GET_DYNAMIC_CONVERSION (page 5-118) and GET_EXTERNAL_DYNAMIC_CONVERSION (page 5-122).

setInputConversionInfo

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB (page 5-120).

setLocalValues

Sets the associated local values and passes the specified parameters. Called as a Java method by these services:

- ADD_GROUP (page 5-181)
- ADD_PROBLEMREPORT (page 6-3)
- ADD_PROXIEDCOLLECTION (page 7-6)
- ADD_ROLE (page 5-181)
- ADD_USER (page 5-182)
- ASSIGN_DOCINFO_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_ARCHIVE_NO_NOTIFY (page 5-25)

- CHECKIN_LIST (page 5-30)
- CHECKIN_NEW_FORM (page 5-37)
- CHECKIN_PUBLISH (page 9-4)
- CHECKIN_SIMILAR_FORM (page 5-47)
- CLEAR_SERVER_OUTPUT (page 5-134)
- CONTINUE_CHECKIN (page 5-59)
- CONTINUE_SUBMIT_HTML_FORM (page 5-59)
- DELETE_CHECKIN_CACHE (page 5-61)
- DELETE_ROLE (page 5-190)
- EDIT_CLBRA_ACCESS_LIST_FORM (page 2-4)
- EDIT_USER (page 5-195)
- EDIT_USER_PROFILE (page 5-198)
- GET_ACTIVE_WORKFLOWS (page 6-37)
- GET_CLBRA_DOCUMENTS (page 2-5)
- GET_DOC_CONFIG_INFO (page 5-77)
- GET_EXPIRED (page 5-78)
- GET_EXTERNAL_DYNAMIC_CONVERSION (page 5-122)
- GET_SYSTEM_AUDIT_INFO (page 5-13)
- GET_WORKFLOWS (page 6-44)
- LOAD_PNE_PORTAL (page 5-202)
- LOAD_WORKFLOW_QUEUE (page 6-8)
- REGISTER_USER (page 5-204)
- REMOVE_TRANSFER (page 7-31)
- RESUBMIT_FOR_CONVERSION (page 5-82)
- REVIEW_WORKFLOW_DOC (page 6-10)
- UPDATE_BYREV (page 5-101)
- UPDATE_DOCINFO_BYREV (page 5-109)
- UPDATE_DOCINFO_SUB (page 5-110)
- VALIDATE_DOCINFO (page 5-112)
- WORK_IN_PROGRESS (page 5-112)
- WORKFLOW_APPROVE (page 6-45)

- WORKFLOW_CHECKIN_SUB (page 6-11)
- WORKFLOW_REJECT (page 6-50)

setOutputConversionInfo

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB (page 5-120).

setStatusMessage

Sets the status message and passes the specified parameter. Called as a Java method by these services:

- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- DELETE_BYCLASS (page 5-60)
- DELETE_BYNAME (page 5-60)
- DELETE_BYREV (page 5-61)
- DELETE_DOC (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)
- REPLACE_METAFILE_SUB (page 5-82)
- UPDATE_BYREV (page 5-101)
- WORKFLOW_CHECKIN_SUB (page 6-11)

setTemplateConversionInfo

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB (page 5-120).

setTraceFlags

Called as a Java method by EDIT_TRACE_OPTIONS (page 5-10).

startCriteriaWorkflow

Enables the criteria workflow. Called as a Java method by CRITERIAWORKFLOW_ENABLE (page 6-28).

startWorkflow

Starts the workflow process and passes *WfDocuments* as a parameter. Called as a Java method by WORKFLOW_START (page 6-53).

storeUserDatabaseProfileData

Stores the user profile information in the database. Called as a Java method by these services:

- ADD_USER (page 5-182)
- EDIT_USER (page 5-195)
- EDIT_USER_PROFILE (page 5-198)
- REGISTER_USER (page 5-204)

submitForm

Submits the form. Called as a Java method by FORM_SUBMIT (page 5-76).

submitHtmlForm

Submits the form. Called as a Java method by these services:

- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CONTINUE_SUBMIT_HTML_FORM (page 5-59)
- SUBMIT_HTML_FORM (page 5-85)

T

testProvider

Tests the provider. Called as a Java method by TEST_PROVIDER (page 5-162).

testWorkflowScript

Called as a Java method by TEST_WORKFLOW_SCRIPT (page 6-44).

U

Ualias

Updates an alias. Called as an Execute Query action by EDIT_ALIAS (page 5-191).

UcheckoutRevision

Updates the revision as checked out. Called as an Execute Query action by CHECKOUT_SUB (page 5-57).

Ucollaboration

Called as an Execute Query action by EDIT_CLBRA_ACCESS_LIST (page 2-4) and EDIT_COLLABORATION (page 2-4).

UdeleteRevision

Updates the revision status as deleted. Called as an Execute Query action by these services:

- DELETE_BYREV (page 5-61)
- DELETE_REV (page 5-62)
- DELETE_REV_EX (page 5-63)

UdocFormat

Updates the content item format information. Called as an Execute Query action by EDIT_DOCFORMAT (page 5-9).

UdocType

Updates the content item type. Called as an Execute Query action by EDIT_DOCTYPE (page 5-10).

UextensionMap

Updates the extension map. Called as an Execute Query action by EDIT_DOCEXTENSION (page 5-8).

Umeta

Updates the metadata information. Called as an Execute Query action by these services:

- UPDATE_BYREV (page 5-101)
- UPDATE_DOCINFO_BYREV (page 5-109)

- UPDATE_DOCINFO_SUB (page 5-110)
- UPDATE_METADATA (page 5-112)
- WORKFLOW_CHECKIN_SUB (page 6-11)

Umetadef

Updates the meta data definition information. Called as an Execute Query action by EDIT_METADEF (page 5-138).

UnextCounter

Updates the counter. Called as an Execute Query action by ADD_PROBLEMREPORT (page 6-3).

UnextRevID

Updates the revision ID. Called as an Execute Query action by these services:

- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- INSERT_NEW (page 7-28)

uninstallComponent

Called as a Java method by UNINSTALL_COMPONENT (page 5-135).

unregisterProject

Called as a Java method by UNREGISTER_PROJECT (page 9-8).

updateAffectedWorkflows

Called as a Java method by EDIT_COLLABORATION (page 2-4).

updateCache

Called as a Java method by UPDATE_FILTER_INFO (page 5-209).

updateCollaborationCache

Called as a Java method by these services:

- ADD_COLLABORATION (page 2-3)
- DELETE_COLLABORATION (page 2-4)

- EDIT_CLBRA_ACCESS_LIST (page 2-4)
- EDIT_COLLABORATION (page 2-4)

updateComponentConfig

Called as a Java method by UPDATE_COMPONENT_CONFIG (page 5-135).

updateConversionCache

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB (page 5-120).

updateDocFormatsConfig

Updates content item format configuration. Called as a Java method by EDIT_DOC_FORMATS (page 5-75).

updateDocInfo

Called as a Java method by UPDATE_DOCINFO_BYREV (page 5-109) and UPDATE_DOCINFO_SUB (page 5-110).

updateFilterConfig

Called as a Java method by UPDATE_FILTER_INFO (page 5-209).

updateLicense

Called as a Java method by UPDATE_LICENSE (page 5-135).

updateMetaTable

Updates the meta data information table. Called as a Java method by UPDATE_META_TABLE (page 5-142).

updateOptionList

Updates the option list table. Called as a Java method by UPDATE_OPTION_LIST (page 5-142).

updateProblemReport

Updates the problem report information in the database. Called as a Java method by UPDATE_PROBLEMREPORT (page 6-10).

updatePublishInfo

Called as a Java method by CHECKIN_PUBLISH (page 9-4).

updateResultTemplate

Updates the result template. Called as a Java method by UPDATE_RESULT_TEMPLATE (page 5-155).

updateRevisionIDAndLabel

Updates the revision ID and revision label. Called as a Java method by CHECKIN_SEL_FORM (page 5-45).

updateSubscriptionType

Updates the subscription type. Called as a Java method by UPDATE_SUBSCRIPTION_TYPE (page 5-110).

updateUserLocale

Called as a Java method by EDIT_USER_PROFILE (page 5-198).

updateUserMeta

Called as a Java method by UPDATE_USER_META (page 5-141).

updateUserMetaTable

Called as a Java method by UPDATE_USER_META_TABLE (page 5-141).

updateWorkflowAndDocState

Updates the workflow and content item state information. Called as a Java method by WORKFLOW_APPROVE (page 6-45) and WORKFLOW_REJECT (page 6-50).

updateWorkflowState

Updates the workflow state. Called as a Java method by DELETE_WORKFLOWDOCUMENTS (page 6-30).

updateWorkflowStateAfterCheckin

Called as a Java method by WORKFLOW_CHECKIN_SUB (page 6-11).

UproblemReport

Updates the problem report information in the database. Called as an Execute Query action by UPDATE_PROBLEMREPORT (page 6-10).

Urevision

Called as an Execute Query action by WORKFLOW_CHECKIN_SUB (page 6-11).

Urevision2

Called as an Execute Query action by these services:

- UPDATE_BYREV (page 5-101)
- UPDATE_DOCINFO_BYREV (page 5-109)
- UPDATE_DOCINFO_SUB (page 5-110)

UrevisionStatus

Updates the revision status. Called as an Execute Query action by WORKFLOW_REJECT (page 6-50).

UroleDefinition

Updates the role definition. Called as an Execute Query action by ADD_GROUP (page 5-181) and ADD_ROLE (page 5-181).

UsecurityGroup

Updates the security group. Called as an Execute Query action by EDIT_GROUP (page 5-194).

UsubscriptionNotification

Updates the subscription notification data. Called as an Execute Query action by UPDATE_SUBSCRIPTION_NOTIFY (page 5-110).

UsubscriptionUse

Updates the user subscription. Called as an Execute Query action by UPDATE_SUBSCRIPTION_USED (page 5-111).

Uuncheckout

Updates the checkout information. Called as an Execute Query action by these services:

- UNDO_CHECKOUT (page 5-92)
- UNDO_CHECKOUT_BY_NAME (page 5-95)
- UPDATE_BYREV (page 5-101)
- WORKFLOW_CHECKIN_SUB (page 6-11)

UuncheckoutPrevID

Updates the revision ID. Called as an Execute Query action by CHECKIN_BYNAME (page 5-26) and CHECKIN_SEL_SUB (page 5-47).

UuserAuthType

Updates the user authorization type. Called as an Execute Query action by CHANGE_USER_AUTH_TYPE (page 5-184).

Uworkflow

Updates the workflow. Called as an Execute Query action by EDIT_WORKFLOW (page 6-32).

UworkflowCriteria

Updates the workflow criteria. Called as an Execute Query action by EDIT_WORKFLOWCRITERIA (page 6-34).

UworkflowDocStep

Updates the content item workflow step. Called as an Execute Query action by WORKFLOW_REJECT (page 6-50).

UworkflowWithProject

Updates workflow project information. Called as an Execute Query action by EDIT_WORKFLOWCRITERIA (page 6-34).

V

validateCheckinData

Validates the check in data and passes the specified parameter. Called as a Java method by these services:

- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)
- UPDATE_BYREV (page 5-101)

- UPDATE_DOCINFO_BYREV (page 5-109)
- UPDATE_DOCINFO_SUB (page 5-110)
- VALIDATE_DOCINFO (page 5-112)
- WORKFLOW_CHECKIN_SUB (page 6-11)

validateCollaboration

Called as a Java method by these services:

- ADD_COLLABORATION (page 2-3)
- EDIT_CLBRA_ACCESS_LIST (page 2-4)
- EDIT_COLLABORATION (page 2-4)

validateDelete

Validates the delete request. Called as a Java method by DELETE_USER (page 5-191)

validateMetaData

Called as a Java method by UPDATE_METADATA (page 5-112).

validateProblemReport

Validates the problem report. Called as a Java method by ADD_PROBLEMREPORT (page 6-3) and UPDATE_PROBLEMREPORT (page 6-10).

validateStandard

Validates the standard data. Called as a Java method by these services:

- CACHE_CHECKIN_NEW (page 5-24)
- CACHE_CHECKIN_SEL (page 5-24)
- CACHE_SUBMIT_HTML_FORM (page 5-24)
- CACHE_WORKFLOW_CHECKIN (page 5-25)
- CHECKIN_BYNAME (page 5-26)
- CHECKIN_NEW_SUB (page 5-38)
- CHECKIN_SEL_SUB (page 5-47)
- INSERT_NATIVE (page 7-27)
- INSERT_NEW (page 7-28)
- UPDATE_BYREV (page 5-101)
- UPDATE_DOCINFO_BYREV (page 5-109)

- UPDATE_DOCINFO_SUB (page 5-110)
- VALIDATE_DOCINFO (page 5-112)
- WORKFLOW_CHECKIN_SUB (page 6-11)

validateSteps

Validates the steps for *WfSteps* and *QworkflowStepAliases*. Called as a Java method by CRITERIAWORKFLOW_DISABLE (page 6-27) and WORKFLOW_START (page 6-53).

validateUserNameAndType

Called as a Java method by ADD_USER (page 5-182) and CHANGE_USER_AUTH_TYPE (page 5-184).

viewDoc

Displays the content item and passes *SearchCollectionDocInfo* as a parameter. Called as a Java method by VIEW_DOC (page 5-177).

Appendix

B

THIRD PARTY LICENSES

OVERVIEW

This appendix includes a description of the Third Party Licenses for all the third party products included with this product.

- Apache Software License (page B-1)
- W3C® Software Notice and License (page B-2)
- Zlib License (page B-3)
- General BSD License (page B-4)
- General MIT License (page B-5)
- Unicode License (page B-5)
- Miscellaneous Attributions (page B-7)

APACHE SOFTWARE LICENSE

```
* Copyright 1999-2004 The Apache Software Foundation.
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*      http://www.apache.org/licenses/LICENSE-2.0
*
```

```
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
```

W3C® SOFTWARE NOTICE AND LICENSE

```
* Copyright © 1994-2000 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 W3C work (including software, documents, or other related items) is
* being provided by the copyright holders under the following license. By
* obtaining, using and/or copying this work, you (the licensee) agree that
* you have read, understood, and will comply with the following terms and
* conditions:
*
* Permission to use, copy, modify, and distribute this software and its
* documentation, with or without modification, for any purpose and without
* fee or royalty is hereby granted, provided that you include the following
* on ALL copies of the software and documentation or portions thereof,
* including modifications, that you make:
*
*   1. The full text of this NOTICE in a location viewable to users of the
*      redistributed or derivative work.
*
*   2. Any pre-existing intellectual property disclaimers, notices, or terms
*      and conditions. If none exist, a short notice of the following form
*      (hypertext is preferred, text is permitted) should be used within the
*      body of any redistributed or derivative code: "Copyright ©
*      [$date-of-software] 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/"
*
*   3. Notice of any changes or modifications to the W3C files, including the
*      date changes were made. (We recommend you provide URIs to the location
*      from which the code is derived.)
*
* THIS SOFTWARE AND DOCUMENTATION IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS
* MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
* NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR
* PURPOSE OR THAT THE USE OF THE SOFTWARE OR DOCUMENTATION WILL NOT INFRINGE
* ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.
*
* COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR
* CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE SOFTWARE OR
* DOCUMENTATION.
*
* The name and trademarks of copyright holders may NOT be used in advertising
* or publicity pertaining to the software without specific, written prior
* permission. Title to copyright in this software and any associated
* documentation will at all times remain with copyright holders.
*
```

ZLIB LICENSE

* zlib.h -- interface of the 'zlib' general purpose compression library
  version 1.2.3, July 18th, 2005

Copyright (C) 1995-2005 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.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly jloup@gzip.org

Mark Adler madler@alumni.caltech.edu

GENERAL BSD LICENSE

Copyright (c) 1998, Regents of the University of California

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

"Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

"Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

"Neither the name of the <ORGANIZATION> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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.

GENERAL MIT LICENSE

Copyright (c) 1998, Regents of the Massachusetts Institute of Technology

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

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.

UNICODE LICENSE

UNICODE, INC. LICENSE AGREEMENT - DATA FILES AND SOFTWARE

Unicode Data Files include all data files under the directories http://www.unicode.org/Public/, http://www.unicode.org/reports/, and http://www.unicode.org/cldr/data/ . Unicode Software includes any source code published in the Unicode Standard or under the directories http://www.unicode.org/Public/, http://www.unicode.org/reports/, and http://www.unicode.org/cldr/data/.

NOTICE TO USER: Carefully read the following legal agreement. BY DOWNLOADING, INSTALLING, COPYING OR OTHERWISE USING UNICODE INC.'S DATA FILES ("DATA FILES"), AND/OR SOFTWARE ("SOFTWARE"), YOU UNEQUIVOCALLY ACCEPT, AND AGREE TO BE BOUND BY, ALL OF THE TERMS AND CONDITIONS OF THIS AGREEMENT. IF YOU DO NOT AGREE, DO NOT DOWNLOAD, INSTALL, COPY, DISTRIBUTE OR USE THE DATA FILES OR SOFTWARE.

COPYRIGHT AND PERMISSION NOTICE

Copyright © 1991-2006 Unicode, Inc. All rights reserved. Distributed under the Terms of Use in http://www.unicode.org/copyright.html.

Permission is hereby granted, free of charge, to any person obtaining a copy of the Unicode data files and any associated documentation (the "Data Files") or Unicode software and any associated documentation (the "Software") to deal in the Data Files or Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Data Files or Software, and to permit persons to whom the Data Files or Software are furnished to do so, provided that (a) the above copyright notice(s) and this permission notice appear with all copies of the Data Files or Software, (b) both the above copyright notice(s) and this permission notice appear in associated documentation, and (c) there is clear notice in each modified Data File or in the Software as well as in the documentation associated with the Data File(s) or Software that the data or software has been modified.

THE DATA FILES AND SOFTWARE ARE 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 OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR 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 THE DATA FILES OR SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in these Data Files or Software without prior written authorization of the copyright holder.

_____Unicode and the Unicode logo are trademarks of Unicode, Inc., and may be registered in some jurisdictions. All other trademarks and registered trademarks mentioned herein are the property of their respective owners

MISCELLANEOUS ATTRIBUTIONS

Adobe, Acrobat, and the Acrobat Logo are registered trademarks of Adobe Systems Incorporated.

FAST Instream is a trademark of Fast Search and Transfer ASA.

HP-UX is a registered trademark of Hewlett-Packard Company.

IBM, Informix, and DB2 are registered trademarks of IBM Corporation.

Jaws PDF Library is a registered trademark of Global Graphics Software Ltd.

Kofax is a registered trademark, and Ascent and Ascent Capture are trademarks of Kofax Image Products.

Linux is a registered trademark of Linus Torvalds.

Mac is a registered trademark, and Safari is a trademark of Apple Computer, Inc.

Microsoft, Windows, and Internet Explorer are registered trademarks of Microsoft Corporation.

MrSID is property of LizardTech, Inc. It is protected by U.S. Patent No. 5,710,835. Foreign Patents Pending.

Oracle is a registered trademark of Oracle Corporation.

Portions Copyright © 1994-1997 LEAD Technologies, Inc. All rights reserved.

Portions Copyright © 1990-1998 Handmade Software, Inc. All rights reserved.

Portions Copyright © 1988, 1997 Aladdin Enterprises. All rights reserved.

Portions Copyright © 1997 Soft Horizons. All rights reserved.

Portions Copyright © 1995-1999 LizardTech, Inc. All rights reserved.

Red Hat is a registered trademark of Red Hat, Inc.

Sun is a registered trademark, and Sun ONE, Solaris, iPlanet and Java are trademarks of Sun Microsystems, Inc.

Sybase is a registered trademark of Sybase, Inc.

UNIX is a registered trademark of The Open Group.

Verity is a registered trademark of Autonomy Corporation plc

Index

A

access level attribute, 4-6
action format, 4-12, A-1
actions
    control mask, 4-15
    error message, 4-18
    name, 4-14
    parameters, 4-15
    service resource, 4-21
    type, 4-13
Actions column
    service ResultSet, 4-12
ACTIVATE_SERVICE, 10-4
ADD_ALIAS, 5-179
ADD_ARCHIVE, 7-5
ADD_COLLECTION, 7-5
ADD_DOC_ACCOUNT, 5-4
ADD_DOCEXTENSION, 5-4
ADD_DOCFORMAT, 5-5
ADD_DOCPROFILE, 5-113
ADD_DOCRULE, 5-114
ADD_DOCTYPE, 5-6
ADD_EDIT_PROVIDER, 5-157
ADD_GROUP, 5-181
ADD_METADEF, 5-136
ADD_PROBLEMREPORT, 6-3
ADD_PROXIEDCOLLECTION, 7-6, 7-6
ADD_ROLE, 5-181
ADD_SCHEMA_FIELD, 5-163
ADD_SCHEMA_RELATION, 5-164
ADD_SCHEMA_VIEW, 5-164
ADD_USER, 5-182
ADD_WEB_APP, 5-118, 7-6
ADD_WF_TEMPLATE, 6-13, 7-6
ADD_WORKFLOW, 6-22
ADD_WORKFLOW_SCRIPT, 6-23
ADD_WORKFLOW_TOKEN, 6-23
ADD_WORKFLOWALIASES, 6-24
ADD_WORKFLOWDOCUMENT_SUB, 6-25
ADD_WORKFLOWDOCUMENTS, 6-25
ADD_WORKFLOWSTEP, 6-26
addAliases, A-2
addCollaboration, A-2
addContext, A-2
addFiles, A-2
ADDOREDIT_SCHEMA_TABLE, 5-164
addOrEditDocMetaData, A-2
addOrEditProvider, A-2
addOrEditSchemaTable, A-2
addProblemReport, A-3
addRegisteredUserAttributes, A-3
addSchemaRelation, A-3
addSchemaView, A-3
addSubscription, A-3
addTemplate, A-3
addUserAttributes, A-3
addWfDocuments, A-3
addWorkflow, A-3
addWorkflowScript, A-3
addWorkflowStep, A-4
addWorkflowStepScript, A-4
addWorkflowToken, A-4
Alias, A-4
AliasUserMap, A-4
allowProblemReportAction, A-4
APPEND_DATABASE_AUDIT_INFO, 5-158
APPEND_FILE_CACHING_INFO, 5-118
APPEND_SEARCH_AUDIT_INFO, 5-170
appendCommonSystemInfo, A-4
appendDatabaseAuditMessage, A-4
appendFileCachingMessage, A-4
APPLET_DOCINFO, 5-6, 7-6
APPROVE_DELETE_SERVICE, 10-5
approveDoc, A-4
archive service sample, 7-1
archive services
    overview, 7-1
ASSIGN_DOCINFO_FORM, 5-24
attributes
    service resources, 4-20
Attributes column
    service ResultSet, 4-3
authentication challenges, 3-10

B

blDateFormat, 3-8
blFieldTypes, 3-8
BROWSE_CATEGORY_FORM, 10-5
BROWSE_FOLDER_FORM, 10-5
BROWSE_SERIES_FORM, 10-5
buildAllWebStringFiles, A-5
buildExpiredContentQuery, A-5
buildPreviewList, A-5
buildSearchIndex, A-5
buildSourceInfo, A-5

C

CACHE_CHECKIN_NEW, 5-24
CACHE_CHECKIN_SEL, 5-24
CACHE_SUBMIT_HTML_FORM, 5-24
CACHE_WORKFLOW_CHECKIN, 5-25
cacheCheckin, A-5
cacheTemplates, A-5
CANCEL_ARCHIVE, 7-6, 7-6
CANCEL_COMPONENT_INSTALL, 5-134
CANCEL_SEARCH_INDEX, 5-124
cancelComponentInstall, A-6
cancelCriteriaWorkflow, A-6
cancelSearchIndex, A-6
cancelWorkflow, A-6
canDeleteStep, A-6
CHANGE_USER_AUTH_TYPE, 5-184
CHECK_USER_CREDENTIALS, 5-185
checkCanCreateDocSecurity, A-6
checkCollaborationAccess, A-7
checkConversionCache, A-7
checkCounters, A-7
checkCriteriaWorkflow, A-7
checkDocRules, A-7
checkDocState, A-8
checkForceLogin, A-8
checkForPublish, A-8
checkForRefreshingCachedResources, A-8
CHECKIN_ARCHIVE, 7-6
CHECKIN_ARCHIVE_NO_NOTIFY, 5-25
CHECKIN_BYNAME, 5-26
CHECKIN_CONFIRM_FORM, 5-30
CHECKIN_LIST, 5-30
CHECKIN_NEW, 5-31
CHECKIN_NEW_FORM, 5-37
CHECKIN_NEW_REVISION_SERVICE, 10-5
CHECKIN_NEW_SUB, 5-38
CHECKIN_PUBLISH, 9-4
CHECKIN_SEL, 5-40
CHECKIN_SEL_FORM, 5-45
CHECKIN_SEL_SUB, 5-47
CHECKIN_SIMILAR_FORM, 5-47, 10-6
CHECKIN_UNIVERSAL, 5-48
checkInByID, A-8
checkInByName, A-8
checkIsLatestRev, A-8
checkIsSelf, A-8
CHECKOUT, 5-52
CHECKOUT_BY_NAME, 5-55
CHECKOUT_OK, 5-56
CHECKOUT_SUB, 5-57
checkParametersAgainstResultSet, A-9
checkProblemReportSecurity, A-9
checkRefreshUsers, A-9
checkRevisionProperties, A-9
checkSecurity, A-9
checkSubAdmin, A-12
checkUserAuthType, A-13
checkWorkflow, A-13
checkWorkflowAdminAccess, A-14
CHUNKED_UPLOAD, 5-143
CLEAR_FOLDER_CANCELLED_DATE, 10-6
CLEAR_FOLDER_EXPIRATION_DATE, 10-6
CLEAR_FOLDER_OSBOLETE_DATE, 10-6
CLEAR_FOLDER_RESCINDED_DATE, 10-6
CLEAR_FOLDER_REVIEW_DATE, 10-7
CLEAR_RECORD_CANCELLED_DATE, 10-7
CLEAR_RECORD_EXPIRATION_DATE, 10-7
CLEAR_RECORD_OBSOLETE_DATE, 10-7
CLEAR_SERVER_OUTPUT, 5-134
ClientEncoding, 3-8
CLOSE_FOLDER, 10-7
COLLECTION_ADD, 8-5
COLLECTION_ADD_LINK, 8-6
COLLECTION_ADD_MOUNT, 8-6
COLLECTION_BROWSE, 8-7
COLLECTION_CHECKIN_NEW, 8-7
COLLECTION_CHECKIN_REVISION, 8-7
COLLECTION_CHECKIN_SEL_SUB, 8-8
COLLECTION_COPY_ALL, 8-8
COLLECTION_COPY_COLLECTION, 8-8
COLLECTION_COPY_ITEM, 8-9
COLLECTION_COPY_LOT, 8-9
COLLECTION_DELETE_ALL, 8-9
COLLECTION_DELETE_COLLECTION, 8-9
COLLECTION_DELETE_ITEM, 8-9
COLLECTION_DELETE_LOT, 8-10
COLLECTION_DISPLAY, 8-10
COLLECTION_EDIT, 8-11
COLLECTION_GET_ADMIN_CONFIG, 8-11
COLLECTION_GET_ADMIN_INHERIT_CONFIG, 8-11
COLLECTION_GET_ADMIN_MARKED_CONFIG, 8-11
COLLECTION_GET_ADMIN_META_CONFIG, 8-12

COLLECTION_GET_ADMIN_METADATA_DEFAULTS, 8-12
COLLECTION_GET_ADMIN_MOUNTED_CONFIG, 8-12
COLLECTION_GET_ARCHIVE, 8-12
COLLECTION_GET_BRANCH, 8-13
COLLECTION_GET_COLLECTION, 8-13
COLLECTION_GET_COLLECTIONS, 8-13
COLLECTION_GET_CONTENT_FILE, 8-13
COLLECTION_GET_CONTENTS, 8-14
COLLECTION_GET_FILE, 8-14
COLLECTION_GET_INFO, 8-14
COLLECTION_GET_LINKS, 8-14
COLLECTION_GET_META_MAPPING, 8-14
COLLECTION_GET_PROFILE_METADATA_DEFAULTS, 8-15
COLLECTION_GET_PROFILE_METADATA_REVISION_DEFAULTS, 8-15
COLLECTION_GET_REFERENCE, 8-15
COLLECTION_GET_SYSTEM_FILE, 8-16
COLLECTION_GET_USER_CONFIG, 8-16
COLLECTION_INDEX_MOUNTS, 8-16
COLLECTION_INFO, 8-16
COLLECTION_ISVALID_MET, 8-17
COLLECTION_LOCK, 8-17
COLLECTION_MOVE_ALL, 8-17
COLLECTION_MOVE_COLLECTION, 8-17
COLLECTION_MOVE_ITEM, 8-17
COLLECTION_MOVE_LOT, 8-18
COLLECTION_NEW, 8-18
COLLECTION_PROFILE_UPDATE_COLUMNS, 8-19
COLLECTION_REBUILD_MOUNT, 8-19
COLLECTION_REMOVE_MOUNT, 8-19
COLLECTION_RESTORE_COLLECTION, 8-19
COLLECTION_RESTORE_ITEM, 8-20
COLLECTION_SEARCH_CONTENT, 8-20
COLLECTION_SEARCH_RESULTS, 8-20
COLLECTION_SET_ARCHIVE, 8-20
COLLECTION_SET_USER_CONFIG, 8-20
COLLECTION_UNLOCK, 8-21
COLLECTION_UPDATE, 8-21
COLLECTION_UPDATE_ADMIN_CONFIG, 8-22
COLLECTION_UPDATE_ADMIN_INHERIT_CONFIG, 8-22
COLLECTION_UPDATE_ADMIN_METADATA_DEFAULTS, 8-22
COLLECTION_UPDATE_ALL, 8-22
COLLECTION_UPDATE_ITEM, 8-22
COLLECTION_UPDATE_META, 8-23
COLLECTION_UPDATE_META_TABLE, 8-23
COLLECTION_UPDATE_MOUNTS, 8-23
COLLECTION_UPDATE_PROFILE_METADATA_DEFAULTS, 8-23
COLLECTION_UPDATE_PROFILE_METADATA_REVISION_DEFAULTS, 8-24
COLLECTION_UPDATE_STRUCTURE, 8-24
Component Wizard
  creating service resources, 4-29
computeCompanionDirectory, A-14
computeDeleteSecurity, A-14
computeDocID, A-14
computeDocInfoInHtmlPage, A-14
computeDocSubscribers, A-14
computeProjectInfo, A-15
computeRejectTargetStep, A-15
computeRemainingUsers, A-15
computeURLs, A-15
computeWfDocumentsInfo, A-15
conditionalDetermineCheckin, A-15
CONFIG_INFO, 5-7
CONTINUE_CHECKIN, 5-59
CONTINUE_SUBMIT_HTML_FORM, 5-59
continueCheckin, A-15
control mask, 4-15
CONTROL_SCHEMA, 5-164
CONTROL_SEARCH_INDEX, 5-125
controlIndexing, A-15
convertDatabaseDate, 3-9
COPY_ARCHIVE, 7-11
CREATE_FOLDER, 10-8
CREATE_FOLDER_FORM, 10-8
CREATE_SUBSCRIPTION_TYPE, 5-60
createAddEditProviderForm, A-15
createArchiveFileName, A-16
createFileName, A-16
createNewRev, A-16
createResultSetSQL, A-16
createSubscriptionType, A-16
createWebFileNameFromRelativeUrl, A-17
createWorkflowID, A-17
creating
  service resource, 4-29
CRITERIAWORKFLOW_DISABLE, 6-27
CRITERIAWORKFLOW_DISABLE_SUB, 6-28
CRITERIAWORKFLOW_ENABLE, 6-28
customizing local parameters, 3-8

D

Dalias, A-17
DaliasUsers, A-17
Date and Time Formatting, 3-3
Dcollaboration, A-17
DdocAccount, A-17
DdocFormat, A-17
DdocType, A-17

Ddocument, A-17
decodeTopicValues, A-18
DEL_METADEF, 5-138
DELETE_ALIAS, 5-189
DELETE_ALL_BUT_LAST_N_REVISIONS_SERVICE, 10-8
DELETE_ARCHIVE, 7-11
DELETE_BATCH_FILE, 7-12
DELETE_BATCH_FILE_DOCUMENTS, 7-12
DELETE_BATCH_FILE_TABLES, 7-13
DELETE_BYCLASS, 5-60
DELETE_BYNAME, 5-60
DELETE_BYREV, 5-61
DELETE_BYREV_REVISION, 5-61
DELETE_CHECKIN_CACHE, 5-61
DELETE_DOC, 5-61
DELETE_DOC_ACCOUNT, 5-7
DELETE_DOCEXTENSION, 5-7
DELETE_DOCFORMAT, 5-8
DELETE_DOCPROFILE, 5-114
DELETE_DOCRULE, 5-115
DELETE_DOCTYPE, 5-8
DELETE_FOLDER, 10-9
DELETE_GROUP, 5-190
DELETE_PROBLEMREPORT, 6-4
DELETE_PROVIDER, 5-158
DELETE_RESULT_TEMPLATE, 5-145
DELETE_REV, 5-62
DELETE_REV_EX, 5-63
DELETE_REVISION_SERVICE, 10-9
DELETE_ROLE, 5-190
DELETE_SCHEMA_FIELD, 5-164
DELETE_SCHEMA_RELATION, 5-165
DELETE_SCHEMA_TABLE, 5-165
DELETE_SCHEMA_VIEW, 5-165
DELETE_SUBSCRIPTION_TYPE, 5-64
DELETE_USER, 5-191
DELETE_WF_TEMPLATE, 6-13
DELETE_WFCONTRIBUTORS, 6-28
DELETE_WORKFLOW, 6-29
DELETE_WORKFLOW_SCRIPT, 6-29
DELETE_WORKFLOW_TOKEN, 6-30
DELETE_WORKFLOWCRITERIA, 6-30
DELETE_WORKFLOWDOCUMENTS, 6-30
DELETE_WORKFLOWSTEP, 6-32
deleteAliases, A-18
deleteCollaboration, A-18
deleteDoc, A-18
deleteDocumentSubscription, A-18
deleteProblemReport, A-18
deleteProvider, A-18
deleteResultTemplate, A-18
deleteRev, A-19
deleteRevFiles, A-19
deleteSchemaRelation, A-19
deleteSchemaTable, A-19
deleteSchemaView, A-19
deleteSubscriptionType, A-19
deleteTemplate, A-19
deleteWfDesign, A-19
deleteWfDocuments, A-19
deleteWorkflowScript, A-20
deleteWorkflowStepScript, A-20
deleteWorkflowToken, A-20
determineCheckin, A-20
DextensionMap, A-20
DgroupRole, A-20
disableSendFile, A-20
Dmetadef, A-20
DOC_FORMATS_WIZARD, 5-64
DOC_INFO, 5-64
DOC_INFO_BY_NAME, 5-68
DOC_INFO_LATESTRELEASE, 5-70
DOC_INFO_SIMPLE, 5-71
DOC_INFO_SIMPLE_BYREV, 5-72
DOC_SUBS_LIST, 5-73
doCachedCheckinCleanup, A-20
doCachedCleanup, A-21
DocFormats, A-21
docHistoryInfo, A-21
DOCPROFILE_PREVIEW, 5-115
docRefinery, A-22
documentation overview, 2-1
doSubService, A-22
doSubserviceIfMetafile, A-23
doUpload, A-23
DOWNLOAD_COMPONENT, 5-134
DOWNLOAD_LISTBOX_ITEMS, 5-144
downloadComponent, A-23
downloadItems, A-23
doWorkflowAction, A-23
DprjDocumentsDeleted, A-23
DproblemReport, A-23
Drole, A-23
DsecurityGroup, A-23
DsubscriptionType, A-24
Duser, A-24
DuserSecurityAttributes, A-24
DuserSubscription, A-24
Dworkflow, A-24
DworkflowAliasAll, A-24
DworkflowCriteria, A-24
DworkflowDocState, A-24
DworkflowDocumentAll, A-24
DworkflowStateAll, A-25
DworkflowStep, A-25
DworkflowStepAliases, A-25
DworkflowStepsAll, A-25

DworkflowUserAttributeAll, A-25
dynamic page, 3-4

E

EDIT_ALIAS, 5-191
EDIT_ARCHIVE, 7-13
EDIT_ARCHIVEDATA, 7-14
EDIT_DOC_FORMATS, 5-75
EDIT_DOCEXTENSION, 5-8
EDIT_DOCFORMAT, 5-9
EDIT_DOCPROFILE, 5-115
EDIT_DOCPROFILE_TRIGGER, 5-116
EDIT_DOCRULE, 5-116
EDIT_DOCTYPE, 5-10
EDIT_EXPORTERS, 7-14
EDIT_FOLDER, 10-9
EDIT_FOLDER_FORM, 10-9
EDIT_GROUP, 5-194
EDIT_METADEF, 5-138
EDIT_ROLE, 5-194
EDIT_SCHEMA_FIELD, 5-165
EDIT_SCHEMA_NODE, 5-166
EDIT_SCHEMA_RELATION, 5-166
EDIT_SCHEMA_VIEW, 5-166
EDIT_SCHEMA_VIEW_VALUES, 5-167
EDIT_TRACE_OPTIONS, 5-10
EDIT_TRANSFEROPTIONS, 7-15
EDIT_USER, 5-195
EDIT_USER_PROFILE, 5-198
EDIT_WF_TEMPLATE, 6-14
EDIT_WORKFLOW, 6-32
EDIT_WORKFLOW_SCRIPT, 6-33
EDIT_WORKFLOW_TOKEN, 6-34
EDIT_WORKFLOWCRITERIA, 6-34, 6-34
EDIT_WORKFLOWSTEP, 6-35
editCollaboration, A-25
editCriteria, A-26
editRole, A-26
editSchemaRelation, A-26
editSchemaView, A-26
editSchemaViewValues, A-26
editTemplate, A-26
editWorkflowScript, A-26
editWorkflowStep, A-26
editWorkflowStepScript, A-26
editWorkflowToken, A-26
ENABLE_DISABLE_PROVIDER, 5-159
enableDisableProvider, A-26
END_PUBLISHING, 9-4
error message, 4-18
error message attribute, 4-11
examples
    service actions, 4-21
    service attributes, 4-20
    service resource, 4-2, 4-18
EXECUTE_BATCH, 7-15
executeArchiveMethod, A-27
executeCommands, A-28
executeManifest, A-28
executePageService, A-28
EXPORT_ARCHIVE, 7-15
ExtensionFormatMap, A-28
extraFieldTypes, 3-9

F

filterUpdateData, A-28
folder service sample, 8-1
folder services overview, 8-1
FORM_PROCESS, 5-76
FORM_SUBMIT, 5-76
format
    action, 4-12, A-1
FREEZE_FOLDER, 10-10
FREEZE_RECORD, 10-10

G

generic services
    workflows, 6-2
GET_ACTIVE_WORKFLOWS, 6-37
GET_ADD_EDIT_PROVIDER_FORM, 5-160
GET_ADMIN_PAGE, 5-145
GET_ADVANCED_SEARCH_OPTIONS, 5-139
GET_ALIASES, 5-200
GET_ALL_PROVIDERS, 5-161
GET_ALL_WORKFLOWDOCREVISIONS, 6-37
GET_ARCHIVECOLLECTIONS, 7-17
GET_ARCHIVED_FILE, 7-17
GET_ARCHIVERELATIONQUERY, 7-19
GET_ARCHIVES, 7-19
GET_ARCHIVETABLECONTENT, 7-17
GET_BATCH_FILE_DOCUMENTS, 7-19
GET_BATCH_PROPERTIES, 7-20
GET_BATCH_SCHEMA, 7-20
GET_BATCH_VALUES, 7-21
GET_BATCHFILES, 7-21
GET_CACHED_CHECKIN_INFO, 5-77
GET_COMPONENT_CONFIG, 5-134
GET_COMPONENT_INSTALL_FORM, 5-134
GET_COMPONENT_INSTALL_PROMPTS_FORM, 5-1
    34
GET_COMPONENT_INSTALL_SETTINGS, 5-134

GET_CRITERIA_WORKFLOWS_FOR_GROUP, 6-5, 6-38
get_criteria_workflows_for_group, 6-38
GET_DATARESULTSET, 5-10
GET_DOC_CONFIG_INFO, 5-77
GET_DOC_METADATA_INFO, 5-139
GET_DOC_PAGE, 5-146
GET_DOC_SUBSCRIBERS, 5-78
GET_DOCEXTENSIONS, 5-11
GET_DOCFORMATS, 5-11
GET_DOCPROFILE, 5-116
GET_DOCPROFILES, 5-117
GET_DOCRULE, 5-117
GET_DOCRULES, 5-117
GET_DOCTYPES, 5-11
GET_DOCUMENT_PROBLEMREPORTS, 6-5
GET_DYNAMIC_CONVERSION, 5-118
GET_DYNAMIC_CONVERSION_SUB, 5-120
GET_DYNAMIC_PAGE, 5-149
GET_DYNAMIC_URL, 5-121
GET_ENVIRONMENT, 8-25
GET_EXPIRED, 5-78
GET_EXTERNAL_DOC_INFO, 5-170
GET_EXTERNAL_DYNAMIC_CONVERSION, 5-122
GET_EXTERNAL_HIGHLIGHT_INFO, 5-171
GET_EXTERNAL_XML_HIGHLIGHT_INFO, 5-171
GET_FILE, 5-129
GET_FILELIST, 5-12
GET_FILTER_ADMIN_PAGE, 5-200
GET_HIGHLIGHT_INFO, 5-172
GET_LOCAL_REGISTRATION_FORM, 5-135
GET_MANIFEST_INFO, 5-135
GET_METADEFS, 5-12
GET_OPTION_LIST, 5-140
GET_OPTION_LISTS, 8-24
GET_PACKAGE_ENVIRONMENT_PAGE, 5-79
GET_PORTAL_PAGE, 5-150
GET_PREVIEW_LIST, 9-5
GET_PREVIEW_LIST_EX, 9-5
GET_PROBLEMREPORTS_SEARCH_ FORM, 6-6
GET_PROBLEMREPORTS_SEARCH_ RESULTS, 6-6
GET_PROJECT_INFO, 9-6
GET_PROJECT_REGISTRATION, 9-6
GET_PROVIDER_INFO, 5-161
GET_PROXIED_ARCHIVECOLLECTIONS, 7-22
GET_PROXIEDSERVERS, 7-22
GET_REGISTEREDPROJECTS, 9-7
GET_REPLICATION_DATA, 7-22
GET_RESULT_OPTIONS, 5-12
GET_SCHEMA_FIELD_INFO, 5-167
GET_SCHEMA_FIELDS, 5-167
GET_SCHEMA_RELATIONS, 5-168
GET_SCHEMA_STATS, 5-167
GET_SCHEMA_TABLE_INFO, 5-168
GET_SCHEMA_TABLES, 5-168
GET_SCHEMA_VIEW_EDIT_INFO, 5-168
GET_SCHEMA_VIEW_FRAGMENT, 5-168
GET_SCHEMA_VIEW_INFO, 5-169
GET_SCHEMA_VIEW_VALUES, 5-169
GET_SCHEMA_VIEWS, 5-169
GET_SEARCH_RESULTS, 5-174
GET_SEARCH_RESULTS_FORCELOGIN, 5-175
GET_SECURE_PAGE, 5-151
GET_SELF_REGISTER_PAGE, 5-200
GET_SERVER_OUTPUT, 5-135
GET_SYSTEM_AUDIT_INFO, 5-13, 5-200
GET_TABLE, 5-13
GET_TABLECOLUMNLIST, 7-23
GET_TARGET_INFO, 7-23
GET_TARGET_TRANSFER_STATUS, 7-23
GET_TEMPLATE_CONVERSIONS, 5-122
GET_TRANSFER_SOURCE_INFO, 7-24
GET_UPDATE_FORM, 5-80
GET_UPDATE_PROBLEMREPORT_FORM, 6-7
GET_USER_INFO, 5-200
GET_USER_METADATA_INFO, 5-140
GET_USER_METADEFS, 5-14
GET_USERS, 5-202
GET_WEB_APP_STATUS, 5-122
GET_WF_COMPANION_INFO, 6-38
GET_WF_TEMPLATE, 6-20
GET_WF_TEMPLATES, 6-20
GET_WORKFLOW, 6-39
GET_WORKFLOW_INFO, 6-39
GET_WORKFLOW_INFO_BYNAME, 6-40
GET_WORKFLOW_SCRIPT, 6-42
GET_WORKFLOWDOCREVISIONS, 6-42
GET_WORKFLOWDOCUMENTS, 6-7
GET_WORKFLOWS, 6-44
GET_WORKFLOWS_FOR_ALL, 6-8
GET_XML_HIGHLIGHT_INFO, 5-175
getCollaborationData, A-28
getCollaborationDocs, A-29
getCollaborations, A-29
getCompanionHistory, A-29
getComponentConfig, A-29
getDefaultDocFormats, A-29
getDocFormats, A-29
getDocSubscriptionInfo, A-29
getDocumentList, A-30
getDynamicPage, A-30
getExternalDocInfo, A-30
getExternalSecurityInfo, A-30
getFilesInAppDir, A-30
getHighlightInfo, A-30
getLatestID, A-30
getLatestIDRevInfo, A-31
getOptionList, A-31

getOutgoingProviders, A-31
getProxiedArchiveCollections, A-31
getSchemaRelations, A-31
getSchemaTableInfo, A-31
getSchemaTables, A-31
getSchemaViewInfo, A-31
getSchemaViews, A-31
getSearchResults, A-32
getSubscriptionList, A-32
getTable, A-32
getTemplate, A-32
getTemplateConversions, A-32
getTemplates, A-32
getURLAbsolute, A-32
getUserAttributes, A-32
getUserCollaborationList, A-32
getUserMailAddress, A-32
getUserProfile, A-33
getUsers, A-33
getUserUnique, A-33
getViewEditInfo, A-33
getViewValues, A-33
getWebAppStatus, A-33
getWfDocuments, A-33
getWorkflowDesignInfo, A-34
getWorkflowDocumentInfo, A-34
getWorkflowInfo, A-34
getWorkflowScript, A-34
getWorkflowStepAliasesInfo, A-34
GOTO_COLLECTION, 8-24
GOTO_ROOT_COLLECTION, 8-24

H

HEADER_ENCODING, 3-8
hidePassword, A-34

I

Ialias, A-34
Icollaboration, A-34
IdocAccount, A-35
IdocFormat, A-35
IdocType, A-35
IextensionMap, A-35
Imeta, A-35
Imetadef, A-35
IMPORT_ARCHIVE, 7-24
IMPORT_ARCHIVE_START_AUTOMATED, 7-26
IMPORT_BATCHFILE, 7-26
IMPORT_DOCUMENT, 7-26
IMPORT_TABLE_ENTRY, 7-27
INFO_CATEGORY_FORM, 10-10
INFO_FOLDER_FORM, 10-11
INFO_FOLDER_LIFECYCLE, 10-11
INFO_FOLDER_METADATA_HISTORY, 10-11
INFO_FOLDER_REVIEW_HISTORY, 10-11
INFO_RECORD_LIFECYCLE, 10-11
INFO_RECORD_METADATA_HISTORY, 10-12
INFO_RECORD_REVIEW_HISTORY, 10-12
INFO_SERIES_FORM, 10-12
INSERT_NATIVE, 7-27, 7-28
INSERT_NEW, 7-28
insertAliasUsers, A-35
insertGroupRow, A-35
IproblemReport, A-36
Irevision, A-36
Irole, A-36
isDocCheckedOut, A-36
IsecurityGroup, A-36
Isubscription, A-36
Iworkflow, A-36
IworkflowDocument, A-36
IworkflowState, A-37

J

JAVA_PROPERTIES, 5-14

L

LM_BUILD_WEB_STRING_FILES, 5-14
LM_LOAD_LAYOUTS, 5-14
LM_LOAD_LAYOUTS_SUB, 5-14
LOAD_DOC_ENVIRONMENT, 5-14
LOAD_GLOBALINCLUDES, 5-153
LOAD_PNE_PORTAL, 5-202
LOAD_RESOURCE_FILE, 5-122
LOAD_USER_LOCALIZATION, 5-144
LOAD_USER_TOPIC, 5-203
LOAD_WORKFLOW_QUEUE, 6-8
loadActiveComponentData, A-37
loadAndValidateValues, A-37
loadCollaborationList, A-37
loadComponentInstallInfo, A-37
loadComponentInstallSettings, A-37
loadConfigurationInfo, A-37
loadDefaultInfo, A-37
loadDocConfig, A-38
loadDocDefaults, A-38
loadFilterConfig, A-38
loadGlobalIncludes, A-38
loadMetaDefaults, A-38
loadMetaOptionsLists, A-39

loadPRStateLists, A-39
loadRegisteredProjects, A-39
loadServerOutput, A-39
loadSharedTable, A-39
loadTopic, A-40
loadTraceFlags, A-40
loadUserAndCheckEditAllowed, A-40
loadUserMetaData, A-40
loadWfCompanionInfo, A-40
locale parameters, 3-8
LOGIN, 5-15

M

makeNewRevClass, A-41
mapDocNamedResultSetValuesCheckMetaChange, A-41
mapDocResultSetCheckMetaChange, A-41
mapNamedResultSetValues, A-41
mapResultSet, A-41
MARK_FOLDER_ACTIVATION_DATE, 10-12
MARK_FOLDER_CANCELLED_DATE, 10-12
MARK_FOLDER_EXPIRATION_DATE, 10-13
MARK_FOLDER_OBSOLETE_DATE, 10-13
MARK_FOLDER_RESCINDED_DATE, 10-13
MARK_FOLDER_REVIEW_DATE, 10-13
MARK_FOLDER_REVIEW_DATE_RECURSIVE, 10-14
MARK_RECORD_CANCELLED_DATE, 10-14
MARK_RECORD_EXPIRATION_DATE, 10-14
MARK_RECORD_OBSOLETE_DATE, 10-14
MARK_RECORD_RESCINDED_DATE, 10-15
MARK_RECORD_REVIEW_DATE, 10-15
markDocDeleted, A-42
markRevDeleted, A-42
MERGE_TABLE, 5-16
mergeTable, A-42
MOVE_FOLDER, 10-15
MOVE_METADEF, 5-140

N

Name column
  service ResultSet, 4-2
NOTIFY_CHANGE, 5-161
NOTIFY_CONTRIBUTOR, 6-9
NOTIFY_INDEXER, 5-81
notifyChange, A-42
notifyCollaborationUsers, A-43
notifyContributor, A-43
notifyEndPublish, A-43
notifyPublisher, A-43
notifyStartPublish, A-43

O

ODMA_DOC_INFO_SIMPLE, 5-81
organization, 1-2
overview
  actions, A-1
  archive services, 7-1
  Audience, 1-2
  Content Publisher Services, 9-1
  folder services, 8-1
  Record Manager services, 10-1
  workflow services, 6-1

P

PACKAGE_ENVIRONMENT, 5-81
packageEnvironment, A-43
packageLocalization, A-43
PAGE_HANDLER, 5-153
pageRequest, A-43
parameters
  action, 4-15
performPreview, A-44
PING_SERVER, 5-19
PNE_GET_SEARCH_RESULTS, 5-176
PNE_SAVE_QUERY, 5-154
PNE_UPDATE_PERSONAL_URLS, 5-154
PNE_UPDATE_PORTAL_INFO, 5-155
postCheckinFilter, A-44
postHtmlFormCheckin, A-44
prepareCheckinSecurity, A-44
prepareDocInfoValidate, A-44
prepareFormContinue, A-44
prepareForPreview, A-44
prepareInsertNew, A-44
prepareMailForStepUsers, A-44
prepareRedirect, A-45
prepareTopicEdits, A-46
prepareWebViewableDelivery, A-46
prepSubscription, A-46
prepSubscriptionDateUpdate, A-46
PREVIEW_DOCUMENT, 9-7
PREVIEW_RECORD_LIFECYCLE, 10-15
PROBLEMREPORT_INFO, 6-9
processCheckinArchive, A-46
processForm, A-47
PROXIED_REQUEST, 5-135
PUBLISH_SCHEMA, 5-169

Q

QactiveWorkflowProjects, A-47

Qalias, A-47
QaliasesForUser, A-47
QarchivedDoc, A-47
Qcache, A-47
QcheckinCache, A-47
QcheckinCachesForUser, A-48
Qcollaboration, A-48
QdocAccount, A-48
QdocAccounts, A-48
QdocInfo, A-49
QdocInfoSimilarCheckin, A-50
QdocName, A-50
QdocNameMeta, A-50
QdocType, A-50
QdocTypeDocuments, A-50
QdocTypes, A-51
QdocumentProblemReports, A-51
Qdocuments, A-51
QextensionFormatMap, A-51
QextensionMap, A-51
QformatMap, A-51
Qgroup, A-51
QgroupRevisions, A-51
QgroupRole, A-51
QgroupWF, A-52
QisAliasSubscribed, A-52
QisDocWFLocked, A-52
QisWfTokenUsed, A-52
QlatestIDByName, A-52
Qmetadef, A-52
QmetaFieldInfo, A-52
QODMAdocInfo, A-52
QprjDocumentsDeleted, A-53
QprjDocumentsForAgent, A-53
QproblemReport, A-53
QprojectDocument, A-53
QregisteredProject, A-53
QregisteredProjects, A-54
QreleasedDocInfo, A-54
QrevHistory, A-54
QrevHistoryReleased, A-54
QrevisionsByClass, A-54
QrevisionsByName, A-54
Qrole, A-54
Qroles, A-55
QsecurityGroup, A-55
QUERY_DOC_ACCOUNTS, 5-20
QUERY_GROUP, 5-203
QUERY_USER_ATTRIBUTES, 5-203
Quser, A-55
QuserMetaFieldInfo, A-55
QuserSecurityAttributeByType, A-55
QuserSubscription, A-55
QwfCriteriaDocRevs, A-55
QwfDocInformation, A-56
QwfDocName, A-56
QwfDocState, A-56
QwfStates, A-56
Qworkflow, A-56
QworkflowAlias, A-57
QworkflowAliases, A-57
QworkflowCriteriaForGroup, A-57
QworkflowCriteriaStepsForGroup, A-57
QworkflowDocument, A-57
QworkflowDocuments, A-58
QworkflowForID, A-58
Qworkflows, A-58
QworkflowStep, A-58
QworkflowSteps, A-58
QworkflowStepsAll, A-58

R

Record Manager services, 10-1
refreshCache, A-58
refreshLayoutLists, A-60
refreshRoles, A-60
REGISTER_IMPORTER, 7-28
REGISTER_PROJECT, 9-7
REGISTER_USER, 5-204
registerProject, A-60
rejectDoc, A-60
remoteCredentialsCheck, A-60
REMOVE, 7-30
REMOVE_COLLECTION, 7-29
REMOVE_EXPORTER, 7-29
REMOVE_IMPORTER, 7-30
REMOVE_METAFILE_SUB, 5-81
REMOVE_PROXIEDTRANSFER, 7-30
REMOVE_QUEUED_IMPORT, 7-30
REMOVE_TRANSFER, 7-31
REMOVE_WEB_APP, 5-124
removeCachedUser, A-60
removeContext, A-61
removeSubscription, A-61
renameValues, A-61
REPLACE_METAFILE_SUB, 5-82
REQUEST_SECURITYINFO, 5-162
REQUEST_TRANSFER, 7-31
requestSecurityInfo, A-62
RESEND_PROBLEMREPORT, 6-9
RESUBMIT_FOR_CONVERSION, 5-82
resubmitDocToConversion, A-62
retrieveAllProviderInfo, A-62
retrieveCachedInfo, A-62
retrieveProblemReportInfo, A-62
retrieveProviderInfo, A-62

retrieveUserDatabaseProfileData, A-63
REV_HISTORY, 5-84
REVIEW_WORKFLOW_DOC, 6-10
RMA_CLOSE_SERVICE, 10-16
RMA_CUTOFF_SERVICE, 10-16
RMA_DESTROY_SERVICE, 10-16
RMA_EXPORT_ARCHIVE_SERVICE, 10-16
RMA_EXPORT_SERVICE, 10-17
RMA_MARK_COMPLETED, 10-17
RMA_NO_ACTION_SERVICE, 10-17
RMA_OBSOLETE_SERVICE, 10-17
RMA_SCRUB_SERVICE, 10-18
RMA_SUPERSEDE_SERVICE, 10-18
runHtmlConversion, A-63

S

samples
  archive service, 7-1
  folder services, 8-1
  workflow service, 6-1
SAVE_GLOBALINCLUDES, 5-155
SAVE_TEMPLATE_CONVERSIONS, 5-124
SAVE_USER_TOPICS, 5-205
saveGlobalIncludes, A-63
saveTemplateConversions, A-63
saveUserOptionList, A-63
searchCacheReport, A-63
searchProblemReports, A-63
SELECTDOC, 5-84
sendMailTo, A-63
service attributes
  access level, 4-6
  error message, 4-11
  service class, 4-4
  service type, 4-9
  subjects notified, 4-9
  template page, 4-8
service class attribute, 4-4
service definition table, 4-2
service resource
  attributes, 4-20
service resources
  actions, 4-21
  creating, 4-29
  example, 4-2, 4-18
service ResultSet
  Actions column, 4-12
  Attributes column, 4-3
  Name column, 4-2
service type attribute, 4-9
services
  actions, 4-12
  archive, 7-3
setConditionVars, A-63
setDateToPresent, A-65
setDeleteRevReleaseState, A-65
setExternalDocInfoFields, A-65
setFileConversionInfo, A-65
setInputConversionInfo, A-65
setLocalValues, A-65
setOutputConversionInfo, A-67
setStatusMessage, A-67
setTemplateConversionInfo, A-67
setTraceFlags, A-67
SOAP_FAULT, 5-20
START_PUBLISHING, 9-7
START_SEARCH_INDEX, 5-132
startCriteriaWorkflow, A-68
startWorkflow, A-68
static page, 3-4
storeUserDatabaseProfileData, A-68
subjects notified attribute, 4-9
SUBMIT_HTML_FORM, 5-85
submitForm, A-68
submitHtmlForm, A-68
SUBSCRIBE, 5-86
SUBSCRIBE_DOC_USER, 5-88
SUBSCRIBE_EX, 5-89
SUBSCRIBE_FORM, 5-90
SUBSCRIPTION_LIST, 5-91
SuppressResultLocalization, 3-9
syntax
  service action, 4-12, A-1

T

template page attribute, 4-8
TEST_PROVIDER, 5-162
TEST_WORKFLOW_SCRIPT, 6-44
testProvider, A-68
testWorkflowScript, A-68
TRANSFER_ARCHIVE, 7-31

U

Ualias, A-69
UcheckoutRevision, A-69
Ucollaboration, A-69
UdeleteRevision, A-69
UdocFormat, A-69
UdocType, A-69
UextensionMap, A-69
Umeta, A-69
Umetadef, A-70

UNCLOSE_FOLDER, 10-18
UNDO_CHECKOUT, 5-92
UNDO_CHECKOUT_BY_NAME, 5-95
UnextCounter, A-70
UnextRevID, A-70
UNFREEZE_FOLDER, 10-18
UNFREEZE_RECORD, 10-19
UNINSTALL_COMPONENT, 5-135
uninstallComponent, A-70
UNREGISTER_PROJECT, 9-8
unregisterProject, A-70
UNSUBSCRIBE, 5-95
UNSUBSCRIBE_FORM, 5-97
UNSUBSCRIBE_FROM_LIST, 5-98
UNSUBSCRIBE_FROM_LIST_EX, 5-100
UPDATE_ADVANCED_SEARCH_OPTIONS, 5-141
UPDATE_BYREV, 5-101
UPDATE_COMPONENT_CONFIG, 5-135
UPDATE_DOCINFO, 5-102
UPDATE_DOCINFO_BYFORM, 5-105
UPDATE_DOCINFO_BYREV, 5-109, 5-109
UPDATE_DOCINFO_METAFILE_BYREV, 5-109
UPDATE_DOCINFO_STATUS, 5-109
UPDATE_DOCINFO_SUB, 5-110
UPDATE_FILTER_INFO, 5-209
UPDATE_LICENSE, 5-135
UPDATE_META_TABLE, 5-142
UPDATE_METADATA, 5-112
UPDATE_OPTION_LIST, 5-142
UPDATE_PROBLEMREPORT, 6-10
UPDATE_RESULT_TEMPLATE, 5-155
UPDATE_SUBSCRIPTION_NOTIFY, 5-110
UPDATE_SUBSCRIPTION_TYPE, 5-110
UPDATE_SUBSCRIPTION_USED, 5-111
UPDATE_TARGET_TOTALS, 7-32
UPDATE_TRANSFER_STATUS, 7-32
UPDATE_USER_META, 5-141
UPDATE_USER_META_TABLE, 5-141
UPDATE_USEROPTION_LIST, 5-209
updateAffectedWorkflows, A-70
updateCache, A-70
updateCollaborationCache, A-70
updateComponentConfig, A-71
updateConversionCache, A-71
updateDocFormatsConfig, A-71
updateDocInfo, A-71
updateFilterConfig, A-71
updateLicense, A-71
updateMetaTable, A-71
updateOptionList, A-71
updateProblemReport, A-71
updatePublishInfo, A-71
updateResultTemplate, A-72
updateRevisionIDAndLabel, A-72
updateSubscriptionType, A-72
updateUserLocale, A-72
updateUserMeta, A-72
updateUserMetaTable, A-72
updateWorkflowAndDocState, A-72
updateWorkflowState, A-72
updateWorkflowStateAfterCheckin, A-72
UPLOAD_ARCHIVE_TRANSFER, 7-32
UPLOAD_NEW_COMPONENT, 5-135
UproblemReport, A-72
Urevision, A-72
Urevision2, A-73
UrevisionStatus, A-73
UroleDefinition, A-73
UsecurityGroup, A-73
UserDateFormat, 3-8
UsubscriptionNotification, A-73
UsubscriptionUse, A-73
Uuncheckout, A-73
UuncheckoutPrevID, A-73
UuserAuthType, A-74
Uworkflow, A-74
UworkflowCriteria, A-74
UworkflowDocStep, A-74
UworkflowWithProject, A-74

V

VALIDATE_DOCINFO, 5-112
validateCheckinData, A-74
validateCollaboration, A-75
validateDelete, A-75
validateMetaData, A-75
validateProblemReport, A-75
validateStandard, A-75
validateSteps, A-76
validateUserNameAndType, A-76
VIEW_DOC, 5-177
viewDoc, A-76

W

WORK_IN_PROGRESS, 5-112
workflow services overview, 6-1
WORKFLOW_APPROVE, 6-45
WORKFLOW_CANCEL, 6-45
WORKFLOW_CHECKIN, 6-45
WORKFLOW_CHECKIN_SUB, 6-11
WORKFLOW_EDIT_APPROVE, 6-50
WORKFLOW_EDIT_REV, 6-11
WORKFLOW_NEW_REV, 6-12
WORKFLOW_REJECT, 6-50

WORKFLOW_REJECT_FORM, 6-12
WORKFLOW_START, 6-53