Oracle® Fusion Middleware

Services Reference for Oracle WebCenter Content

11*g* Release 1 (11.1.1)

E11011-08

April 2018

Documentation for administrators and developers that provides information about Oracle WebCenter Content services, which are functions and procedures performed by Content Server, Inbound Refinery, Records, and more.

ORACLE

Oracle Fusion Middleware Services Reference for Oracle WebCenter Content, 11*g* Release 1 (11.1.1)

E11011-08

Primary Author: Ajith Srinivas

Contents

Preface xxiii

Audience xxiii
Documentation Accessibility xxiii
Related Documents xxiii
Conventions xxiii

What's New in This Guide xxv

New and Changed Features for 11*g* Release 1 (11.1.1.9.0) xxv
New and Changed Features for 11*g* Release 1 (11.1.1.8.0) xxv

1 Introduction to Oracle WebCenter Content Services

1.1 List of Oracle WebCenter Content Services 1-1
1.2 Oracle WebCenter Content Terminology 1-2

2 Using Services

2.1 Overview of Services 2-1
2.1.1 Service Requests and Responses 2-1
2.1.1.1 Internal Service Requests 2-2
2.1.1.2 External Service Requests 2-2
2.1.1.3 Request Parameters 2-3
2.1.1.4 Date and Time Formatting 2-3
2.1.1.5 Case Sensitivity Considerations 2-3
2.1.2 Page Retrieval 2-4
2.1.3 Search Services 2-4
2.1.4 Integration Methods 2-4
2.1.5 Calling Services Using Persistent URLs 2-5
2.1.6 Customizing Locale Parameters 2-6
2.1.7 Forcing Authentication Challenges 2-8
2.2 Custom Application Example 2-8
2.2.1 Services Called 2-8
2.2.2 Private Functions 2-9
2.2.3 Sample Code 2-9
2.3 Redirecting Template Page for Response Output 2-14
2.3.1 Basic Concepts 2-15
2.3.2 Creating a HCST Page 2-15

2.3.3 Reformatting the Search Results Page 2-15
2.3.4 Additional Options 2-16

3 Customizing Services

3.1 Service Structure Overview 3-1
3.1.1 Name 3-2
3.1.2 Attributes 3-2
3.1.2.1 Service Class 3-3
3.1.2.2 Access Level 3-4
3.1.2.3 Template Page 3-6
3.1.2.4 Service Type 3-6
3.1.2.5 Subjects Notified 3-6
3.1.2.6 Error Message 3-8
3.1.3 Actions 3-8
3.1.3.1 Action Type 3-9
3.1.3.2 Action Name 3-10
3.1.3.3 Action Parameters 3-10
3.1.3.4 Action Control Mask 3-10
3.1.3.5 Action Error Message 3-12
3.2 Service Example 3-12
3.2.1 DOC_INFO Service Definition 3-12
3.2.2 DOC_INFO Attributes 3-13
3.2.3 DOC_INFO Actions 3-14
3.2.3.1 Action 1 Definition and Description 3-14
3.2.3.2 Action 2 Definition and Description 3-14
3.2.3.3 Action 3 Definition and Description 3-14
3.2.3.4 Action 4 Definition and Description 3-15
3.2.3.5 Action 5 Definition and Description 3-15
3.2.3.6 Action 6 Definition and Description 3-15
3.2.3.7 Action 7 Definition and Description 3-16
3.2.3.8 Action 8 Definition and Description 3-16
3.2.3.9 Action 9 Definition and Description 3-16
3.2.3.10 Action 10 Definition and Description 3-16
3.2.4 DOC_INFO Template 3-17
3.3 Creating a Service Resource 3-20
3.3.1 Creating a Custom Service Manually 3-20
3.3.2 Define the service in an HTM file 3-20
3.3.2.1 Load the service in the custom component HDA file 3-21
3.3.3 Creating a Custom Service using Component Wizard 3-22

4 Core Content Server Services

4.1 About Core Content Server Services 4-1
4.2 General Services (Core Content Server) 4-2
4.2.1 ADD_DOC_ACCOUNT 4-3
4.2.2 ADD_DOCEXTENSION 4-3
4.2.3 ADD_DOCFORMAT 4-3
4.2.4 ADD_DOCTYPE 4-4

4.2.5 APPLET_DOCINFO ... 4-4
4.2.6 CONFIG_INFO ... 4-4
4.2.7 DELETE_DOC_ACCOUNT ... 4-5
4.2.8 DELETE_DOCEXTENSION ... 4-5
4.2.9 DELETE_DOCFORMAT ... 4-5
4.2.10 DELETE_DOCTYPE ... 4-5
4.2.11 EDIT_DOCEXTENSION ... 4-6
4.2.12 EDIT_DOCFORMAT ... 4-6
4.2.13 EDIT_DOCTYPE ... 4-6
4.2.14 EDIT_TRACE_OPTIONS ... 4-7
4.2.15 GET_DATARESULTSET ... 4-7
4.2.16 GET_DOCEXTENSIONS ... 4-7
4.2.17 GET_DOCFORMATS ... 4-7
4.2.18 GET_DOCTYPES ... 4-8
4.2.19 GET_FIELD_LENGTHS ... 4-8
4.2.20 GET_FILELIST ... 4-8
4.2.21 GET_METADEFS ... 4-8
4.2.22 GET_RESULT_OPTIONS ... 4-8
4.2.23 GET_SYSTEM_AUDIT_INFO ... 4-9
4.2.24 GET_TABLE ... 4-9
4.2.25 GET_USER_METADEFS ... 4-9
4.2.26 JAVA_PROPERTIES ... 4-9
4.2.27 LM_BUILD_WEB_STRING_FILES ... 4-9
4.2.28 LM_LOAD_LAYOUTS ... 4-10
4.2.29 LM_LOAD_LAYOUTS_SUB ... 4-10
4.2.30 LOAD_DOC_ENVIRONMENT ... 4-10
4.2.31 LOGIN ... 4-10
4.2.32 MERGE_TABLE ... 4-11
4.2.33 PING_SERVER ... 4-13
4.2.34 QUERY_DOC_ACCOUNTS ... 4-14
4.2.35 SOAP_FAULT ... 4-14
4.3 Doc Services (Core Content Server) ... 4-14
4.3.1 ASSIGN_DOCINFO_FORM ... 4-16
4.3.2 CACHE_CHECKIN_NEW ... 4-17
4.3.3 CACHE_CHECKIN_SEL ... 4-17
4.3.4 CACHE_SUBMIT_HTML_FORM ... 4-17
4.3.5 CACHE_WORKFLOW_CHECKIN ... 4-17
4.3.6 CHECKIN_ARCHIVE_NO_NOTIFY ... 4-17
4.3.7 CHECKIN_BYNAME ... 4-18
4.3.8 CHECKIN_CONFIRM_FORM ... 4-21
4.3.9 CHECKIN_LIST ... 4-21
4.3.10 CHECKIN_NEW ... 4-22
4.3.11 CHECKIN_NEW_FORM ... 4-26
4.3.12 CHECKIN_NEW_SUB ... 4-27
4.3.13 CHECKIN_SEL ... 4-29
4.3.14 CHECKIN_SEL_FORM ... 4-32
4.3.15 CHECKIN_SEL_SUB ... 4-33

4.3.16 CHECKIN_SIMILAR_FORM ... 4-34
4.3.17 CHECKIN_UNIVERSAL ... 4-34
4.3.18 CHECKOUT ... 4-37
4.3.19 CHECKOUT_BY_NAME ... 4-39
4.3.20 CHECKOUT_OK ... 4-40
4.3.21 CHECKOUT_SUB ... 4-41
4.3.22 CONTINUE_CHECKIN ... 4-42
4.3.23 CONTINUE_SUBMIT_HTML_FORM ... 4-42
4.3.24 COPY_REVISION ... 4-42
4.3.25 CREATE_SUBSCRIPTION_TYPE ... 4-43
4.3.26 DELETE_BYCLASS ... 4-43
4.3.27 DELETE_BYNAME ... 4-43
4.3.28 DELETE_BYREV ... 4-43
4.3.29 DELETE_BYREV_REVISION ... 4-43
4.3.30 DELETE_CHECKIN_CACHE ... 4-44
4.3.31 DELETE_DOC ... 4-44
4.3.32 DELETE_REV ... 4-44
4.3.33 DELETE_REV_EX ... 4-45
4.3.34 DELETE_SUBSCRIPTION_TYPE ... 4-46
4.3.35 DOC_FORMATS_WIZARD ... 4-46
4.3.36 DOC_INFO ... 4-46
4.3.37 DOC_INFO_BY_NAME ... 4-50
4.3.38 DOC_INFO_LATESTRELEASE ... 4-54
4.3.39 DOC_INFO_SIMPLE ... 4-54
4.3.40 DOC_INFO_SIMPLE_BYREV ... 4-55
4.3.41 DOC_SUBS_LIST ... 4-55
4.3.42 EDIT_DOC_FORMATS ... 4-57
4.3.43 FORM_PROCESS ... 4-58
4.3.44 FORM_SUBMIT ... 4-58
4.3.45 GET_CACHED_CHECKIN_INFO ... 4-58
4.3.46 GET_DOC_CONFIG_INFO ... 4-58
4.3.47 GET_DOC_SUBSCRIBERS ... 4-59
4.3.48 GET_DOCUMENT_HISTORY_REPORT ... 4-59
4.3.49 GET_ENVIRONMENT ... 4-59
4.3.50 GET_EXPIRED ... 4-59
4.3.51 GET_PACKAGE_ENVIRONMENT_PAGE ... 4-61
4.3.52 GET_UPDATE_FORM ... 4-61
4.3.53 NOTIFY_INDEXER ... 4-61
4.3.54 ODMA_DOC_INFO_SIMPLE ... 4-62
4.3.55 PACKAGE_ENVIRONMENT ... 4-62
4.3.56 REMOVE_METAFILE_SUB ... 4-62
4.3.57 REPLACE_METAFILE_SUB ... 4-62
4.3.58 RESTORE_REVISION ... 4-62
4.3.59 RESUBMIT_FOR_CONVERSION ... 4-62
4.3.60 REV_HISTORY ... 4-64
4.3.61 SELECTDOC ... 4-64
4.3.62 SUBMIT_HTML_FORM ... 4-65

4.3.63 SUBSCRIBE ... 4-66
4.3.64 SUBSCRIBE_DOC_USER ... 4-67
4.3.65 SUBSCRIBE_EX ... 4-68
4.3.66 SUBSCRIBE_FORM ... 4-68
4.3.67 SUBSCRIPTION_LIST ... 4-69
4.3.68 UNDO_CHECKOUT ... 4-70
4.3.69 UNDO_CHECKOUT_BY_NAME ... 4-72
4.3.70 UNSUBSCRIBE ... 4-72
4.3.71 UNSUBSCRIBE_FORM ... 4-74
4.3.72 UNSUBSCRIBE_FROM_LIST ... 4-75
4.3.73 UNSUBSCRIBE_FROM_LIST_EX ... 4-76
4.3.74 UPDATE_BYREV ... 4-77
4.3.75 UPDATE_DOCINFO ... 4-77
4.3.76 UPDATE_DOCINFO_BYFORM ... 4-80
4.3.77 UPDATE_DOCINFO_BYREV ... 4-83
4.3.78 UPDATE_DOCINFO_METAFILE_BYREV ... 4-83
4.3.79 UPDATE_DOCINFO_STATUS ... 4-83
4.3.80 UPDATE_DOCINFO_SUB ... 4-83
4.3.81 UPDATE_SUBSCRIPTION_NOTIFY ... 4-83
4.3.82 UPDATE_SUBSCRIPTION_TYPE ... 4-83
4.3.83 UPDATE_SUBSCRIPTION_USED ... 4-84
4.3.84 UPDATE_METADATA ... 4-84
4.3.85 VALIDATE_DOCINFO ... 4-85
4.3.86 WORK_IN_PROGRESS ... 4-85
4.4 Doc Profile Services (Core Content Server) ... 4-85
4.4.1 ADD_DOCPROFILE ... 4-86
4.4.2 ADD_DOCRULE ... 4-86
4.4.3 DELETE_DOCPROFILE ... 4-86
4.4.4 DELETE_DOCRULE ... 4-87
4.4.5 DOCPROFILE_PREVIEW ... 4-87
4.4.6 EDIT_DOCPROFILE ... 4-87
4.4.7 EDIT_DOCPROFILE_TRIGGER ... 4-87
4.4.8 EDIT_DOCRULE ... 4-88
4.4.9 GET_DOCPROFILE ... 4-88
4.4.10 GET_DOCPROFILES ... 4-88
4.4.11 GET_DOCRULE ... 4-88
4.4.12 GET_DOCRULES ... 4-88
4.5 File Services (Core Content Server) ... 4-88
4.5.1 ADD_WEB_APP ... 4-89
4.5.2 APPEND_FILE_CACHING_INFO ... 4-89
4.5.3 GET_DYNAMIC_CONVERSION ... 4-89
4.5.4 GET_DYNAMIC_CONVERSION_SUB ... 4-91
4.5.5 GET_DYNAMIC_URL ... 4-91
4.5.6 GET_TEMPLATE_CONVERSIONS ... 4-91
4.5.7 GET_WEB_APP_STATUS ... 4-92
4.5.8 LOAD_RESOURCE_FILE ... 4-92
4.5.9 REMOVE_WEB_APP ... 4-93

4.5.10 SAVE_TEMPLATE_CONVERSIONS ... 4-93
4.6 Indexer Services (Core Content Server) ... 4-93
4.6.1 CANCEL_SEARCH_INDEX ... 4-93
4.6.2 CONTROL_SEARCH_INDEX ... 4-93
4.6.3 GET_FILE ... 4-97
4.6.4 START_SEARCH_INDEX ... 4-99
4.7 Internal Services (Core Content Server) ... 4-100
4.7.1 CANCEL_COMPONENT_INSTALL ... 4-100
4.7.2 CLEAR_SERVER_OUTPUT ... 4-100
4.7.3 DOWNLOAD_COMPONENT ... 4-100
4.7.4 GET_COMPONENT_CONFIG ... 4-100
4.7.5 GET_COMPONENT_INSTALL_FORM ... 4-100
4.7.6 GET_COMPONENT_INSTALL_PROMPTS_FORM ... 4-100
4.7.7 GET_COMPONENT_INSTALL_SETTINGS' ... 4-100
4.7.8 GET_LOCAL_REGISTRATION_FORM ... 4-101
4.7.9 GET_MANIFEST_INFO ... 4-101
4.7.10 GET_SERVER_OUTPUT ... 4-101
4.7.11 PROXIED_REQUEST ... 4-101
4.7.12 UNINSTALL_COMPONENT ... 4-101
4.7.13 UPDATE_COMPONENT_CONFIG ... 4-101
4.7.14 UPDATE_LICENSE ... 4-101
4.7.15 UPLOAD_NEW_COMPONENT ... 4-101
4.8 Meta Services (Core Content Server) ... 4-101
4.8.1 ADD_METADEF ... 4-102
4.8.2 DEL_METADEF ... 4-103
4.8.3 EDIT_METADEF ... 4-103
4.8.4 GET_ADVANCED_SEARCH_OPTIONS ... 4-104
4.8.5 GET_DISPLAY_FIELDS ... 4-104
4.8.6 GET_DOC_METADATA_INFO ... 4-107
4.8.7 GET_OPTION_LIST ... 4-108
4.8.8 GET_USER_METADATA_INFO ... 4-108
4.8.9 MOVE_METADEF ... 4-108
4.8.10 UPDATE_ADVANCED_SEARCH_OPTIONS ... 4-108
4.8.11 UPDATE_USER_META ... 4-108
4.8.12 UPDATE_USER_META_TABLE ... 4-108
4.8.13 UPDATE_META_TABLE ... 4-109
4.8.14 UPDATE_OPTION_LIST ... 4-109
4.9 Miscellaneous Services (Core Content Server) ... 4-110
4.9.1 CHUNKED_UPLOAD ... 4-110
4.9.2 DOWNLOAD_LISTBOX_ITEMS ... 4-110
4.9.3 LOAD_USER_LOCALIZATION ... 4-111
4.10 Page Handler/Page Request Services (Core Content Server) ... 4-111
4.10.1 DELETE_RESULT_TEMPLATE ... 4-111
4.10.2 GET_ADMIN_PAGE ... 4-111
4.10.3 GET_DOC_PAGE ... 4-112
4.10.4 GET_DYNAMIC_PAGE ... 4-114
4.10.5 GET_PERSONALIZED_JAVASCRIPT ... 4-115

4.10.6 GET_PORTAL_PAGE 4-115
4.10.7 GET_SECURE_PAGE 4-116
4.10.8 LOAD_GLOBALINCLUDES 4-117
4.10.9 PAGE_HANDLER 4-117
4.10.10 PNE_SAVE_QUERY 4-117
4.10.11 PNE_UPDATE_PERSONAL_URLS 4-118
4.10.12 PNE_UPDATE_PORTAL_INFO 4-118
4.10.13 SAVE_GLOBALINCLUDES 4-118
4.10.14 UPDATE_RESULT_TEMPLATE 4-119
4.11 Provider Manager Services (Core Content Server) 4-120
4.11.1 ADD_EDIT_PROVIDER 4-120
4.11.2 APPEND_DATABASE_AUDIT_INFO 4-121
4.11.3 DELETE_PROVIDER 4-121
4.11.4 ENABLE_DISABLE_PROVIDER 4-121
4.11.5 GET_ADD_EDIT_PROVIDER_FORM 4-122
4.11.6 GET_ALL_PROVIDERS 4-122
4.11.7 GET_PROVIDER_INFO 4-122
4.11.8 NOTIFY_CHANGE 4-123
4.11.9 REQUEST_SECURITYINFO 4-123
4.11.10 TEST_PROVIDER 4-123
4.12 Schema Services (Core Content Server) 4-123
4.12.1 ADD_SCHEMA_FIELD 4-124
4.12.2 ADD_SCHEMA_RELATION 4-124
4.12.3 ADD_SCHEMA_VIEW 4-124
4.12.4 ADDOREDIT_SCHEMA_TABLE 4-125
4.12.5 CONTROL_SCHEMA 4-125
4.12.6 DELETE_SCHEMA_FIELD 4-125
4.12.7 DELETE_SCHEMA_RELATION 4-125
4.12.8 DELETE_SCHEMA_TABLE 4-125
4.12.9 DELETE_SCHEMA_VIEW 4-125
4.12.10 EDIT_SCHEMA_FIELD 4-125
4.12.11 EDIT_SCHEMA_NODE 4-126
4.12.12 EDIT_SCHEMA_RELATION 4-126
4.12.13 EDIT_SCHEMA_VIEW 4-126
4.12.14 EDIT_SCHEMA_VIEW_VALUES 4-126
4.12.15 GET_SCHEMA_FIELD_INFO 4-126
4.12.16 GET_SCHEMA_FIELDS 4-127
4.12.17 GET_SCHEMA_RELATIONS 4-127
4.12.18 GET_SCHEMA_STATS 4-127
4.12.19 GET_SCHEMA_TABLE_INFO 4-127
4.12.20 GET_SCHEMA_TABLES 4-127
4.12.21 GET_SCHEMA_VIEW_EDIT_INFO 4-127
4.12.22 GET_SCHEMA_VIEW_FRAGMENT 4-127
4.12.23 GET_SCHEMA_VIEW_INFO 4-128
4.12.24 GET_SCHEMA_VIEW_VALUES 4-128
4.12.25 GET_SCHEMA_VIEWS 4-128
4.12.26 PUBLISH_SCHEMA 4-128

4.13 Search Services (Core Content Server) ... 4-128
4.13.1 APPEND_SEARCH_AUDIT_INFO ... 4-128
4.13.2 GET_EXTERNAL_DOC_INFO ... 4-129
4.13.3 GET_EXTERNAL_HIGHLIGHT_INFO ... 4-129
4.13.4 GET_EXTERNAL_XML_HIGHLIGHT_INFO ... 4-129
4.13.5 GET_HIGHLIGHT_INFO ... 4-130
4.13.6 GET_SEARCH_RESULTS ... 4-131
4.13.7 GET_SEARCH_RESULTS_FORCELOGIN ... 4-132
4.13.8 GET_XML_HIGHLIGHT_INFO ... 4-132
4.13.9 PNE_GET_SEARCH_RESULTS ... 4-133
4.13.10 VIEW_DOC ... 4-133
4.14 User Services (Core Content Server) ... 4-134
4.14.1 ADD_ALIAS ... 4-135
4.14.2 ADD_GROUP ... 4-136
4.14.3 ADD_ROLE ... 4-137
4.14.4 ADD_USER ... 4-137
4.14.5 CHANGE_USER_AUTH_TYPE ... 4-139
4.14.6 CHECK_USER_CREDENTIALS ... 4-139
4.14.7 DELETE_ALIAS ... 4-142
4.14.8 DELETE_GROUP ... 4-142
4.14.9 DELETE_ROLE ... 4-143
4.14.10 DELETE_USER ... 4-143
4.14.11 EDIT_ALIAS ... 4-143
4.14.12 EDIT_GROUP ... 4-145
4.14.13 EDIT_ROLE ... 4-145
4.14.14 EDIT_USER ... 4-146
4.14.15 EDIT_USER_PROFILE ... 4-148
4.14.16 GET_ALIASES ... 4-150
4.14.17 GET_FILTER_ADMIN_PAGE ... 4-150
4.14.18 GET_SELF_REGISTER_PAGE ... 4-150
4.14.19 GET_USERDOCPROFILES ... 4-150
4.14.20 GET_USER_INFO ... 4-150
4.14.21 GET_USERS ... 4-152
4.14.22 LOAD_PNE_PORTAL ... 4-152
4.14.23 LOAD_USER_TOPIC ... 4-152
4.14.24 QUERY_GROUP ... 4-153
4.14.25 QUERY_USER_ATTRIBUTES ... 4-153
4.14.26 REGISTER_USER ... 4-153
4.14.27 SAVE_USER_TOPICS ... 4-154
4.14.28 UPDATE_FILTER_INFO ... 4-157
4.14.29 UPDATE_USEROPTION_LIST ... 4-157
4.15 Collaboration Services (Core Content Server) ... 4-157
4.15.1 ADD_COLLABORATION ... 4-158
4.15.2 ADD_COLLABORATION_FORM ... 4-158
4.15.3 DELETE_COLLABORATION ... 4-158
4.15.4 EDIT_CLBRA_ACCESS_LIST ... 4-158
4.15.5 EDIT_CLBRA_ACCESS_LIST_FORM ... 4-158

4.15.6 EDIT_COLLABORATION ... 4-159
4.15.7 EDIT_COLLABORATION_FORM ... 4-159
4.15.8 GET_CLBRA_DOCUMENTS ... 4-159
4.15.9 GET_CLBRA_INFO ... 4-159
4.15.10 GET_COLLABORATION_LIST ... 4-159
4.15.11 GET_USER_CLBRA_LIST ... 4-159

5 Workflow Services

5.1 About Workflow Services ... 5-1
5.2 Doc and General Services (Workflows) ... 5-1
5.2.1 ADD_PROBLEMREPORT ... 5-2
5.2.2 DELETE_PROBLEMREPORT ... 5-3
5.2.3 GET_CRITERIA_WORKFLOWS_FOR_GROUP ... 5-3
5.2.4 GET_DOCUMENT_PROBLEMREPORTS ... 5-4
5.2.5 GET_PROBLEMREPORTS_SEARCH_FORM ... 5-4
5.2.6 GET_PROBLEMREPORTS_SEARCH_RESULTS ... 5-4
5.2.7 GET_UPDATE_PROBLEMREPORT_FORM ... 5-4
5.2.8 GET_WORKFLOWDOCUMENTS ... 5-5
5.2.9 GET_WORKFLOWS_FOR_ALL ... 5-5
5.2.10 LOAD_WORKFLOW_QUEUE ... 5-5
5.2.11 NOTIFY_CONTRIBUTOR ... 5-6
5.2.12 PROBLEMREPORT_INFO ... 5-6
5.2.13 RESEND_PROBLEMREPORT ... 5-6
5.2.14 REVIEW_WORKFLOW_DOC ... 5-6
5.2.15 UPDATE_PROBLEMREPORT ... 5-7
5.2.16 WORKFLOW_CHECKIN_SUB ... 5-7
5.2.17 WORKFLOW_EDIT_REV ... 5-7
5.2.18 WORKFLOW_NEW_REV ... 5-8
5.2.19 WORKFLOW_REJECT_FORM ... 5-8
5.3 Workflow Template Services ... 5-8
5.3.1 ADD_WF_TEMPLATE ... 5-8
5.3.2 DELETE_WF_TEMPLATE ... 5-9
5.3.3 EDIT_WF_TEMPLATE ... 5-9
5.3.4 GET_WF_TEMPLATE ... 5-13
5.3.5 GET_WF_TEMPLATES ... 5-13
5.4 Workflow Services ... 5-13
5.4.1 ADD_WORKFLOW ... 5-15
5.4.2 ADD_WORKFLOW_SCRIPT ... 5-15
5.4.3 ADD_WORKFLOW_TOKEN ... 5-15
5.4.4 ADD_WORKFLOWALIASES ... 5-16
5.4.5 ADD_WORKFLOWDOCUMENT ... 5-16
5.4.6 ADD_WORKFLOWDOCUMENT_SUB ... 5-17
5.4.7 ADD_WORKFLOWDOCUMENTS ... 5-17
5.4.8 ADD_WORKFLOWSTEP ... 5-17
5.4.9 CRITERIAWORKFLOW_DISABLE ... 5-18
5.4.10 CRITERIAWORKFLOW_DISABLE_SUB ... 5-18
5.4.11 CRITERIAWORKFLOW_ENABLE ... 5-19

5.4.12 DELETE_WFCONTRIBUTORS ... 5-19
5.4.13 DELETE_WORKFLOW ... 5-19
5.4.14 DELETE_WORKFLOW_SCRIPT ... 5-19
5.4.15 DELETE_WORKFLOW_TOKEN ... 5-20
5.4.16 DELETE_WORKFLOWCRITERIA ... 5-20
5.4.17 DELETE_WORKFLOWDOCUMENTS ... 5-20
5.4.18 DELETE_WORKFLOWSTEP ... 5-21
5.4.19 EDIT_WORKFLOW ... 5-22
5.4.20 EDIT_WORKFLOW_SCRIPT ... 5-22
5.4.21 EDIT_WORKFLOW_TOKEN ... 5-22
5.4.22 EDIT_WORKFLOWCRITERIA ... 5-23
5.4.23 EDIT_WORKFLOWSTEP ... 5-23
5.4.24 GET_ACTIVE_WORKFLOWS ... 5-25
5.4.25 GET_ALL_WORKFLOWDOCREVISIONS ... 5-25
5.4.26 GET_CRITERIA_WORKFLOWS_FOR_GROUP ... 5-25
5.4.27 GET_WF_COMPANION_INFO ... 5-26
5.4.28 GET_WORKFLOW ... 5-26
5.4.29 GET_WORKFLOW_INFO ... 5-26
5.4.30 GET_WORKFLOW_INFO_BYNAME ... 5-27
5.4.31 GET_WORKFLOW_SCRIPT ... 5-29
5.4.32 GET_WORKFLOWDOCREVISIONS ... 5-29
5.4.33 GET_WORKFLOWS ... 5-30
5.4.34 TEST_WORKFLOW_SCRIPT ... 5-30
5.4.35 WORKFLOW_APPROVE ... 5-30
5.4.36 WORKFLOW_CANCEL ... 5-31
5.4.37 WORKFLOW_CHECKIN ... 5-31
5.4.38 WORKFLOW_EDIT_APPROVE ... 5-34
5.4.39 WORKFLOW_REJECT ... 5-35
5.4.40 WORKFLOW_START ... 5-37

6 Archive Services

6.1 About Archive Services ... 6-1
6.2 Archive Services ... 6-2
6.2.1 ADD_ARCHIVE ... 6-3
6.2.2 ADD_COLLECTION ... 6-3
6.2.3 ADD_PROXIEDCOLLECTION ... 6-4
6.2.4 CANCEL_ARCHIVE ... 6-4
6.2.5 CHECKIN_ARCHIVE ... 6-4
6.2.6 COPY_ARCHIVE ... 6-8
6.2.7 DELETE_ARCHIVE ... 6-8
6.2.8 DELETE_BATCH_FILE ... 6-8
6.2.9 DELETE_BATCH_FILE_DOCUMENTS ... 6-9
6.2.10 DELETE_BATCH_FILE_TABLES ... 6-9
6.2.11 EDIT_ARCHIVE ... 6-9
6.2.12 EDIT_ARCHIVEDATA ... 6-10
6.2.13 EDIT_EXPORTERS ... 6-10
6.2.14 EDIT_TRANSFEROPTIONS ... 6-10

6.2.15 EXECUTE_BATCH ... 6-11
6.2.16 EXPORT_ARCHIVE ... 6-11
6.2.17 GET_ARCHIVECOLLECTIONS ... 6-12
6.2.18 GET_ARCHIVETABLECONTENT ... 6-12
6.2.19 GET_ARCHIVED_FILE ... 6-12
6.2.20 GET_ARCHIVES ... 6-13
6.2.21 GET_ARCHIVERELATIONQUERY ... 6-14
6.2.22 GET_BATCH_FILE_DOCUMENTS ... 6-14
6.2.23 GET_BATCH_PROPERTIES ... 6-14
6.2.24 GET_BATCH_SCHEMA ... 6-14
6.2.25 GET_BATCH_VALUES ... 6-15
6.2.26 GET_BATCHFILES ... 6-15
6.2.27 GET_PROXIED_ARCHIVECOLLECTIONS ... 6-15
6.2.28 GET_PROXIEDSERVERS ... 6-16
6.2.29 GET_REPLICATION_DATA ... 6-16
6.2.30 GET_TABLECOLUMNLIST ... 6-16
6.2.31 GET_TARGET_INFO ... 6-16
6.2.32 GET_TARGET_TRANSFER_STATUS ... 6-17
6.2.33 GET_TRANSFER_SOURCE_INFO ... 6-17
6.2.34 IMPORT_ARCHIVE ... 6-17
6.2.35 IMPORT_ARCHIVE_START_AUTOMATED ... 6-18
6.2.36 IMPORT_BATCHFILE ... 6-19
6.2.37 IMPORT_DOCUMENT ... 6-19
6.2.38 IMPORT_TABLE_ENTRY ... 6-19
6.2.39 INSERT_NATIVE ... 6-19
6.2.40 INSERT_NEW ... 6-20
6.2.41 REGISTER_IMPORTER ... 6-20
6.2.42 REMOVE_COLLECTION ... 6-20
6.2.43 REMOVE_EXPORTER ... 6-21
6.2.44 REMOVE_IMPORTER ... 6-21
6.2.45 REMOVE_PROXIEDTRANSFER ... 6-21
6.2.46 REMOVE_QUEUED_IMPORT ... 6-21
6.2.47 REMOVE_TRANSFER ... 6-22
6.2.48 REQUEST_TRANSFER ... 6-22
6.2.49 TRANSFER_ARCHIVE ... 6-22
6.2.50 UPDATE_TARGET_TOTALS ... 6-22
6.2.51 UPDATE_TRANSFER_STATUS ... 6-22
6.2.52 UPLOAD_ARCHIVE_TRANSFER ... 6-23

7 Contribution Folders Services

7.1 About Contribution Folders Services ... 7-1
7.2 Contribution Folders Services ... 7-1
7.2.1 COLLECTION_ADD ... 7-3
7.2.2 COLLECTION_ADD_LINK ... 7-5
7.2.3 COLLECTION_BROWSE ... 7-6
7.2.4 COLLECTION_CHECKIN_NEW ... 7-6
7.2.5 COLLECTION_CHECKIN_REVISION ... 7-6

7.2.6 COLLECTION_CHECKIN_SEL_SUB 7-7
7.2.7 COLLECTION_COPY_ALL 7-7
7.2.8 COLLECTION_COPY_COLLECTION 7-7
7.2.9 COLLECTION_COPY_ITEM 7-8
7.2.10 COLLECTION_COPY_LOT 7-9
7.2.11 COLLECTION_DELETE 7-10
7.2.12 COLLECTION_DELETE_ALL 7-11
7.2.13 COLLECTION_DELETE_COLLECTION 7-11
7.2.14 COLLECTION_DELETE_ITEM 7-11
7.2.15 COLLECTION_DELETE_LOT 7-11
7.2.16 COLLECTION_DISPLAY 7-13
7.2.17 COLLECTION_EDIT 7-14
7.2.18 COLLECTION_GET_ADMIN_CONFIG 7-14
7.2.19 COLLECTION_GET_ADMIN_INHERIT_CONFIG 7-14
7.2.20 COLLECTION_GET_ADMIN_MARKED_CONFIG 7-14
7.2.21 COLLECTION_GET_ADMIN_META_CONFIG 7-15
7.2.22 COLLECTION_GET_ADMIN_METADATA_DEFAULTS 7-15
7.2.23 COLLECTION_GET_ADMIN_MOUNTED_CONFIG 7-15
7.2.24 COLLECTION_GET_ARCHIVE 7-15
7.2.25 COLLECTION_GET_BRANCH 7-15
7.2.26 COLLECTION_GET_COLLECTION 7-15
7.2.27 COLLECTION_GET_COLLECTIONS 7-16
7.2.28 COLLECTION_GET_CONTENT_FILE 7-16
7.2.29 COLLECTION_GET_CONTENTS 7-16
7.2.30 COLLECTION_GET_FILE 7-17
7.2.31 COLLECTION_GET_INFO 7-17
7.2.32 COLLECTION_GET_LINKS 7-17
7.2.33 COLLECTION_GET_META_MAPPING 7-18
7.2.34 COLLECTION_GET_PROFILE_METADATA_DEFAULTS 7-18
7.2.35 COLLECTION_GET_PROFILE_METADATA_REVISION_DEFAULTS 7-18
7.2.36 COLLECTION_GET_REFERENCE 7-18
7.2.37 COLLECTION_GET_SEARCH_FORM 7-19
7.2.38 COLLECTION_GET_SYSTEM_FILE 7-19
7.2.39 COLLECTION_GET_USER_CONFIG 7-19
7.2.40 COLLECTION_INFO 7-20
7.2.41 COLLECTION_ISVALID_META 7-20
7.2.42 COLLECTION_LOCK 7-21
7.2.43 COLLECTION_MOVE_ALL 7-21
7.2.44 COLLECTION_MOVE_COLLECTION 7-22
7.2.45 COLLECTION_MOVE_ITEM 7-23
7.2.46 COLLECTION_MOVE_LOT 7-24
7.2.47 COLLECTION_NEW 7-25
7.2.48 COLLECTION_PROFILE_GET_COLUMNS 7-26
7.2.49 COLLECTION_PROFILE_UPDATE_COLUMNS 7-26
7.2.50 COLLECTION_RESTORE_COLLECTION 7-26
7.2.51 COLLECTION_RESTORE_ITEM 7-26
7.2.52 COLLECTION_SEARCH_CONTENT 7-27

7.2.53 COLLECTION_SEARCH_RESULTS 7-27
7.2.54 COLLECTION_SET_ARCHIVE 7-28
7.2.55 COLLECTION_SET_USER_CONFIG 7-28
7.2.56 COLLECTION_UNLOCK 7-28
7.2.57 COLLECTION_UPDATE 7-28
7.2.58 COLLECTION_UPDATE_ADMIN_CONFIG 7-29
7.2.59 COLLECTION_UPDATE_ADMIN_INHERIT_CONFIG 7-30
7.2.60 COLLECTION_UPDATE_ADMIN_METADATA_DEFAULTS 7-30
7.2.61 COLLECTION_UPDATE_ALL 7-30
7.2.62 COLLECTION_UPDATE_ITEM 7-30
7.2.63 COLLECTION_UPDATE_META 7-30
7.2.64 COLLECTION_UPDATE_META_TABLE 7-31
7.2.65 COLLECTION_UPDATE_PROFILE_METADATA_DEFAULTS 7-31
7.2.66 COLLECTION_UPDATE_PROFILE_METADATA_REVISION_DEFAULTS 7-31
7.2.67 COLLECTION_UPDATE_STRUCTURE 7-31
7.2.68 COLLECTION_VERIFY_FOLDER_NAME 7-31
7.2.69 FOLDERSLOCAL_BUILD_MOUNT 7-33
7.2.70 FOLDERSLOCAL_CREATE_MOUNT 7-33
7.2.71 FOLDERSLOCAL_DELETE_MOUNT 7-33
7.2.72 FOLDERSLOCAL_UPDATE_MOUNT 7-33
7.2.73 GET_FOLDER_HISTORY_REPORT 7-33
7.2.74 GET_OPTION_LISTS 7-34
7.2.75 GOTO_COLLECTION 7-34
7.2.76 GOTO_ROOT_COLLECTION 7-34

8 Folders Services

8.1 About Folders Services 8-1
8.2 Folders Services 8-1
8.2.1 FLD_BROWSE 8-2
8.2.2 FLD_BROWSE_POPUP 8-5
8.2.3 FLD_COPY 8-5
8.2.4 FLD_CREATE_FILE 8-6
8.2.5 FLD_CREATE_FILE_FORM 8-8
8.2.6 FLD_CREATE_FOLDER 8-8
8.2.7 FLD_CREATE_FOLDER_FORM 8-10
8.2.8 FLD_DELETE 8-11
8.2.9 FLD_EDIT_FILE 8-11
8.2.10 FLD_EDIT_FILE_FORM 8-12
8.2.11 FLD_EDIT_FOLDER 8-12
8.2.12 FLD_EDIT_FOLDER_FORM 8-12
8.2.13 FLD_EDIT_METADATA_RULES 8-13
8.2.14 FLD_EDIT_METADATA_RULES_FORM 8-13
8.2.15 FLD_FOLDER_MIGRATION_STATUS 8-14
8.2.16 FLD_FOLDER_SEARCH 8-14
8.2.17 FLD_FOLDER_SEARCH_FORM 8-14
8.2.18 FLD_GET_CHOOSE_DESTINATION_DIALOG 8-14
8.2.19 FLD_GET_CREATE_LINK_DIALOG 8-15

8.2.20 FLD_GET_CREATE_SHORTCUT_DIALOG ... 8-15
8.2.21 FLD_GET_RENAME_FILE_DIALOG ... 8-16
8.2.22 FLD_GET_RENAME_FOLDER_DIALOG ... 8-16
8.2.23 FLD_INFO ... 8-16
8.2.24 FLD_LOAD_SOFT_LINKS_FOR_DOCUMENT ... 8-17
8.2.25 FLD_MIGRATION_FOLDER_DATA ... 8-17
8.2.26 FLD_MOVE ... 8-17
8.2.27 FLD_PRE_CHECKIN ... 8-18
8.2.28 FLD_PROPAGATE ... 8-19
8.2.29 FLD_PROPAGATE_FORM ... 8-19
8.2.30 FLD_REINDEX_FOLDER_CONTENTS ... 8-20
8.2.31 FLD_RETRIEVE_CHILD_FILES ... 8-20
8.2.32 FLD_RETRIEVE_CHILD_FOLDERS ... 8-21
8.2.33 FLD_SUBSCRIBE ... 8-22
8.2.34 FLD_TEST_USER_CAPABILITIES ... 8-22
8.2.35 FLD_UNSUBSCRIBE ... 8-26
8.2.36 FLD_UNFILE ... 8-26

9 Electronic Signature Services

9.1 About Electronic Signature Services ... 9-1
9.2 Electronic Signature Services ... 9-1
9.2.1 ESIG_GET_ADMIN_PAGE ... 9-2
9.2.2 ESIG_GET_CONTENT_SIGNATURES ... 9-2
9.2.3 ESIG_GET_CONTENT_SIGNATURES_AND_INFO ... 9-2
9.2.4 ESIG_GET_SIGNATURE_DETAILS ... 9-3
9.2.5 ESIG_GET_SIGN_CONTENT_FORM ... 9-3
9.2.6 ESIG_GET_SIGN_WORKFLOW_FORM ... 9-4
9.2.7 ESIG_GET_USER_DEFINED_FIELDS ... 9-4
9.2.8 ESIG_GET_VALIDATE_FILE_FORM ... 9-5
9.2.9 ESIG_SET_CONFIG ... 9-5
9.2.10 ESIG_SIGN_CONTENT ... 9-6
9.2.11 ESIG_VALIDATE_FILE ... 9-6
9.2.12 ESIG_VALIDATE_FILE_GLOBAL ... 9-7
9.2.13 ESIG_WORKFLOW_SIGN_AND_APPROVE ... 9-7

10 Records Services

10.1 About Records Services ... 10-1
10.2 Records Services ... 10-1
10.2.1 ACTIVATE_SERVICE ... 10-3
10.2.2 APPROVE_DELETE_SERVICE ... 10-3
10.2.3 BROWSE_CATEGORY_FORM ... 10-3
10.2.4 BROWSE_FOLDER_FORM ... 10-3
10.2.5 BROWSE_SERIES_FORM ... 10-4
10.2.6 CHECKIN_NEW_REVISION_SERVICE ... 10-4
10.2.7 CHECKIN_SIMILAR_FORM ... 10-4
10.2.8 CLEAR_FOLDER_CANCELLED_DATE ... 10-4
10.2.9 CLEAR_FOLDER_EXPIRATION_DATE ... 10-4

10.2.10 CLEAR_FOLDER_OSBOLETE_DATE 10-4
10.2.11 CLEAR_FOLDER_RESCINDED_DATE 10-4
10.2.12 CLEAR_FOLDER_REVIEW_DATE 10-5
10.2.13 CLEAR_RECORD_CANCELLED_DATE 10-5
10.2.14 CLEAR_RECORD_EXPIRATION_DATE 10-5
10.2.15 CLEAR_RECORD_OBSOLETE_DATE 10-5
10.2.16 CLOSE_FOLDER 10-5
10.2.17 CREATE_FOLDER 10-5
10.2.18 CREATE_FOLDER_FORM 10-6
10.2.19 DELETE_ALL_BUT_LAST_N_REVISIONS_ SERVICE 10-6
10.2.20 DELETE_FOLDER 10-6
10.2.21 DELETE_REVISION_SERVICE 10-6
10.2.22 EDIT_FOLDER 10-6
10.2.23 EDIT_FOLDER_FORM 10-6
10.2.24 FREEZE_FOLDER 10-6
10.2.25 FREEZE_RECORD 10-7
10.2.26 INFO_CATEGORY_FORM 10-7
10.2.27 INFO_FOLDER_FORM 10-7
10.2.28 INFO_FOLDER_LIFECYCLE 10-7
10.2.29 INFO_FOLDER_METADATA_HISTORY 10-7
10.2.30 INFO_FOLDER_REVIEW_HISTORY 10-7
10.2.31 INFO_RECORD_LIFECYCLE 10-8
10.2.32 INFO_RECORD_METADATA_HISTORY 10-8
10.2.33 INFO_RECORD_REVIEW_HISTORY 10-8
10.2.34 INFO_SERIES_FORM 10-8
10.2.35 MARK_FOLDER_ACTIVATION_DATE 10-8
10.2.36 MARK_FOLDER_CANCELLED_DATE 10-8
10.2.37 MARK_FOLDER_EXPIRATION_DATE 10-9
10.2.38 MARK_FOLDER_OBSOLETE_DATE 10-9
10.2.39 MARK_FOLDER_RESCINDED_DATE 10-9
10.2.40 MARK_FOLDER_REVIEW_DATE 10-9
10.2.41 MARK_FOLDER_REVIEW_DATE_ RECURSIVE 10-9
10.2.42 MARK_RECORD_CANCELLED_DATE 10-9
10.2.43 MARK_RECORD_EXPIRATION_DATE 10-10
10.2.44 MARK_RECORD_OBSOLETE_DATE 10-10
10.2.45 MARK_RECORD_RESCINDED_DATE 10-10
10.2.46 MARK_RECORD_REVIEW_DATE 10-10
10.2.47 MOVE_FOLDER 10-10
10.2.48 PREVIEW_RECORD_LIFECYCLE 10-10
10.2.49 RMA_CLOSE_SERVICE 10-11
10.2.50 RMA_CUTOFF_SERVICE 10-11
10.2.51 RMA_DESTROY_SERVICE 10-11
10.2.52 RMA_EXPORT_ARCHIVE_SERVICE 10-11
10.2.53 RMA_EXPORT_SERVICE 10-11
10.2.54 RMA_MARK_COMPLETED 10-11
10.2.55 RMA_NO_ACTION_SERVICE 10-11
10.2.56 RMA_OBSOLETE_SERVICE 10-12

10.2.57 RMA_SCRUB_SERVICE 10-12
10.2.58 RMA_SUPERSEDE_SERVICE 10-12
10.2.59 UNCLOSE_FOLDER 10-12
10.2.60 UNFREEZE_FOLDER 10-12
10.2.61 UNFREEZE_RECORD 10-12

11 Physical Content Management Services

11.1 About Physical Content Management Services 11-1
11.2 Physical Content Management Services 11-1
11.2.1 ADD_FUNCTION_BARCODE 11-4
11.2.2 ADJUST_INVOICE 11-5
11.2.3 ADJUST_INVOICE_FORM 11-5
11.2.4 BARCODE_PROCESS 11-5
11.2.5 BATCH_STORAGE_CREATION_FORM 11-5
11.2.6 BLOCK_STORAGE 11-5
11.2.7 BROWSE_CHARGE_INVOICES_FORM 11-6
11.2.8 BROWSE_CHARGE_TRANSACTIONS_BY_INVOICE_FORM 11-6
11.2.9 BROWSE_CHARGE_TRANSACTIONS_FORM 11-6
11.2.10 BROWSE_CHARGE_TRANSACTIONS_WITH_NO_INVOICE_FORM 11-6
11.2.11 BROWSE_RESERVATION_ITEMS_FORM 11-6
11.2.12 BROWSE_STORAGE_FORM 11-6
11.2.13 CHECKIN_EXTERNAL_ITEM 11-7
11.2.14 CHECKOUT_EXTERNAL_ITEM 11-7
11.2.15 CLEAN_UP_STORAGE 11-7
11.2.16 CONFIG_CHARGE_BACKS 11-7
11.2.17 CONFIG_CHARGE_BACKS_FORM 11-7
11.2.18 CONFIGURE_CHARGE_BILLERS_FORM 11-7
11.2.19 CONFIGURE_CHARGE_PAYMENT_TYPES_FORM 11-7
11.2.20 CONFIGURE_CHARGE_TYPES_FORM 11-8
11.2.21 CREATE_CHARGE_BILLER 11-8
11.2.22 CREATE_CHARGE_BILLER_FORM 11-8
11.2.23 CREATE_CHARGE_PAYMENT_TYPE 11-8
11.2.24 CREATE_CHARGE_PAYMENT_TYPE_FORM 11-8
11.2.25 CREATE_CHARGE_TYPE 11-8
11.2.26 CREATE_CHARGE_TYPE_FORM 11-9
11.2.27 CREATE_CHARGE_TRANSACTION 11-9
11.2.28 CREATE_CHARGE_TRANSACTION_FORM 11-9
11.2.29 CREATE_CHARGE_TRANSACTION_MULTIPLE 11-9
11.2.30 CREATE_EXTERNAL_ITEM 11-10
11.2.31 CREATE_EXTERNAL_ITEM_FORM 11-10
11.2.32 CREATE_EXTERNAL_ITEM_SIMILAR_FORM 11-10
11.2.33 CREATE_MEDIA_TYPE 11-10
11.2.34 CREATE_MEDIA_TYPE_FORM 11-11
11.2.35 CREATE_OBJECT_TYPE 11-11
11.2.36 CREATE_OBJECT_TYPE_FORM 11-11
11.2.37 CREATE_RESERVATION 11-11
11.2.38 CREATE_RESERVATION_FORM 11-11

11.2.39 CREATE_STORAGE 11-11
11.2.40 CREATE_STORAGE_FORM 11-12
11.2.41 CREATE_STORAGE_TYPE 11-12
11.2.42 CREATE_STORAGE_TYPE_FORM 11-12
11.2.43 DELETE_CHARGE_BILLER 11-12
11.2.44 DELETE_CHARGE_INVOICE 11-12
11.2.45 DELETE_CHARGE_PAYMENT_TYPE 11-13
11.2.46 DELETE_CHARGE_TRANSACTION 11-13
11.2.47 DELETE_CHARGE_TYPE 11-13
11.2.48 DELETE_COMPLETED_RESERVATIONS 11-13
11.2.49 DELETE_EXTERNAL_ITEM 11-13
11.2.50 DELETE_MEDIA_TYPE 11-13
11.2.51 DELETE_OBJECT_TYPE 11-14
11.2.52 DELETE_RESERVATION 11-14
11.2.53 DELETE_RESERVATION_ITEM 11-14
11.2.54 DELETE_STORAGE 11-14
11.2.55 DELETE_STORAGE_TYPE 11-14
11.2.56 EDIT_CHARGE_BILLER 11-15
11.2.57 EDIT_CHARGE_BILLER_FORM 11-15
11.2.58 EDIT_CHARGE_PAYMENT_TYPE 11-15
11.2.59 EDIT_CHARGE_TYPE_FORM 11-15
11.2.60 EDIT_EXTERNAL_ITEM 11-15
11.2.61 EDIT_EXTERNAL_ITEM_FORM 11-15
11.2.62 EDIT_MEDIA_TYPE 11-16
11.2.63 EDIT_MEDIA_TYPE_FORM 11-16
11.2.64 EDIT_OBJECT_TYPE 11-16
11.2.65 EDIT_OBJECT_TYPE_FORM 11-16
11.2.66 EDIT_OBJECT_TYPE_RELATIONSHIPS 11-16
11.2.67 EDIT_OBJECT_TYPE_RELATIONSHIPS_FORM 11-17
11.2.68 EDIT_RESERVATION 11-17
11.2.69 EDIT_RESERVATION_FORM 11-17
11.2.70 EDIT_RESERVATION_ITEM 11-17
11.2.71 EDIT_RESERVATION_ITEM_FORM 11-18
11.2.72 EDIT_RESERVATION_ITEM_STATUS 11-18
11.2.73 EDIT_STORAGE 11-18
11.2.74 EDIT_STORAGE_FORM 11-18
11.2.75 EDIT_STORAGE_TYPE 11-18
11.2.76 EDIT_STORAGE_TYPE_FORM 11-19
11.2.77 GENERATE_CHARGE_INVOICE 11-19
11.2.78 GET_EXPORT_INVOICES 11-19
11.2.79 GET_EXTERNAL_ITEM_SEARCH_RESULTS 11-19
11.2.80 GET_RELATED_CONTENT 11-19
11.2.81 INFO_CHARGE_BILLER_FORM 11-20
11.2.82 INFO_CHARGE_INVOICE_FORM 11-20
11.2.83 INFO_CHARGE_PAYMENT_TYPE_FORM 11-20
11.2.84 INFO_CHARGE_TRANSACTION_FORM 11-20
11.2.85 INFO_CHARGE_TYPE_FORM 11-21

11.2.86 INFO_EXTERNAL_ITEM_FORM 11-21
11.2.87 INFO_MEDIA_TYPE_FORM 11-21
11.2.88 INFO_OBJECT_TYPE_FORM 11-21
11.2.89 INFO_RESERVATION_FORM 11-21
11.2.90 INFO_RESERVATION_ITEM_FORM 11-22
11.2.91 INFO_STORAGE_FORM 11-22
11.2.92 INFO_STORAGE_TYPE_FORM 11-22
11.2.93 MARK_INVOICE_PAID 11-22
11.2.94 MARK_INVOICE_PAID_FORM 11-22
11.2.95 MOVE_STORAGE 11-23
11.2.96 NOTIFY_OVERDUE_USERS 11-23
11.2.97 PROCESS_OVERDUE_REQUESTS 11-23
11.2.98 PROCESS_STORAGE_SPACE_COUNTS 11-23
11.2.99 RESERVE_STORAGE 11-23
11.2.100 RUN_CHARGEBACK_REPORT 11-23
11.2.101 RUN_CHARGEBACK_REPORT_MULTIPLE 11-24
11.2.102 RUN_FUNCTION_BARCODE_LABEL 11-24
11.2.103 RUN_RESERVATION_REPORT 11-24
11.2.104 RUN_RESERVATION_SEARCH_RESULTS_REPORT 11-24
11.2.105 RUN_STORAGE_BARCODE_LABEL 11-24
11.2.106 RUN_STORAGE_REPORT 11-24
11.2.107 RUN_USER_BARCODE_LABEL 11-25
11.2.108 SCREEN_CHARGE_BACKS_ADVANCED_FORM 11-25
11.2.109 SCREEN_CHARGE_BACKS_FORM 11-25
11.2.110 SEARCH_EXTERNAL_ITEM_FORM 11-25
11.2.111 UNRESERVE_STORAGE 11-25
11.2.112 UPDATE_STORAGE_DEPTHS 11-25
11.2.113 UPDATE_USERS_WITH_NO_BARCODES 11-26

12 Extended User Attributes Services

12.1 About Extended User Attributes Services 12-1
12.2 Extended User Attributes Services 12-1
12.2.1 ADD_EXTENDED_USER_ATTRIBUTES 12-2
12.2.2 EDIT_EXTENDED_USER_ATTRIBUTES 12-3
12.2.3 DELETE_EXTENDED_USER_ATTRIBUTES 12-3
12.2.4 DELETE_EXTENDED_ATTRIBUTES_BY_APPLICATION 12-4
12.2.5 DELETE_ALL_EXTENDED_ATTRIBUTES_FOR_USER 12-4
12.2.6 DELETE_ALL_EXTENDED_ATTRIBUTES_FOR_APPLICATION 12-5
12.2.7 QUERY_EXTENDED_USER_ATTRIBUTES 12-5
12.2.8 QUERY_EXTENDED_ATTRIBUTE_MAPPINGS 12-6
12.2.9 EC_SET_PROPERTY 12-6
12.2.10 EC_DELETE_PROPERTY 12-7
12.2.11 EC_GET_PROPERTY 12-7
12.2.12 EC_GET_PROPERTY_BY_KEY 12-8
12.2.13 SET_DEFAULT_ATTRIBUTES 12-8
12.2.14 DELETE_DEFAULT_ATTRIBUTES 12-9
12.2.15 GET_DEFAULT_ATTRIBUTES 12-9

12.2.16 SET_EXTENDED_ATTRIBUTE_MAPPINGS 12-10
12.2.17 DELETED_EXTENDED_ATTRIBUTE_MAPPINGS 12-10
12.2.18 ADD_USER 12-11
12.2.19 EDIT_USER 12-12
12.2.20 DELETE_USER 12-13
12.2.21 QUERY_USER_ATTRIBUTES 12-13

13 Folios Services

13.1 About Folios Services 13-1
13.2 Folio Services 13-1
13.2.1 LOAD_FOLIO_NODE 13-2
13.2.2 UPDATE_FOLIO 13-2
13.2.3 CHECKIN_NEW_FOLIO 13-6
13.2.4 CREATE_FOLIO_SNAPSHOT 13-6
13.2.5 LOCK_FOLIO 13-6
13.2.6 UNLOCK_FOLIO 13-7
13.2.7 CREATE_FOLIO_RENDITION 13-7
13.2.8 GENERATE_GUIDS 13-7

14 Link Manager Services

14.1 About Link Manager Services 14-1
14.2 Link Manager Services 14-1
14.2.1 ABORT_LINKS_ACTIVITY 14-2
14.2.2 ADD_MANAGED_DOCLINKS 14-2
14.2.3 DELETE_MANAGED_DOCLINKS 14-2
14.2.4 GET_LINK_INFO 14-2
14.2.5 GET_LINKS_ADMIN_PAGE 14-3
14.2.6 LK_GET_SEARCH_RESULTS 14-4
14.2.7 RECOMPUTE_MANAGED_LINKS 14-4
14.2.8 REFRESH_MANAGED_DOCLINKS 14-4
14.2.9 REFRESH_MANAGED_LINKS 14-4
14.2.10 REFRESH_REFS_MANAGED_LINKS 14-5

15 Virtual Content Repository Services

15.1 About Virtual Content Repository Services 15-1
15.2 Virtual Content Repository Services 15-1
15.2.1 VCR_FOLDER_INFO 15-2
15.2.2 VCR_GET_CONTENT_TYPE 15-4
15.2.3 VCR_GET_CONTENT_TYPES 15-4
15.2.4 VCR_GET_DOCUMENT 15-4
15.2.5 VCR_GET_DOCUMENT_BY_NAME 15-5

16 Security Services

16.1 About Security Services 16-1
16.2 Security Services 16-1

16.2.1 ASC_GET_SECURITY_CONFIGURATIONS 16-1
16.2.2 ASC_UPDATE_SECURITY_CONFIGURATIONS 16-2

A Actions

A.1 About Service Actions A-1
A.2 A A-2
A.3 B A-4
A.4 C A-4
A.5 D A-13
A.6 E A-20
A.7 F A-22
A.8 G A-22
A.9 H A-26
A.10 I A-26
A.11 L A-28
A.12 M A-31
A.13 N A-33
A.14 P A-33
A.15 Q A-36
A.16 R A-44
A.17 S A-47
A.18 T A-51
A.19 U A-51
A.20 V A-56

Preface

This guide provides detailed information about Oracle WebCenter Content services, which are functions or procedures performed by the Oracle WebCenter Content Server and predefined components.

Audience

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

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle WebCenter Content documentation set:

- *Oracle Fusion Middleware Administering Oracle WebCenter Content*
- *Oracle Fusion Middleware Managing Oracle WebCenter Content*
- *Oracle Fusion Middleware Developing with Oracle WebCenter Content*
- *Oracle Fusion Middleware Configuration Reference for Oracle WebCenter Content*

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

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in This Guide

This chapter introduces the new and changed services for Oracle WebCenter Content and other significant changes that are described in this guide, and provides pointers to additional information.

New and Changed Features for 11*g* Release 1 (11.1.1.9.0)

The following are new and changed services for Oracle WebCenter Content 11*g* Release 1 (11.1.1).

- GET_DYNAMIC_CONVERSION implements the UseConversionCache optional service parameter instead of the useDocInfoCache parameter.

New and Changed Features for 11*g* Release 1 (11.1.1.8.0)

The following are new and changed services for Oracle WebCenter Content 11*g* Release 1 (11.1.1) :

- COPY_REVISION, a new service that copies one document revision, creating a new rev class from it.
- FLD_BROWSE includes additional optional parameters and returned data information to support the user interface.
- FLD_SUBSCRIBE, a new service that allows a user to subscribe to a folder.
- FLD_TEST_USER_CAPABILITIES, a new service that tests the capability of a user to perform an operation on a document, folder, or library.
- FLD_UNSUBSCRIBE, a new service that allows a user to unsubscribe a folder.
- GET_DISPLAY_FIELDS, a new service that returns information about custom metadata fields, including Dynamic Tree, for different Content Server pages.
- GET_FIELD_LENGTHS, a new service that returns maximum length information for length constrained fields.
- GET_USER_INFO includes the new `UserLocaleLanguageMap` ResultSet, which returns information on available locales for the user.
- GET_USERDOCPROFILES, a new service that returns information about the checkin and search profiles available to the current user.
- RESTORE_REVISION, a new service that restores the document revision corresponding to the supplied dID as a new (latest) revision.

- CONTROL_SEARCH_INDEX adds the `fastRebuild` parameter which can enable the search engine to add new information to the search collection without requiring a full collection rebuild (only applies to OracleTextSearch).

1

Introduction to Oracle WebCenter Content Services

This chapter provides an introduction to Oracle WebCenter Content services, which are functions or procedures performed by Oracle WebCenter Content Server, Oracle WebCenter Content: Inbound Refinery, Oracle WebCenter Content: Records, and other Oracle WebCenter Content features.

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

The information contained in this guide is based on 11*g* Release 1 (11.1.1) . 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 cannot warrant compatibility with all versions and features of third-party products.

> **Note:** For more information about customizing and integrating the Oracle WebCenter Content system, see *Oracle Fusion Middleware Developing with Oracle WebCenter Content*.

This chapter covers the following topics:

- List of Oracle WebCenter Content Services
- Oracle WebCenter Content Terminology

1.1 List of Oracle WebCenter Content Services

This guide is arranged according to Oracle WebCenter Content functionality or component, making it easier for you to find information about specific services. Services that are used extensively have an extended description. In addition, frequently-used services are marked with an asterisk (*) in introductory lists in each chapter.

- Chapter 4, "Core Content Server Services," describes core Content Server services, grouped by common usage.
- Chapter 5, "Workflow Services," describes workflow services used with Content Server.

- Chapter 6, "Archive Services," describes archiving services used with Content Server.
- Chapter 7, "Contribution Folders Services," describes services used with contribution folders (supported by the Folders_g component) in Content Server.
- Chapter 8, "Folders Services," describes services used with folders (supported by the FrameworkFolders component) in Content Server.
- Chapter 10, "Records Services," describes services used with Oracle WebCenter Content: Records.
- Chapter 11, "Physical Content Management Services," describes services used with Physical Content Manager.
- Chapter 12, "Extended User Attributes Services," describes services used with Extended User Attributes in Content Server.
- Chapter 13, "Folios Services," describes services used with Folios in Content Server.
- Chapter 14, "Link Manager Services," describes services used with Link Manager.
- Chapter 15, "Virtual Content Repository Services," describes services used by applications that work with the Virtual Content Repository (VCR) service provider interfaces.
- Appendix A, "Actions," describes the actions used by individual services.

1.2 Oracle WebCenter Content Terminology

- Oracle WebCenter Content documentation uses the following terms when referring to variables in the directories associated with the Oracle WebCenter Content and Content Server configuration:
  - *IdcHomeDir*: This variable refers to the `ucm/idc` directory in the Oracle WebCenter Content home where the Oracle WebCenter Content server media is located. The server media can run Oracle WebCenter Content Server, Oracle WebCenter Content: Inbound Refinery, or Oracle WebCenter Content: Records software. This is essentially a read-only directory. The default location is *WCC_ORACLE_HOME*`/ucm/idc`. The variable portion of the default location can be changed, but the path cannot be changed from `ucm/idc`.
  - *DomainHome*: This variable refers to the user-specified directory where an Oracle WebCenter Content application is deployed to run on an Oracle WebLogic Server application server. The *DomainHome*`/ucm/`*short-product-id*`/bin` directory contains the `intradoc.cfg` file and executables. The default location for *DomainHome* is *MW_HOME*`/user_projects/domains/`*base_domain*, but you can change the path and domain name (*base_domain*) during the deployment of an Oracle WebCenter Content application to an application server.
  - *short-product-id*: This variable refers to the type of Oracle WebCenter Content server deployed to an application server. This name is used as the context root (default `HttpRelativeWebRoot` configuration value). Possible values include:
    - `cs` (Oracle WebCenter Content Server)
    - `ibr` (Oracle WebCenter Content: Inbound Refinery)
    - `urm` (Oracle WebCenter Content: Records)

– *IntradocDir*: This variable refers to the root directory for configuration and data files specific to an Oracle WebCenter Content instance that is part of an Oracle WebCenter Content application deployed to an application server. This Idoc Script variable is configured for one type of Oracle WebCenter Content instance: Content Server (`cs`), Inbound Refinery (`ibr`), or Records (`urm`). This directory can be located elsewhere, but the default location is *`DomainHome`*`/ucm/`*`short-product-id`*. The specified directory must be an absolute path to the instance directory and must be unique to a particular server or node. The directory includes a `bin/` directory, which contains the startup files (`intradoc.cfg` and executables).

2

Using Services

This chapter discusses how to use Oracle WebCenter Content services and provides an example of a custom service.

This chapters covers the following topics:

- Overview of Services
- Custom Application Example
- Redirecting Template Page for Response Output

2.1 Overview of Services

A service is a function or procedure that is performed by the Oracle WebCenter Content system. Calling an Oracle WebCenter Content service (making a service request) is the only way a client can communicate with the Oracle WebCenter Content system or access the Oracle WebCenter Content Server database.

This section covers the following topics:

- Section 2.1.1, "Service Requests and Responses"
- Section 2.1.2, "Page Retrieval"
- Section 2.1.3, "Search Services"
- Section 2.1.4, "Integration Methods"
- Section 2.1.5, "Calling Services Using Persistent URLs"
- Section 2.1.6, "Customizing Locale Parameters"

2.1.1 Service Requests and Responses

Any service can be called either externally (from outside the Oracle WebCenter Content system) or internally (within the Oracle WebCenter Content system 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:

- Section 2.1.1.1, "Internal Service Requests"
- Section 2.1.1.2, "External Service Requests"
- Section 2.1.1.3, "Request Parameters"
- Section 2.1.1.4, "Date and Time Formatting"

- Section 2.1.1.5, "Case Sensitivity Considerations"

2.1.1.1 Internal Service Requests

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

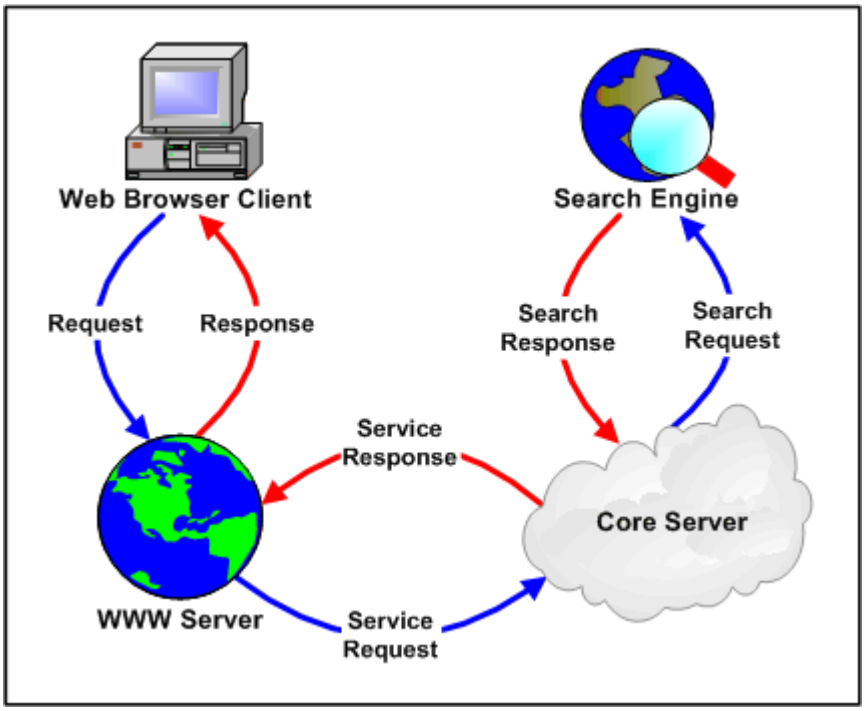
2.1.1.2 External Service Requests

Any external program or HTML page can call any Oracle WebCenter Content service to request information from the Oracle WebCenter Content system 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 client (for example, a web browser) 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 Oracle WebCenter Content system, along with any required and optional parameters. The Oracle WebCenter Content system 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 Oracle WebCenter Content system then returns the results to the Web server, and the Web server returns the results to the web browser client.

Figure 2–1 External Requests and Responses Between Content Management System Elements



2.1.1.3 Request Parameters

A service request must include every parameter 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.example.com/cs/idcplg?
IdcService=DOC_INFO&dID=194
```

2.1.1.4 Date and Time Formatting

Default date and time formatting for Content Server are determined using the Localization tab on the System Properties utility interface. 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` represents the meridian symbol (for example, AM or PM).
- The `m` represents minutes.
- 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. For example, !mXXX,YYY could designate the meridian symbols (where XXX is the symbol meaning "before noon" and YYY is the symbol meaning "after noon"). Everything after the exclamation point passes the meridian and time zone symbols along with the date format.

2.1.1.5 Case Sensitivity Considerations

Case sensitivity is important when calling standard Oracle WebCenter Content 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 Oracle WebCenter Content 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 Oracle WebCenter Content might affect the case sensitivity of parameters and parameter values. For example, some databases do not have case-sensitive column names and other databases are case-insensitive by default. So a database may have stored "DDOCNAME", but the value that the Oracle WebCenter Content system expects is still "dDocName". Also, some databases store values in uppercase, but they perform case-insensitive queries. For example, if you store the dDocName "BouncyCaps", a query for "bouncycaps" would find that row.

2.1.2 Page Retrieval

When a web page is requested from the Oracle WebCenter Content system, 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 Oracle WebCenter Content web site, the only static page is the guest home page (for example, *DomainHome*`/ucm/cs/weblayout/portal.htm`).
- **dynamic page:** A dynamic web page is assembled at the time of the Web server request, using Oracle WebCenter Content services and templates to determine the content and formatting. For example, each user's portal design page is generated using an Oracle WebCenter Content service called GET_PORTAL_PAGE and a template called PNE_PORTAL_DESIGN_PAGE.

2.1.3 Search Services

A search request is a special kind of Oracle WebCenter Content service. When the Oracle WebCenter Content system 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 Oracle WebCenter Content system.

2.1.4 Integration Methods

Service requests can be made by any external program or HTML page using a wide variety of protocols. The Oracle WebCenter Content system 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 Oracle WebCenter Content system by persistent URL. For more information, see Section 2.1.5, "Calling Services Using Persistent URLs."

The following are other possible integration methods:

- Oracle WebLogic Server Web Services (WS) using integrated WebLogic JAXP and SAML support
- Oracle WebCenter Content Web services component for the Content Server and Remote Intradoc Client (RIDC) component for clients
- Java API (IdcCommand) integration using the IdcCommand Java Command Utility
- Java Server Page (JSP) integration from a JSP running in Oracle WebCenter Content, a JSP through the Oracle WebCenter Content JavaBean, or a JSP through the Oracle WebCenter Content Enterprise JavaBean (EJB) deployed on your J2EE application server
- Java 2 Enterprise Edition API (J2EE) integration by deploying the Oracle WebCenter Content Enterprise JavaBean on your J2EE-compliant application server
- Simple Object Access Protocol (SOAP) integration using the SOAP protocol
- Virtual Folders integration using the Folders_g component
- Web Distributed Authoring and Versioning (WebDAV) integration using the WebDAV component
- Component Object Model (COM) integration using the ActiveX utility or the IntradocClient OCX component

Note: For more detailed information on available integration methods, see *Oracle WebCenter Content Developer's Guide for Content Server*.

2.1.5 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 example shows a typical URL; in this case, it is the URL for the Home page:

```
http://cs.example.com/cs/idcplg?
IdcService=GET_DOC_PAGE&Action=GetTemplatePage&Page=HOME_PAGE
```

- **http://cs.example.com/** is the web address of the Content Server instance.
- **cs/idcplg** 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. This parameter is specific to the GET_DOC_PAGE service example.
- **Page=HOME_PAGE** tells the Content Server which template page to use. This parameter is specific to the GET_DOC_PAGE service example.
- The question mark (**?**) indicates the end of the Web server path and the beginning of Content Server parameters.
- Ampersands (**&**) are used as separators between Content Server parameters.
- 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.

Troubleshooting Examples

- IsJava=1
- IsPageDebug=1

Customization Examples

- StdPageWidth=1000
- dDocType=HRForm

Example

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

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=oracle&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. Content Server passes the request to the search engine.
4. The search engine returns the search results to Content Server.
5. Based on the user login and security permissions, 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.

2.1.6 Customizing Locale Parameters

When using the Content Server in a client server operation mode, you can use several parameters 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 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.

  ```
  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 format. If that is unavailable, it uses the Content Server locale date format. 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.

  ```
  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 a formatted string, usually an error message, that has not been localized. You can use this parameter 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. If the field types are not specified in `biFieldTypes`, then fields are treated like plain strings and no localization is performed on them.

  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.

  ```
  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.

  ```
  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 is necessary 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.

  ```
  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 instance to be processed or when the data is to be persistently stored and replayed back to the original Content Server instance as needed.

```
SuppressResultLocalization=1
```

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

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.

2.1.7 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. For more details on this variable, see *Oracle Fusion Middleware Configuration Reference for Oracle WebCenter Content*.

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.

2.2 Custom Application Example

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

- Services Called
- Private Functions
- Sample Code

2.2.1 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

2.2.2 Private Functions

These private functions are defined:

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

2.2.3 Sample Code

This sample is in Visual Basic.

```
' 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:\oracle\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:\oracle\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:\oracle\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:\oracle\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:\oracle\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:\oracle\bin")
str = "@Properties LocalData" + vbCrLf + "IdcService=ADD_USER" + vbCrLf +
"dName=Jennifer" + vbCrLf + "dFullName=Jennifer Smith" + vbCrLf +
"dPassword=password" + vbCrLf + "dEmail=email@example.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:\oracle\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/cs/idcplg?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
```

2.3 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 through the component architecture, and specifying a different template page. Another more flexible way to do this as needed is to specify the urlTemplate parameter, to redirect the response page to a HCSP or HCST, and reformat the results in any way you want.

This section covers the following topics:

- Section 2.3.1, "Basic Concepts"

- Section 2.3.2, "Creating a HCST Page"
- Section 2.3.3, "Reformatting the Search Results Page"
- Section 2.3.4, "Additional Options"

2.3.1 Basic Concepts

You should be somewhat familiar with Idoc Script and with Dynamic Server Pages (HCST, HCSP, HCSF) before attempting this exercise. You should also be somewhat familiar with Content Server feature and component architecture. It would also be helpful to be familiar with HTML FORM objects. For more information see *Oracle WebCenter Content Developer's Guide for Content Server* and *Oracle Fusion Middleware Configuration Reference for Oracle WebCenter Content*.

2.3.2 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 instance. 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/oracle/groups/public/documents/document/test_result.hcst`

2.3.3 Reformatting the Search Results Page

To test our new template, we will start by going to a search page. Enter in any search criteria 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=/oracle/groups/public/documents/adacct/test_result.hcst
```

You should now see the same search results formatted in a minimalist 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='/oracle/groups/public/documents/document/test_result.hcst'>
```

This can be done by creating a component that modifies the include `query_results_options` to contain the sample 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.

2.3.4 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_result.hcst
  ```

  Because RedirectURL doesn't work with all service calls, and pre-6.0 versions of Content Server software have minor data pollution bugs with docTemplateName and docTemplateID, it is 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*). This parameter behaves 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.

3 Customizing Services

This chapter discusses the basic structure of Oracle WebCenter Content services to aid in customizing services.

This chapter covers the following topics:

- Service Structure Overview
- Service Example
- Creating a Service Resource

3.1 Service Structure Overview

This section describes how standard services are implemented in Oracle WebCenter Content Server. For information about calling services from other programs, see the chapter about integration methods in *Oracle Fusion Middleware Developing with Oracle WebCenter Content*.

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

- Section 3.1.1, "Name"
- Section 3.1.2, "Attributes"
- Section 3.1.3, "Actions"

The standard Content Server services are defined in the StandardServices table in the *IdcHomeDir*`/resources/core/tables/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 ResultSet.

The table row displayed in Figure 3–1 is an example of a service definition.

Figure 3–1 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@>

3.1.1 Name

The Name column in the StandardServices table defines the name for each service. For client-side service requests, this is the name called in Section 2.1.5, "Calling Services Using Persistent URLs". For standard web requests, this is almost always the URL to the Content Server web page.

Figure 3–2 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@>

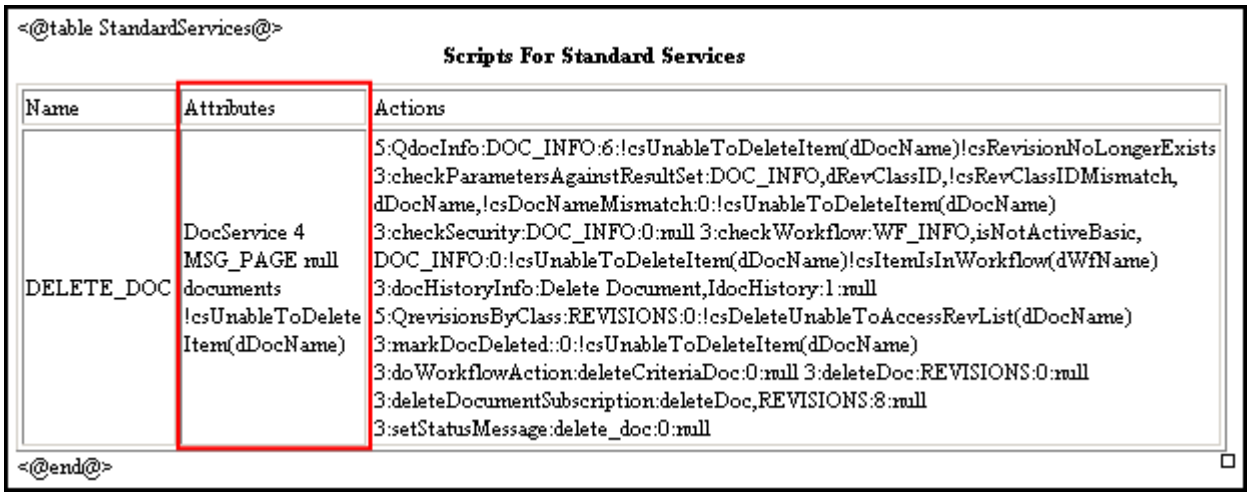
3.1.2 Attributes

The Attributes column in the StandardServices table defines aspects of each service, as discussed in the following sections:

- Section 3.1.2.1, "Service Class"
- Section 3.1.2.2, "Access Level"
- Section 3.1.2.3, "Template Page"
- Section 3.1.2.4, "Service Type"
- Section 3.1.2.5, "Subjects Notified"

- Section 3.1.2.6, "Error Message"

Figure 3–3 The Attributes Column of the DELETE_DOC Service Definition



3.1.2.1 Service Class

The *service class* attribute specifies the Java class object that the service can access. 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.
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.

Service Class	Description
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.
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.

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

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

3.1.2.2 Access Level

The service security model is similar to the document security model used throughout the 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. Generally only one privilege out of READ, WRITE, DELETE, or ADMIN is assigned to a service. 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.
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. The user must have that level of permission in at least one security group 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 through 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

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
```

For details about user accounts and roles permissions see *Oracle Fusion Middleware Administering Oracle WebCenter Content*.

3.1.2.3 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 Idoc Script. The Idoc Script 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 *IdcHomeDir*`/components/Folders/resources/templates/templates.hda` file:

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

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)
```

3.1.2.4 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`.

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)
```

3.1.2.5 Subjects Notified

The s*ubjects 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 instance 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-delimited 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
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
pagelist	Library (Web Layout Editor) pages
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)
usermetaoptlists	User information field option lists
users	User information
wfscripts	Workflow event scripts
wftemplates	Workflow templates
workflows	Workflows

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

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

3.1.2.6 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 *Oracle Fusion Middleware Developing with Oracle WebCenter Content*.

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

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

3.1.3 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:

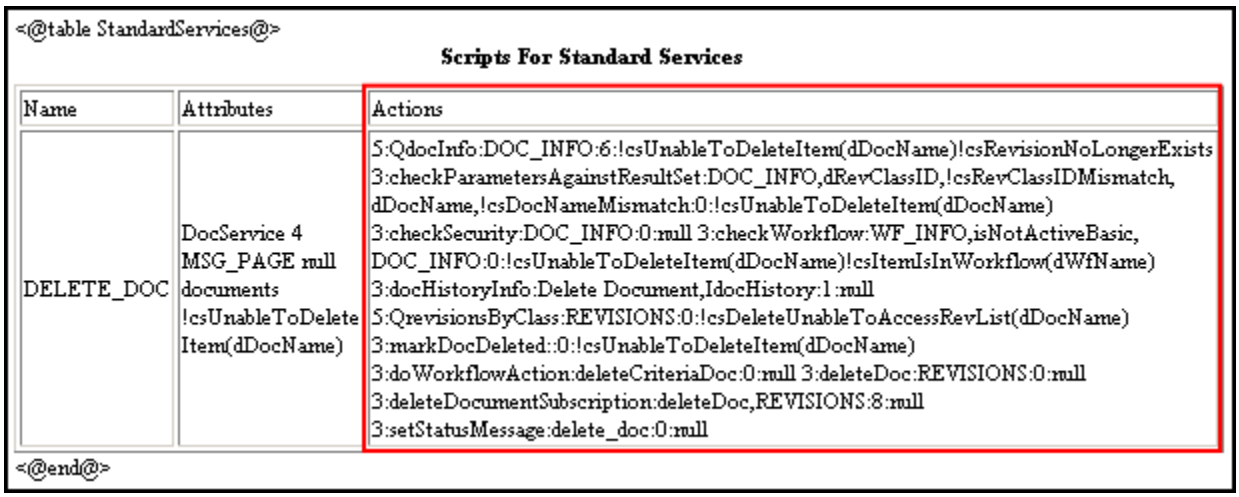
```
type:name:parameters:control mask:error message
```

See the following sections for more information:

- Section 3.1.3.1, "Action Type"
- Section 3.1.3.2, "Action Name"
- Section 3.1.3.3, "Action Parameters"
- Section 3.1.3.4, "Action Control Mask"
- Section 3.1.3.5, "Action Error Message"

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 3–4 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@>

3.1.3.1 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 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 of the action.
2	Execute query	EXECUTE_TYPE	Executes a predefined SQL database query to delete, add, or update information in the database.
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 of the action.

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
```

3.1.3.2 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, 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.
I	Insert (add) information in the database.
U	Update information in the database.

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
```

3.1.3.3 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-delimited 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.

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
```

3.1.3.4 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-delimited 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-delimited list of bit flag string values (custom services).
- **Java Constant**: Used to identify the control mask in the 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 terminate 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.
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 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 Section 3.1.3.1, "Action Type."

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-delimited list of the bit flag string values:

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

3.1.3.5 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 *Oracle Fusion Middleware Developing with Oracle WebCenter Content*.

- 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.

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
```

3.2 Service Example

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

- Section 3.2.1, "DOC_INFO Service Definition"
- Section 3.2.2, "DOC_INFO Attributes"
- Section 3.2.3, "DOC_INFO Actions"
- Section 3.2.4, "DOC_INFO Template"

3.2.1 DOC_INFO Service Definition

This section shows details of the DOC_INFO service definition from the *IdcHomeDir*/resources/core/tables/std_services.htm file.

The following is an example of the DOC_INFO service definition, displayed in a text editor:

```
<@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:DOCINFO,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@>
```

Figure 3–5 Example of the DOC_INFO Service Definition

<@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@>

3.2.2 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.

Attribute	Value	Description
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`

3.2.3 DOC_INFO Actions

The DOC_INFO service executes the following ten actions.

3.2.3.1 Action 1 Definition and Description

Action 1 takes the following form:

- **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`

3.2.3.2 Action 2 Definition and Description

Action 2 takes the following form:

- **Action 2**: 3:mapNamedResultSetValues:DOCINFO,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.

3.2.3.3 Action 3 Definition and Description

Action 3 takes the following form:

- **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}.'
```

3.2.3.4 Action 4 Definition and Description

Action 4 takes the following form:

- **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-delimited 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.

3.2.3.5 Action 5 Definition and Description

Action 5 takes the following form:

- **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.

3.2.3.6 Action 6 Definition and Description

Action 6 takes the following form:

- **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.

3.2.3.7 Action 7 Definition and Description

Action 7 takes the following form:

- **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.

3.2.3.8 Action 8 Definition and Description

Action 8 takes the following form:

- **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.

3.2.3.9 Action 9 Definition and Description

Action 9 takes the following form:

- **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.

3.2.3.10 Action 10 Definition and Description

Action 10 takes the following form:

- **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:

  ```
  Ubpnable to retrieve revision history for ''{dDocName}.''
  ```

3.2.4 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 *IdcHomeDir*/components/Folders/resources/std_page.htm file. Code from both files appear in the following markup section:

Markup from the *IdcHomeDir*/resources/core/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 *IdcHomeDir*/components/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@>
```

3.3 Creating a Service Resource

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

- Section 3.3.1, "Creating a Custom Service Manually"
- Section 3.3.3, "Creating a Custom Service using Component Wizard"

Note: For more information about custom components, see *Oracle Fusion Middleware Developing with Oracle WebCenter Content*.

3.3.1 Creating a Custom Service Manually

To create a custom service resource manually:

- Section 3.3.2, "Define the service in an HTM file".
- Section 3.3.2.1, "Load the service in the custom component HDA file".

3.3.2 Define the service in an HTM file

The HTM file must include a table that is identical in structure to the *StandardServices* table. See Section 3.1, "Service Structure Overview."

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
```

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

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

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

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

Here is an example of a 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 3–6 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@>

3.3.2.1 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, *DomainHome*/ucm/cs/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
```

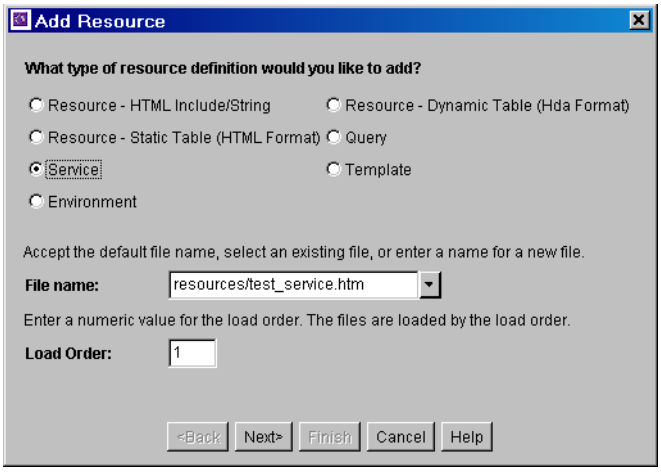
3. Save the file.

3.3.3 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**.

Figure 3–7 Defining a New Custom Service



1. In the Add Resource window, select the **Service** option.
2. Enter the file name for the resource file. The default file name is `resources/`*`componentname`*`_service.htm`.

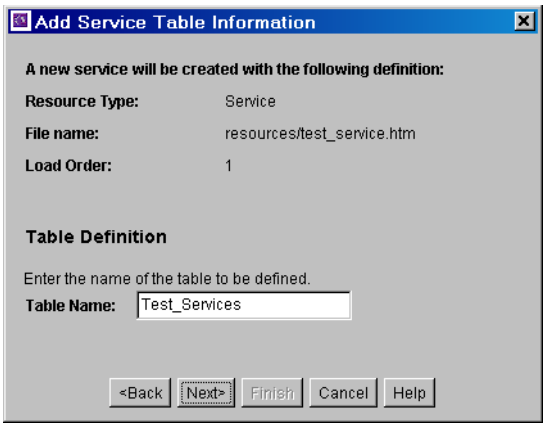
   If a resource file has been created for services, you can append the new service table to the existing file by selecting the file name. Any changes you make to the load order will apply to the entire resource file.

   To create a new resource file with a different file name, enter the file name. For example, *my_services.htm*.
3. 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.

4. Click **Next**.

Figure 3–8 Defining a New Service Table



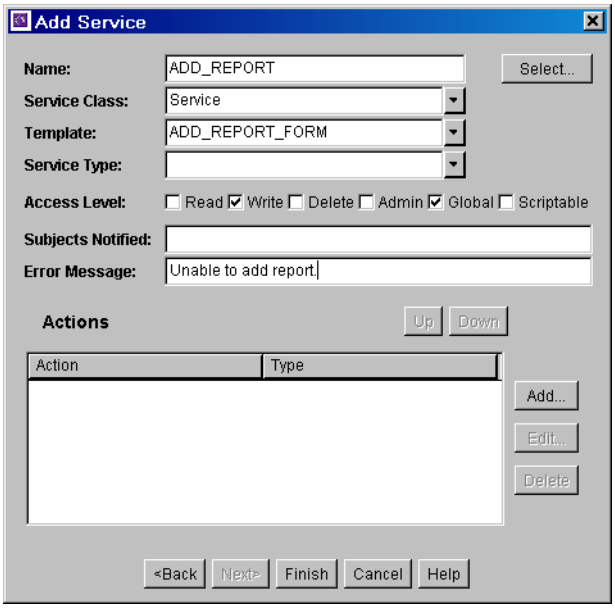
1. In the Add Service Table Information, enter a name for the service table.

   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.

2. Click **Next**.

Figure 3–9 Defining Service Attributes



1. In the Add Service window, 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**.

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

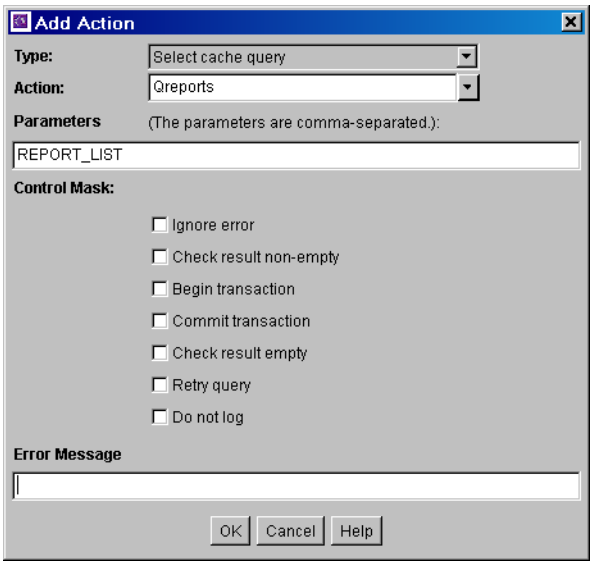
d. Click **OK**. The service attributes and actions are filled in.

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.

2. 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**.

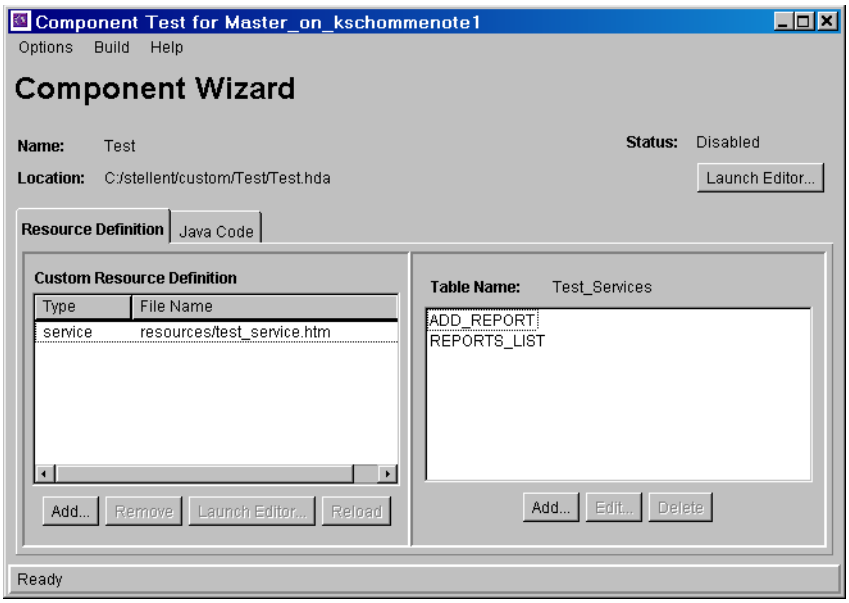
Figure 3–10 Defining an Action



1. Click **Finish**.
2. A dialog box prompts you to launch the text editor to continue editing. 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 3–11 Custom Service Resource Defined in the Component Wizard



4

Core Content Server Services

This chapter describes the core services available for Oracle WebCenter Content Server. The services are divided according to Service Class.

This chapter covers the following topics:

- About Core Content Server Services
- General Services (Core Content Server)
- Doc Services (Core Content Server)
- Doc Profile Services (Core Content Server)
- File Services (Core Content Server)
- Indexer Services (Core Content Server)
- Internal Services (Core Content Server)
- Meta Services (Core Content Server)
- Miscellaneous Services (Core Content Server)
- Page Handler/Page Request Services (Core Content Server)
- Provider Manager Services (Core Content Server)
- Schema Services (Core Content Server)
- Search Services (Core Content Server)
- User Services (Core Content Server)
- Collaboration Services (Core Content Server)

4.1 About Core Content Server Services

Information about what is an Oracle WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific core Content Server services are listed within each individual service.

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

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.

4.2 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
- ADD_DOCEXTENSION
- ADD_DOCFORMAT
- ADD_DOCTYPE
- APPLET_DOCINFO
- CONFIG_INFO
- DELETE_DOC_ACCOUNT
- DELETE_DOCEXTENSION
- DELETE_DOCFORMAT
- DELETE_DOCTYPE
- EDIT_DOCEXTENSION
- EDIT_DOCFORMAT
- EDIT_DOCTYPE
- EDIT_TRACE_OPTIONS
- GET_DATARESULTSET
- GET_DOCEXTENSIONS
- GET_DOCFORMATS
- GET_DOCTYPES
- GET_FIELD_LENGTHS
- GET_FILELIST
- GET_METADEFS
- GET_RESULT_OPTIONS
- GET_SYSTEM_AUDIT_INFO
- GET_TABLE
- GET_USER_METADEFS
- JAVA_PROPERTIES
- LM_BUILD_WEB_STRING_FILES
- LM_LOAD_LAYOUTS

- LM_LOAD_LAYOUTS_SUB
- LOAD_DOC_ENVIRONMENT
- *LOGIN
- MERGE_TABLE
- *PING_SERVER
- QUERY_DOC_ACCOUNTS
- SOAP_FAULT

4.2.1 ADD_DOC_ACCOUNT

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocAccount: The security account for the content item.

Example

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

4.2.2 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
```

4.2.3 ADD_DOCFORMAT

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

Location: *IdcHomeDir*3

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
```

4.2.4 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: *IdcHomeDir*/resources/core/tables/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
```

4.2.5 APPLET_DOCINFO

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

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

4.2.6 CONFIG_INFO

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.7 DELETE_DOC_ACCOUNT

Service that deletes an existing account.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocAccount: The account name.

Example

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

4.2.8 DELETE_DOCEXTENSION

Service that deletes an existing file extension.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

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

Example

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

4.2.9 DELETE_DOCFORMAT

Service that deletes an existing document format.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.2.10 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocType: The content item type.

Example

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

4.2.11 EDIT_DOCEXTENSION

Service that modifies an existing file extension.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.2.12 EDIT_DOCFORMAT

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

Location: *IdcHomeDir*/resources/core/tables/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
```

4.2.13 EDIT_DOCTYPE

Service that modifies an existing content item type.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.2.14 EDIT_TRACE_OPTIONS

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.15 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 result set 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 result set 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: *IdcHomeDir*/resources/core/tables/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 result set of the query.

4.2.16 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.17 GET_DOCFORMATS

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.18 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.19 GET_FIELD_LENGTHS

Service that returns maximum length information for length constrained fields.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Optional Service Parameter

- tableNames: A comma-delineated list of tables to investigate. If a list is not specified, tableNames is set to list the document and folder tables (DocMeta,Documents,Revisions,FolderFolders,FolderFiles).

Results

- ResultSet:
  - FIELD_LENGTH: Result set with field maximum number of characters. This ResultSet contains two columns:
    - fieldName: The name of the field.
    - fieldLength: The maximum number of characters allowed for the field.

4.2.20 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: *IdcHomeDir*/resources/core/tables/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.

4.2.21 GET_METADEFS

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.22 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.2.23 GET_SYSTEM_AUDIT_INFO

Service that retrieves system audit information for the Content Server.

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.2.24 GET_TABLE

Service that exports a database table to a result set 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 result set for later usage.

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

Location: *IdcHomeDir*`/resources/core/tables/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
```

4.2.25 GET_USER_METADEFS

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

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.2.26 JAVA_PROPERTIES

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

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.2.27 LM_BUILD_WEB_STRING_FILES

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

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.2.28 LM_LOAD_LAYOUTS

Service that loads user interface layouts from the *DomainHome*/ucm/cs/weblayout/common/layouts directory. Calls LM_LOAD_LAYOUTS_SUB.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.29 LM_LOAD_LAYOUTS_SUB

SubService used to call layouts from the *DomainHome*/ucm/cs/weblayout/common/layouts directory.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.30 LOAD_DOC_ENVIRONMENT

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.31 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: *IdcHomeDir*/resources/core/templates/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
```

4.2.32 MERGE_TABLE

Service that merges a result set 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 terminates 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 Content Server due to constraint violations.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- tableName: The name of the database table to merge into. The result set 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-delimited 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 terminate 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 terminates 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
```

4.2.33 PING_SERVER

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

> **Tip:** Execute a PING_SERVER request before calling other services to ensure that there is a connection to the Content Server instance and that you are logged in as a user authorized to execute commands.

Access Level: N/A (0)

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Results

- Local Data:
  - dUser
  - StatusMessage
- 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
```

4.2.34 QUERY_DOC_ACCOUNTS

Service that returns a list of all accounts in the Content Server.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.2.35 SOAP_FAULT

Service used to process failed SOAP requests.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3 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 asterisk (*) in the following list.

This section describes these services:

- ASSIGN_DOCINFO_FORM
- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_ARCHIVE_NO_NOTIFY
- CHECKIN_BYNAME
- *CHECKIN_CONFIRM_FORM
- *CHECKIN_LIST
- *CHECKIN_NEW
- *CHECKIN_NEW_FORM
- *CHECKIN_NEW_SUB
- *CHECKIN_SEL
- *CHECKIN_SEL_FORM
- CHECKIN_SEL_SUB
- *CHECKIN_SIMILAR_FORM
- *CHECKIN_UNIVERSAL

- *CHECKOUT
- CHECKOUT_BY_NAME
- *CHECKOUT_OK
- *CHECKOUT_SUB
- CONTINUE_CHECKIN
- CONTINUE_SUBMIT_HTML_FORM
- COPY_REVISION
- CREATE_SUBSCRIPTION_TYPE
- DELETE_BYCLASS
- DELETE_BYNAME
- DELETE_BYREV
- DELETE_BYREV_REVISION
- DELETE_CHECKIN_CACHE
- DELETE_DOC
- *DELETE_REV
- DELETE_REV_EX
- DELETE_SUBSCRIPTION_TYPE
- DOC_FORMATS_WIZARD
- *DOC_INFO
- *DOC_INFO_BY_NAME
- *DOC_INFO_LATESTRELEASE
- *DOC_INFO_SIMPLE
- *DOC_INFO_SIMPLE_BYREV
- *DOC_SUBS_LIST
- EDIT_DOC_FORMATS
- FORM_PROCESS
- FORM_SUBMIT
- GET_CACHED_CHECKIN_INFO
- GET_DOC_CONFIG_INFO
- GET_DOC_SUBSCRIBERS
- GET_DOCUMENT_HISTORY_REPORT
- GET_ENVIRONMENT
- *GET_EXPIRED
- GET_PACKAGE_ENVIRONMENT_PAGE
- *GET_UPDATE_FORM
- NOTIFY_INDEXER
- ODMA_DOC_INFO_SIMPLE

- PACKAGE_ENVIRONMENT
- REMOVE_METAFILE_SUB
- REPLACE_METAFILE_SUB
- RESTORE_REVISION
- *RESUBMIT_FOR_CONVERSION
- *REV_HISTORY
- SELECTDOC
- *SUBMIT_HTML_FORM
- *SUBSCRIBE
- SUBSCRIBE_DOC_USER
- SUBSCRIBE_EX
- *SUBSCRIBE_FORM
- *SUBSCRIPTION_LIST
- *UNDO_CHECKOUT
- UNDO_CHECKOUT_BY_NAME
- *UNSUBSCRIBE
- *UNSUBSCRIBE_FORM
- *UNSUBSCRIBE_FROM_LIST
- UNSUBSCRIBE_FROM_LIST_EX
- UPDATE_BYREV
- *UPDATE_DOCINFO
- *UPDATE_DOCINFO_BYFORM
- UPDATE_DOCINFO_BYREV
- UPDATE_DOCINFO_METAFILE_BYREV
- UPDATE_DOCINFO_STATUS
- UPDATE_DOCINFO_SUB
- UPDATE_SUBSCRIPTION_NOTIFY
- UPDATE_SUBSCRIPTION_TYPE
- UPDATE_SUBSCRIPTION_USED
- UPDATE_METADATA
- VALIDATE_DOCINFO
- *WORK_IN_PROGRESS

4.3.1 ASSIGN_DOCINFO_FORM

Service that retrieves the DOCINFO_FORM and assigns it to content.

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.3.2 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.

To disallow the check in of empty files, set the following:
`ValidatePrimaryFileNotEmpty=1`

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.3.3 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.

To disallow the check in of empty files, set the following:
`ValidatePrimaryFileNotEmpty=1`

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.3.4 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.3.5 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.

To have the service return an error if the content item file is empty, use the configuration setting `ValidatePrimaryFileNotEmpty=1`.

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.3.6 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

Additional Required Service Parameters

For additional parameters, see CHECKIN_ARCHIVE.

Optional Service Parameters

For additional parameters, see CHECKIN_ARCHIVE.

Example

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

4.3.7 CHECKIN_BYNAME

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

Location: *IdcHomeDir*/resources/core/tables/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 or AllowAlternateMetaFile environment variables, or both) 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.

To disallow the check in of empty files, set the following:
`ValidatePrimaryFileNotEmpty=1`

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 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, you can use a metafile in its place. Only one metafile can exist for each content item (a primary AND alternate meta file cannot coexist).
  - 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 coexist.)

  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
```

4.3.8 CHECKIN_CONFIRM_FORM

Service that returns confirmation upon successful checkin through a browser.

Access Level: Write, Global, Scriptable (50)

Location: *IdcHomeDir*/resources/core/tables/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

4.3.9 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: *IdcHomeDir*/resources/core/tables/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

- ResultSets:
  - 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

4.3.10 CHECKIN_NEW

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

Access Level: Write (2)

Location: *IdcHomeDir*`/resources/core/tables/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 or AllowAlternateMetaFile environment variables, or both) 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 Content Server.
- To disallow the checkin of empty files, set the following: `ValidatePrimaryFileNotEmpty=1`

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 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, you can use a metafile in its place. Only one metafile can exist for each content item (a primary AND alternate meta file cannot coexist).
  - 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 coexist.)
  - 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 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*.
- DirectReleaseNewCheckinDoc: Directs content items being checked in to bypass Inbound Refinery, workflow, and indexing. The default is 0. To enable, set `DirectReleaseNewCheckinDoc=1`.

  This parameter can be used to support a high ingestion rate when transferring content items from an external repository into Content Server storage. Because this

setting increases the speed of content checkin, it increases the number of check-ins that Content Server can support at a time. This capability is highly desirable in applications (for example, scanning applications) where large numbers of items are expected to be checked in at once.

Before you enable this setting, take into account the following important considerations.

- A content item checked in this way can only be found by searching with DATABASE.METADATA. However, other content items can still go through regular checkin and be FULLTEXT indexed.
- A rebuild of the index does not cause content checked in with this parameter to go through indexing.
- Content modification (updating or checking in a new revision) causes indexing to occur on a document checked in with this parameter.

- 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.
- 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.
- 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
```

4.3.11 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: *IdcHomeDir*/resources/core/templates//std_services.htm

Results

- ResultSets:
  - 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

4.3.12 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: *IdcHomeDir*`/resources/core/templates//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

4.3.13 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: *IdcHomeDir*`/resources/core/tables/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).
- If metafile generation is enabled (by setting the AllowPrimaryMetaFile or AllowAlternateMetaFile environment variables, or both) 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.
- To disallow the check in of empty files, set the following: `ValidatePrimaryFileNotEmpty=1`

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 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, you can use a metafile in its place. Only one metafile can exist for each content item (a primary AND alternate meta file cannot coexist).
  - 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 coexist.)
  - 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.
- dPublishState: The publish state. If the content item exists in the system dPublishState must be empty.
- 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
```

4.3.14 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 cannot be found, or when the system is unable to check revision properties.

Access Level: Write, Scriptable (34)

Queries Executed: QdocID, Qrevisions, QdocInfo

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Optional Service Parameters

- dDocName
- dWfName

Results

- ResultSets:
  - 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
```

4.3.15 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.16 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Results

- ResultSets:
  - 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)

4.3.17 CHECKIN_UNIVERSAL

Service that performs an Content Server controlled checkin.

Access Level: Write (2)

Location: *IdcHomeDir*/resources/core/tables/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 slash (/) as the file separator. This is because the backslash (\) 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 system, 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 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, you can use a metafile in its place. Only one metafile can exist for each content item (a primary AND alternate meta file cannot coexist).
  - 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 coexist.)
  - 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.
- For additional information, see WORKFLOW_CHECKIN.

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
```

4.3.18 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: *IdcHomeDir*/resources/core/tables/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
```

4.3.19 CHECKOUT_BY_NAME

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

Location: *IdcHomeDir*/resources/core/tables/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
```

4.3.20 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: *IdcHomeDir*/resources/core/tables/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

- ResultSets: 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
```

4.3.21 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: *IdcHomeDir*/resources/core/tables/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 through the processForm method)

4.3.22 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

4.3.23 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

4.3.24 COPY_REVISION

Service that copies one document revision, creating a new rev class from it. Most of the document metadata from the original revision will be carried over. The caller can also overwrite any metadata value by setting it in the request parameters.

The user only needs READ permissions to the original document revision. However, because the new revision inherits the metadata of the original, new security metadata values must be set if the user does not have WRITE permissions to the original revision.

This service assumes that GetCopyAccess is set to TRUE in the Content Server configuration.

Service Class: DocService

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Required Service Parameters

- dID: (integer) The ID of the document revision to copy.
- dDocName: The content ID of the document revision to copy. Used for nice error messages.

Optional Service Parameter

- newDocName: The dDocName of the new rev class. If this is not specified and auto-numbering is enabled, a dDocName will be generated automatically.

Results

- ResultSets:
  - NewDocInfo: Information about the newly created document.
  - OriginalDocInfo: Information about the original document that was copied.

4.3.25 CREATE_SUBSCRIPTION_TYPE

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

Location: *IdcHomeDir*/resources/core/tables/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
```

4.3.26 DELETE_BYCLASS

SubService used to delete content items based on specific parameters.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.27 DELETE_BYNAME

SubService used to delete content items based on specific parameters.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.28 DELETE_BYREV

SubService used to delete content items based on specific parameters.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.29 DELETE_BYREV_REVISION

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.30 DELETE_CHECKIN_CACHE

SubService used to delete content items based on specific parameters.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.31 DELETE_DOC

Service that deletes the entire content item and all revisions. The user must have permission to delete the content item. The most likely error is when the content item no longer exists in the system.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The generated content item ID.
- dDocName: The documentID for the document; the actual content ID.

Example

```
IdcService=DELETE_DOC
dID=48
dDocName=test1000
```

4.3.32 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Results

- ResultSets: 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
```

4.3.33 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

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

4.3.34 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- scpType: The subscription type.

Example

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

4.3.35 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.36 DOC_INFO

Service that retrieves content item information for a specific revision of a content item. It returns all metadata associated with that revision of the 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 through 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The content item revision ID.
- dDocName: The content item ID.

Optional Service Parameters

- includeFileRenditionsInfo: This parameter is set to *false* by default. Set it to *true* on request to obtain the RENDITIONS_INFO table and the Renditions table in the result set.

Results

- ResultSets:

- 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)
- RENDITIONS_INFO: This result set is returned if includeFileRenditionsInfo = 1. It contains data about all the renditions for a content item. It has the following columns:
  * rendition: This column contains any of the following values:

    primary: Indicates the row for the checked-in file.

    web: Indicates the row for the web-viewable file.

    rendition:<dRendition1 flag>: Indicates the type of rendition in dRendition1.

    rendition:<dRendition2 flag>: Indicates the type of rendition in dRendition2.
  * isExtRendition: Indicates if the rendition is an attachment (it is *false* for the core rows, that is, for primary, web, rendition:<dRendition1 flag>, and rendition:<dRendition2 flag> rows; otherwise, it is *true*).
  * isAvailable: This field is always *true*.
  * isStaticAvailable: This field has no relevance for core renditions (when isExtRendition = 0, it is *false* for the core rows, that is, for primary, web, rendition:<dRendition1 flag>, and rendition:<dRendition2 flag> rows). This field checks if a static URL is available for the renditions listed under rendition or attachments. It is *true* if the rendition exists on the file system, and it is *false* if the rendition is not on the file system.
- Renditions: This result set is returned if includeFileRenditionsInfo = 1. It contains data from the Manifest table and MediaRenditions table that are accessible to the Application Development Framework (ADF) UI. The Renditions table has the following columns:
  * rendName: The rendition name. This field is localized when it is possible.
  * rendDescription: Provides a description of the rendition when the rendition is defined. This field is localized when it is possible. Video renditions do not use this field.
  * rendType: Indicates the type of rendition. There are four static types of renditions:

    primary: The checked-in file.

    web: The web-viewable file.

    dc: Follows the dynamic conversion (dc) rules in the native UI. Configure DynamicConveter for this field to appear.

    video: A rendition is considered to be a video if it is listed in the MediaRenditions table or if it matches extRenditionType from the manifest.
  * rendUrl: The absolute URL to the rendition.
  * rendFormat: The format of the rendition, for example, mime-type.

* rendFileSize: The file size of the rendition.
* rendImgDimensions: The dimensions of an image rendition in the *h x w* format.
* rendImgResolution: The resolution of an image rendition in the *nnn dpi* format.
* rendVidFrameRate: The frame rate of a video rendition in the *nn fps* format.
* rendVidDuration: The length in time of a video rendition rounded off to the nearest second, for example, 14 seconds, 1 minute 10 seconds, and so on.
* rendVidBitrate: The bit rate of a video rendition in the *nn bps* format.
* rendParams: Contains a copy of the MediaRenditions.mediaRendParams field for a video or a copy of the manifest.extRenditionParams field for an image.

Note: Each rendition may not use all the fields listed in the Renditions table, and the fields not used by a rendition are marked Not Applicable.

- 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
- String: DocURL (the full URL to the weblayout file associated with this revision)
- 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
```

4.3.37 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: *IdcHomeDir*/resources/core/tables/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.
- includeFileRenditionsInfo: This parameter is set to *false* by default. Set it to *true* on request to obtain the RENDITIONS_INFO table and the Renditions table in the result set.

Results

- ResultSets:
  - 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)
  - RENDITIONS_INFO: This result set is returned if includeFileRenditionsInfo = 1. It contains data about all the renditions for a content item. It has the following columns:
    * rendition: This column contains any of the following values:

      primary: Indicates the row for the checked-in file.

      web: Indicates the row for the web-viewable file.

      rendition:<dRendition1 flag>: Indicates the type of rendition in dRendition1.

      rendition:<dRendition2 flag>: Indicates the type of rendition in dRendition2.

    * isExtRendition: Indicates if the rendition is an attachment (it is *false* for the core rows, that is, for primary, web, rendition:<dRendition1 flag>, and rendition:<dRendition2 flag> rows; otherwise, it is *true*).
    * isAvailable: This field is always *true*.
    * isStaticAvailable: This field has no relevance for core renditions (when isExtRendition = 0, it is *false* for the core rows, that is, for primary, web, rendition:<dRendition1 flag>, and rendition:<dRendition2 flag> rows). This field checks if a static URL is available for the renditions listed under rendition or attachments. It is *true* if the rendition exists on the file system, and it is *false* if the rendition is not on the file system.

– Renditions: This result set is returned if includeFileRenditionsInfo = 1. It contains data from the Manifest table and MediaRenditions table that are accessible to the Application Development Framework (ADF) UI. The Renditions table has the following columns:
    * rendName: The rendition name. This field is localized when it is possible.
    * rendDescription: Provides a description of the rendition when the rendition is defined. This field is localized when it is possible. Video renditions do not use this field.
    * rendType: Indicates the type of rendition. There are four static types of renditions:

      primary: The checked-in file.

      web: The web-viewable file.

      dc: Follows the dynamic conversion (dc) rules in the native UI. Configure DynamicConveter for this field to appear.

      video: A rendition is considered to be a video if it is listed in the MediaRenditions table or if it matches extRenditionType from the manifest.
    * rendUrl: The absolute URL to the rendition.
    * rendFormat: The format of the rendition, for example, mime-type.
    * rendFileSize: The file size of the rendition.
    * rendImgDimensions: The dimensions of an image rendition in the *h x w* format.
    * rendImgResolution: The resolution of an image rendition in the *nnn dpi* format.
    * rendVidFrameRate: The frame rate of a video rendition in the *nn fps* format.
    * rendVidDuration: The length in time of a video rendition rounded off to the nearest second, for example, 14 seconds, 1 minute 10 seconds, and so on.
    * rendVidBitrate: The bit rate of a video rendition in the *nn bps* format.
    * rendParams: Contains a copy of the MediaRenditions.mediaRendParams field for a video or a copy of the manifest.extRenditionParams field for an image.

Note: Each rendition may not use all the fields listed in the Renditions table, and the fields not used by a rendition are marked Not Applicable.

- 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

4.3.38 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocName: The document name for the content item.

Results

- ResultSets: 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
```

4.3.39 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The Content ID for the content item.

Results

- ResultSets:
  - 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

4.3.40 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocName: The document name for the content item.
- dRevLabel: The revision number for the content item.

Results

- ResultSets:
  - DOC_INFO (all fields from Revisions and DocMeta for specific revision)
- Local Data:
  - dUser
  - dRevLabel
  - dID
  - AuthorAddress
  - dDocName

4.3.41 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: *IdcHomeDir*/resources/core/tables/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-delimited 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: Use 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 is necessary. 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 parameter 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

- ResultSets:
  - 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 is not 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
```

4.3.42 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: *IdcHomeDir*/resources/core/tables/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
```

4.3.43 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Note: This service cannot be executed by Idc Command.

4.3.44 FORM_SUBMIT

Service that submits a PDF form. This service calls the FORM_PROCESS service to process the PDF form before submission.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Note: This service cannot be executed by Idc Command.

4.3.45 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

4.3.46 GET_DOC_CONFIG_INFO

Service that returns Content Server configuration information. Gets a dump of the server environment form external applications.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Results

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

- DOC_DEFAULT_INFO
- DocMetaDefinition
- DocTypes
- DocFormats

Example

- IdcCommand command file format:

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

- HDA format:

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

4.3.47 GET_DOC_SUBSCRIBERS

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

Example

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

4.3.48 GET_DOCUMENT_HISTORY_REPORT

Service that returns changes in the DocumentHistory table to be used by cache services and others objectives.

Service Class: DocService

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Optional Service Parameter

- actionDateGreaterThan: A date in ODBC format used for the query against the DocumentHistory table. Two minutes are automatically subtracted from this date to anticipate clustered servers where the clocks can be slightly misaligned. This parameter defaults to five minutes before the current time

Results

- ResultSets:
  - HistoryReport: Each row contains information about a document history event: rename, update, move, or delete.

Example

```
IdcService=GET_DOCUMENT_HISTORY_REPORT
actionDateGreaterThan={ts '2012-11-19 15:58:00.100'}
```

4.3.49 GET_ENVIRONMENT

Service used to retrieve the settings of configuration values for a Content Server environment.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Optional Service Parameters

- startTime: The time (in milliseconds) that the instance started.

4.3.50 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: *IdcHomeDir*/resources/core/tables/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

4.3.51 GET_PACKAGE_ENVIRONMENT_PAGE

Services used to retrieve the page that initiates the packaging of the environment.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.52 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Results

- ResultSets:
  - 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
```

4.3.53 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.54 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The unique ID for the item revision.

4.3.55 PACKAGE_ENVIRONMENT

Service that creates a packaged zip file containing Content Server environment files.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.56 REMOVE_METAFILE_SUB

SubService called to alter document meta information.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.57 REPLACE_METAFILE_SUB

SubService called to replace document meta information.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.58 RESTORE_REVISION

Service that restores the document revision corresponding to the supplied dID as a new (latest) revision.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Required Service Parameter

- dID: The document ID of the item to restore as the latest revision.

Results

- dID: The document ID of the new revision created.
- dDocName: The document name of the new revision created.
- dDocTitle: The document title of the new revision created.

4.3.59 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 and failed ones.

Access Level: Write (2)

Queries Executed: QdocInfo, QlatestID, Qdocuments

Location: *IdcHomeDir*/resources/core/tables/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
```

4.3.60 REV_HISTORY

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

Access Level: Read, Scriptable (33)

Queries Executed: QdocInfo, QrevHistoryReleased

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Results

- ResultSets:
  - 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
```

4.3.61 SELECTDOC

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

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

4.3.62 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: *IdcHomeDir*/resources/core/tables/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

- ResultSets:
  - 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
```

4.3.63 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. Note that to create a subscription, authentication is required.

Access Level: Read (1)

Queries Executed:

- QdocInfo
- QuserSubscription
- Quser
- UuserEmail
- Isubscription

Location: *IdcHomeDir*/resources/core/tables/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 through 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@example.com>
```

4.3.64 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: *IdcHomeDir*/resources/core/tables/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 the following example).

Example

These parameters are passed in the browser environment to execute this command:

```
<form name=SubscriptionForm action="/cs/idcplg"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="/cs/idcplg?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>
```

4.3.65 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: *IdcHomeDir*/resources/core/tables/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-delimited list of the values of the criteria fields.

For example, if the criteria fields are Author and Security Group, the dSubscriptionID parameter value 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@example.com
resultName=USER_LIST
dataSource=Subscriptions
whereClause=dSubscriptionType='test'
```

4.3.66 SUBSCRIBE_FORM

Service that retrieves the subscription form used for content item subscriptions.

This service is identical to the UNSUBSCRIBE_FORM service except the template it returns.

Access Level: Read (1)

Queries Executed: QdocInfo, QSubscribed

Location: *IdcHomeDir*/resources/core/tables/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 parameter except a link to subscribe is provided if the user is not already subscribed to the subscription (the standard value should be SUBSCRIBE).

Results

- ResultSets:
  - 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

4.3.67 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dUser: The user name of the currently logged-in user.

Results

- ResultSets: 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
```

4.3.68 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: *IdcHomeDir*`/resources/core/tables/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
```

4.3.69 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: *IdcHomeDir*/resources/core/tables/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
```

4.3.70 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: *IdcHomeDir*/resources/core/tables/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-delimited list of the values of the criteria fields.

For example, if the criteria fields are Author and Security Group, the dSubscriptionID parameter setting 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
```

4.3.71 UNSUBSCRIBE_FORM

Service that retrieves the form used when unsubscribing from a content item.

This service is identical to the SUBSCRIBE_FORM service except the template it returns.

Access Level: Read (1)

Queries Executed: QdocInfo, QisSubscribed

Location: *IdcHomeDir*/resources/core/tables/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

- ResultSets:
  - 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

4.3.72 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: *IdcHomeDir*/resources/core/tables/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-delimited list of the values of the criteria fields.

For example, if the criteria fields are Author and Security Group, the dSubscriptionID parameter setting 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

- ResultSets: 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
```

4.3.73 UNSUBSCRIBE_FROM_LIST_EX

Service used by the Repository Manager applet to remove a user or alias from the subscription type. For information about adding a user or alias, see SUBSCRIBE_EX.

This service is normally run by deleting a user from a subscription in the Repository Manager applet.

Location: *IdcHomeDir*/resources/core/tables/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-delimited list of the values of the criteria fields.

For example, if the criteria fields are Author and Security Group, the dSubscriptionID parameter setting 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@example.com
resultName=USER_LIST
dataSource=Subscriptions
whereClause=dSubscriptionType='test'
```

4.3.74 UPDATE_BYREV

SubService used to check in content items based on specific parameters.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.75 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: *IdcHomeDir*/resources/core/tables/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.
- SkipIndexingForUpdate: If set to *true*, the content item will not be indexed as part of the update. (The normal behavior is that an item will be re-indexed after an update.) In either case, content items will be indexed in a full index rebuild.

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
```

4.3.76 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: *IdcHomeDir*/resources/core/tables/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
```

4.3.77 UPDATE_DOCINFO_BYREV

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.78 UPDATE_DOCINFO_METAFILE_BYREV

Location: *IdcHomeDir*/resources/core/templates//std_services.htm

4.3.79 UPDATE_DOCINFO_STATUS

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.80 UPDATE_DOCINFO_SUB

SubService that updates content item information.

Location*I: IdcHomeDir*/resources/core/tables/std_services.htm

4.3.81 UPDATE_SUBSCRIPTION_NOTIFY

Currently unused in the core Content Server software.

Service that updates the notification timestamp and redirects the user to the page specified in the RedirectUrl parameter.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.82 UPDATE_SUBSCRIPTION_TYPE

Service that updates a Criteria subscription. The most likely error is when the specified subscription type does not exist.

Location: *IdcHomeDir*/resources/core/tables/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-delimited 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
```

4.3.83 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: *IdcHomeDir*/resources/core/tables/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-delimited list of the values of the criteria fields.

For example, if the criteria fields are Author and Security Group, the dSubscriptionID parameter setting 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
```

4.3.84 UPDATE_METADATA

SubService that updates metadata.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.85 VALIDATE_DOCINFO

Service used for metadata-only checkins for a heavy client (InternalUpload = 1).

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.3.86 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: *IdcHomeDir*/resources/core/tables/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

- ResultSets: 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

4.4 Doc Profile Services (Core Content Server)

The following services are used when creating, maintaining, or deleting document profiles:

- ADD_DOCPROFILE
- ADD_DOCRULE
- DELETE_DOCPROFILE
- DELETE_DOCRULE
- DOCPROFILE_PREVIEW
- EDIT_DOCPROFILE
- EDIT_DOCPROFILE_TRIGGER

- EDIT_DOCRULE
- GET_DOCPROFILE
- GET_DOCPROFILES
- GET_DOCRULE
- GET_DOCRULES

4.4.1 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: *IdcHomeDir*/resources/core/tables/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.

4.4.2 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: *IdcHomeDir*/resources/core/tables/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.

4.4.3 DELETE_DOCPROFILE

Service used to remove a document profile from the profile listing and removes the content profile definition file.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dpName: The name for the profile.

4.4.4 DELETE_DOCRULE

Service used to remove a content rule from the rule listing and deletes the content rule definition file from the system.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dpRuleName: The name for the rule.

4.4.5 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 and which rule determined the display for each field.

Location: *IdcHomeDir*/resources/core/tables/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.

4.4.6 EDIT_DOCPROFILE

Service used to edit a document profile.

Location: *IdcHomeDir*/resources/core/tables/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.

4.4.7 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: *IdcHomeDir*/resources/core/tables/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.

4.4.8 EDIT_DOCRULE

Service that changes a document rule description.

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

Additional Required Service Parameters

- dpRuleName: The name of the rule to be edited.
- dpRuleDescription: The new description for the rule.

4.4.9 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

Additional Required Service Parameters

- dpName: The name of the profile to be used.

4.4.10 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.4.11 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

Additional Required Service Parameters

- dpRuleName: The name of the profile to be used.

4.4.12 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.5 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
- APPEND_FILE_CACHING_INFO

- GET_DYNAMIC_CONVERSION
- GET_DYNAMIC_CONVERSION_SUB
- *GET_DYNAMIC_URL
- GET_TEMPLATE_CONVERSIONS
- GET_WEB_APP_STATUS
- LOAD_RESOURCE_FILE
- REMOVE_WEB_APP
- SAVE_TEMPLATE_CONVERSIONS

4.5.1 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- pathToWAR: Complete path to the WAR file.

4.5.2 APPEND_FILE_CACHING_INFO

SubService used by GET_SYSTEM_AUDIT_INFO to return information on the System Audit Info page.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.5.3 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: *IdcHomeDir*/resources/core/tables/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 system 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 parameter 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*.
- UseConversionCache: Ignores the cached conversion when requesting a dynamic conversion of a content item. It applies only to the single request. You might use this parameter to force the dynamic conversion during testing or troubleshooting a conversion issue. To use this parameter, append it to your Content Server URL.
- 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
```

4.5.4 GET_DYNAMIC_CONVERSION_SUB

SubService used by GET_DYNAMIC_CONVERSION.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.5.5 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- fileUrl: The relative URL for the item. This can be a checked-in item in the Content Server, a Java Script file for Schema or Layout Manager, or an image file from the /images directory.

Results

- ResultSets:
  - 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.)

4.5.6 GET_TEMPLATE_CONVERSIONS

Service used by the Dynamic Converter to retrieve the list of templates that are used during conversion.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.5.7 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.5.8 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 system assume that it is present, as do other features such as forms.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.5.9 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.5.10 SAVE_TEMPLATE_CONVERSIONS

Service used by the Dynamic Converter to save the conversion template.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.6 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
- CONTROL_SEARCH_INDEX
- *GET_FILE
- START_SEARCH_INDEX

4.6.1 CANCEL_SEARCH_INDEX

Service that cancels the current indexing session.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.6.2 CONTROL_SEARCH_INDEX

Service that updates or rebuilds the search index. Updates and rebuilds are performed automatically in a background thread.

Location: *IdcHomeDir*/resources/core/tables/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 after 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 the following examples).

- **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.
- **fastRebuild**: Setting this parameter to 1 (*true*) enables the search engine to add new information to the search collection without requiring a full collection rebuild; this is called a *fast rebuild*. This parameter works only with OracleTextSearch.
- **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 simultaneously.
  - The default is 25, which 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.

Examples

These examples are for controlling the indexer from another process.

- 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
```

4.6.3 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 instance assume that it is present, as do other features such as forms.

Access Level: Read, Scriptable (33)

Queries Executed:

- QdocInfo
- QlatestIdByName
- QlatestReleasedIDByName

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

> **Note:** 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.
- 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
```

4.6.4 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 instance and only completes successfully if a connection is made to the Content Server instance.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.7 Internal Services (Core Content Server)

The following services are called internally by the Content Server system. **Do not use or modify these services**:

- CANCEL_COMPONENT_INSTALL
- CLEAR_SERVER_OUTPUT
- DOWNLOAD_COMPONENT
- GET_COMPONENT_CONFIG
- GET_COMPONENT_INSTALL_FORM
- GET_COMPONENT_INSTALL_PROMPTS_FORM
- GET_COMPONENT_INSTALL_SETTINGS'
- GET_LOCAL_REGISTRATION_FORM
- GET_MANIFEST_INFO
- GET_SERVER_OUTPUT
- PROXIED_REQUEST
- UNINSTALL_COMPONENT
- UPDATE_COMPONENT_CONFIG
- UPDATE_LICENSE
- UPLOAD_NEW_COMPONENT

4.7.1 CANCEL_COMPONENT_INSTALL

Do not use.

4.7.2 CLEAR_SERVER_OUTPUT

Do not use.

4.7.3 DOWNLOAD_COMPONENT

Do not use.

4.7.4 GET_COMPONENT_CONFIG

Do not use.

4.7.5 GET_COMPONENT_INSTALL_FORM

Do not use.

4.7.6 GET_COMPONENT_INSTALL_PROMPTS_FORM

Do not use.

4.7.7 GET_COMPONENT_INSTALL_SETTINGS'

Do not use.

4.7.8 GET_LOCAL_REGISTRATION_FORM

Do not use.

4.7.9 GET_MANIFEST_INFO

Do not use.

4.7.10 GET_SERVER_OUTPUT

Do not use.

4.7.11 PROXIED_REQUEST

Do not use.

4.7.12 UNINSTALL_COMPONENT

Do not use.

4.7.13 UPDATE_COMPONENT_CONFIG

Do not use.

4.7.14 UPDATE_LICENSE

Do not use.

4.7.15 UPLOAD_NEW_COMPONENT

Do not use.

4.8 Meta Services (Core Content Server)

The Meta Services are used to manage and alter metadata in the Content Server system. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- ADD_METADEF
- DEL_METADEF
- EDIT_METADEF
- GET_ADVANCED_SEARCH_OPTIONS
- GET_DISPLAY_FIELDS
- *GET_DOC_METADATA_INFO
- GET_OPTION_LIST
- GET_USER_METADATA_INFO
- MOVE_METADEF
- UPDATE_ADVANCED_SEARCH_OPTIONS
- UPDATE_USER_META
- UPDATE_USER_META_TABLE

- UPDATE_META_TABLE
- UPDATE_OPTION_LIST

4.8.1 ADD_METADEF

Service that creates a new metadata field.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

> **Note:** For more details, see UPDATE_META_TABLE and START_SEARCH_INDEX. These services must be run before storing data in a new metadata field. For details about updating the *OptionLists* database table, see also UPDATE_OPTION_LIST.

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
```

4.8.2 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dName: The metadata field name, including the 'x' prefix.

Example

```
IdcService=DEL_METADEF
dName=xCustomField
```

4.8.3 EDIT_METADEF

Service that modifies an existing custom metadata field.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.8.4 GET_ADVANCED_SEARCH_OPTIONS

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.8.5 GET_DISPLAY_FIELDS

Service that returns information about custom metadata fields for different Content Server pages. This information can be used by front-end clients to render custom metadata fields.

The service requires an action which maps to a specify Content Server page. The service then returns metadata information for that page.

The service also provides support to generate Dynamic Choice List (DCL) options and Dynamic Tree.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Required Service Parameter

- dpAction: Indicates the Content Server action, which can be
  - CheckinNew
  - CheckinSel
  - CheckinSimilar
  - Info
  - Update
  - Search
  - FLDMetadataUpdate
  - FLDMetadataInfo

Conditionally Required Service Parameters

- dID: Required when dpAction is CheckinSel, CheckinSimilar, Info, or Update.
- fFolderGUID: Required when dpAction is either FLDMetadataUpdate or FLDMetadataInfo.

Optional Service Parameter

- dpTriggerValue: Indicates the trigger value that will be used to load a profile.

Results

- ResultSets:
  - DisplayFieldInfo: Contains information about individual custom metadata fields. It holds the following columns:
    * fieldName: The name of the field.
    * fieldType: The type of field. For example: Text, BigText, Memo, Data, Int, Decimal, File.
    * fieldLabel: The display label for the field.
    * isHidden: Indicates whether the field should be a hidden form field.
    * isReadOnly: Indicates whether the field should be a non-input form field.
    * isRequired: Indicates whether the field should be a required form field.

    * defaultValue: The default value for the field.
    * displayValue: The display value to be used for the corresponding default Value.
    * isOptionList: Indicates whether this is a drop-down option list.
    * optionList: If isOptionList is set to 1, then this column will contain the name of the additional result set that will contain the options that should be used to create the drop-down list.
    * optionListType: Type of option list. For example: choice, chunval, combo, multi2 and multi.
    * isDependent: Indicates whether the field is part of DCL. The value of this field depends on another field.
    * dependentOnField: Indicates the name of the field that will derive the value for this field.
    * isPadMultiselectStorage: Used by option list of type multi*.
    * multiselectDisplaySeparator: Used by option list of type multi*.
    * multiselectStorageSeparator: Used by option list of type multi*.
    * isStoreSelectionPath: Used by Dynamic Tree (for more information, see Dynamic Tree).
    * treeNodeDisplaySeparator: Used by Dynamic Tree.
    * treeNodeStorageSeparator: Used by Dynamic Tree.
    * order: Order of the field.
    * decimalScale
    * isError: If there was an error when retrieving information about a field.
    * errorMsg: Error message.

  – DisplayGroupInfo: Indicates whether certain custom metadata fields should be grouped together. It holds the following columns:
    * parentField: The name of the field that should appear first in the group.
    * groupFieldList: The list of fields that should appear together with the parent field.
    * groupHeader: Name of the group.
    * defaultHide: Indicates whether the group should be collapsed by default.
  – xFieldName.options: Contains the options to be used to create the drop-down list for a specific field.
    * dOption: Internal value of the option.
    * dDescription: The display value of the option.

Supported dpAction List

- CheckinNew: Maps to CHECKIN_NEW_FORM.
- CheckinSet: Maps to CHECKIN_SEL_FORM.
- CheckinSimilar: Maps to CHECKIN_SIMILAR_FORM.
- Info: Maps to DOC_INFO.

- Update: Maps to GET_UPDATE_FORM.
- Search: Maps to GET_SEARCH_FORM.
- FLDMetadataUpdate: Maps to FLD_EDIT_METADATA_RULES_FORM.
- FLDMetadataInfo: Maps to FLD_GET_METADATA_RULES.

Dynamic Choice List

The GET_DISPLAY_FIELDS service also provides support to generate Dynamic Choice Lists (DCLs). It uses the following parameters:

- dFieldName: (Required) The name of the DCL field.
- dParentValue: (Optional) The parent value to be used to narrow down the drop-down options.

  If dParentValue is not specified, then the service returns all the options values for any option list field.

The columns "isDependent" and "dependentOnField" in the DisplayFieldInfo ResultSet are used to generate DCLs. These columns specify if a field is dependent on another field, and on which field it is dependent.

For example, if the field *City* is dependent on *State*, and the field *State* is dependent on the field *Country*, the GET_DISPLAY_FIELDS service will return all the options for *Country*, but it will not return options for *State* and *City*. It will only indicate in the DisplayFieldInfo ResultSet that *State* and *City* are dependent fields. The ResultSet will have isDependent set to 1 for both *State* and *City*, and dependentOnField set to *Country* for *State*, and *State* for *City*.

For example, to get the dependent value for *State*, call GET_DISPLAY_FIELDS with the dFieldName parameter set to *State* and the dParentValue parameter set to the value for *Country*. The service will then return all the options that were passed for *State* and for *Country*.

Dynamic Tree

The GET_DISPLAY_FIELDS service also provides support to generate Dynamic Tree. It uses the following parameters:

- dFieldName: (Required) The name of the DCL field.
- dParentValue: (Optional) The parent value that is used to narrow down the drop-down options.

The field that is using the tree option list will have `tree://` in the optionListType column in the DisplayFieldInfo ResultSet. The response binder also will contain another ResultSet with fieldName.options; for example, xTreeField.options.

The xTreeField.options ResultSet will contain the options that represent the first level of the tree.

To retrieve options for any level of the tree, call GET_DISPLAY_FIELDS with both the dFieldName and appropriate dParentValue parameters.

The DisplayFieldInfo ResultSet that is returned by GET_DISPLAY_FIELDS also will contain the following four new columns:

- isShowSelectionPath: Indicates whether to show the full path. The full path appears on the Info page.
- isStoreSelectionPath: Indicates whether to store the full path in the database.

- treeNodeDisplaySeparator: Indicates the character that must be used to display the separators.
- treeNodeStorageSeparator: Indicates the character that must be used as a separator when the path is saved in the database.

Search

When GET_DISPLAY_FIELDS is called with Search as the dpAction, the service adds additional information to the response binder.

- DisplayFieldInfo ResultSet contains two additional columns:
  - defaultOperator: Indicates the default operator for the field.
  - searchOperatorsRSName: Indicates the name of the ResultSet that contains all the applicable search operators for this field.

The service also adds additional ResultSets to the response binder. These ResultSets contain the search operators for different field types:

- SearchTextField: Contains the search operator for Text, Big Text and Memo fields.
- SearchDateField: Contains the search operator for Date fields.
- SearchIntegerField: Contains the search operator for Integer fields.
- SearchBooleanField: Contains the search operator for Boolean fields. Usually does not exist.

These ResultSets also contain the following columns:

- dOperator: The name of the operator.
- dOperatorDisplay: The Localized Display string for the operator.
- dOperatorExpression: The syntax that must be used to create the query text.

4.8.6 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

Results

- ResultSets:
  - DocMetaDefinition (All rows and fields from DocMetaDefinition 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

4.8.7 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.8.8 GET_USER_METADATA_INFO

Service that returns metadata information for users. Used with SOAP retrievals.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.8.9 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: *IdcHomeDir*/resources/core/tables/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.

4.8.10 UPDATE_ADVANCED_SEARCH_OPTIONS

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.8.11 UPDATE_USER_META

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.8.12 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: *IdcHomeDir*/resources/core/tables/std_services.htm

---

Caution: Data may be lost if a column is deleted. Use this service carefully.

Optional Service Parameters

- MetaFieldsToDelete: Specifies a comma-delimited 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
```

4.8.13 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. Use this service carefully.

Optional Service Parameters

- MetaFieldsToDelete: Specifies a comma-delimited 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
```

4.8.14 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: *IdcHomeDir*/resources/core/tables/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
```

4.9 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
- DOWNLOAD_LISTBOX_ITEMS
- LOAD_USER_LOCALIZATION

4.9.1 CHUNKED_UPLOAD

This service adds support for calling an Content Server service that is very large. It chunks the request.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.9.2 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: *IdcHomeDir*/resources/core/tables/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.

4.9.3 LOAD_USER_LOCALIZATION

Service that loads localization information for user.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.10 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
- *GET_ADMIN_PAGE
- *GET_DOC_PAGE
- *GET_DYNAMIC_PAGE
- GET_PERSONALIZED_JAVASCRIPT
- GET_PORTAL_PAGE
- *GET_SECURE_PAGE
- LOAD_GLOBALINCLUDES
- PAGE_HANDLER
- PNE_SAVE_QUERY
- PNE_UPDATE_PERSONAL_URLS
- PNE_UPDATE_PORTAL_INFO
- SAVE_GLOBALINCLUDES
- UPDATE_RESULT_TEMPLATE

4.10.1 DELETE_RESULT_TEMPLATE

Service that deletes an existing search results template.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- name: The search results template name.

Example

```
IdcService=DELETE_RESULT_TEMPLATE
name=test_template
```

4.10.2 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, the service returns an error message.

Access Level: N/A (0)

Location: *IdcHomeDir*/resources/core/tables/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
```

4.10.3 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: *IdcHomeDir*/resources/core/tables/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

- ResultSets:
  - 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
```

4.10.4 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 the Content Server system.
- 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

Additional Required Service Parameters

- PageName: The name of the page template.

Results

- ResultSets:
  - 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
```

4.10.5 GET_PERSONALIZED_JAVASCRIPT

Service that delivers the navigation information for a particular user's page. If a browser optimally caches the JavaScript, this enables the browser to not request the URL on every page load. If the navigation data changes, a numeric ID which is added at the end of the URL is changed and the browser will reload the page.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.10.6 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- Action: The action to execute. Usually Get*TemplatePage*.
- 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
```

4.10.7 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: *IdcHomeDir*/resources/core/tables/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

- ResultSets:
  - 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
```

4.10.8 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** then **Update Portal**. The content shown on that screen is the content that is returned from the Content Server if you run this service.

Location: *IdcHomeDir*/resources/core/templates//std_services.htm

4.10.9 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 instance and only completes successfully if a connection is made to the Content Server instance.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.10.10 PNE_SAVE_QUERY

Service that saves a search query in the User Profile for the current user.

Location: *IdcHomeDir*/resources/core/tables/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

4.10.11 PNE_UPDATE_PERSONAL_URLS

Service that updates the personal URLs in the User Profile for the current user.

Location: *IdcHomeDir*/resources/core/tables/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=Company
websiteEd=http://www.example.com
```

4.10.12 PNE_UPDATE_PORTAL_INFO

Service that updates the System Links in the User Profile for the current user.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.10.13 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: *IdcHomeDir*/resources/core/tables/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 system to update the portal page.
  - Default is 0 (*false*).
- PageFunction: Must be set to `GetPageList`.

Example

```
IdcService=SAVE_GLOBALINCLUDES
PageFunction=GetPageList
IsRebuild=1
```

4.10.14 UPDATE_RESULT_TEMPLATE

Service that updates a search results template.

When you launch the **Web Layout Editor** applet, then select **Options**, then select **Query Result Page** and click **Add**, **Delete**, or **Edit**, enter template information, and click **OK**, this service is executed.

Location: *IdcHomeDir*`/resources/core/tables/std_services.htm`

> **Note:** See the following 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).

4.11 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
- APPEND_DATABASE_AUDIT_INFO
- DELETE_PROVIDER
- ENABLE_DISABLE_PROVIDER
- GET_ADD_EDIT_PROVIDER_FORM
- GET_ALL_PROVIDERS
- GET_PROVIDER_INFO
- NOTIFY_CHANGE
- REQUEST_SECURITYINFO
- TEST_PROVIDER

4.11.1 ADD_EDIT_PROVIDER

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

Location: *IdcHomeDir*/resources/core/tables/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
```

4.11.2 APPEND_DATABASE_AUDIT_INFO

SubService used by GET_SYSTEM_AUDIT_INFO to return information on the System Audit Info page.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.11.3 DELETE_PROVIDER

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

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- pName: The provider name.

Example

```
IdcService=DELETE_PROVIDE
pName=Proxied_2_on_test13
```

4.11.4 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: *IdcHomeDir*/resources/core/tables/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
```

4.11.5 GET_ADD_EDIT_PROVIDER_FORM

Service that returns a form for adding or editing a provider in a browser.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.11.6 GET_ALL_PROVIDERS

Service that returns a list of all providers.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.11.7 GET_PROVIDER_INFO

Service that returns information about a provider.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.11.8 NOTIFY_CHANGE

Service that notifies an Content Server instance of changes to another Content Server instance through an outgoing provider.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.11.9 REQUEST_SECURITYINFO

Service that returns all security and user information from the Content Server instance.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.11.10 TEST_PROVIDER

Service that tests a provider.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- pName: The provider name.

Example

```
IdcService=TEST_PROVIDER
pName=provider22
```

4.12 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
- ADD_SCHEMA_RELATION
- ADD_SCHEMA_VIEW
- ADDOREDIT_SCHEMA_TABLE
- CONTROL_SCHEMA

- DELETE_SCHEMA_FIELD
- DELETE_SCHEMA_RELATION
- DELETE_SCHEMA_TABLE
- DELETE_SCHEMA_VIEW
- EDIT_SCHEMA_FIELD
- EDIT_SCHEMA_NODE
- EDIT_SCHEMA_RELATION
- EDIT_SCHEMA_VIEW
- EDIT_SCHEMA_VIEW_VALUES
- GET_SCHEMA_FIELD_INFO
- GET_SCHEMA_FIELDS
- GET_SCHEMA_RELATIONS
- GET_SCHEMA_STATS
- GET_SCHEMA_TABLE_INFO
- GET_SCHEMA_TABLES
- GET_SCHEMA_VIEW_EDIT_INFO
- GET_SCHEMA_VIEW_FRAGMENT
- GET_SCHEMA_VIEW_INFO
- GET_SCHEMA_VIEW_VALUES
- GET_SCHEMA_VIEWS
- PUBLISH_SCHEMA

4.12.1 ADD_SCHEMA_FIELD

Service that adds a field to a schema table.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schFieldName: The name of the field to be included.

4.12.2 ADD_SCHEMA_RELATION

Service that adds a relation to an existing schema.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schRelationName: The name of the relation to be included.

4.12.3 ADD_SCHEMA_VIEW

Service used to add a new view for a schema.

Location: *IdcHomeDir*/resources/core/tables/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.

4.12.4 ADDOREDIT_SCHEMA_TABLE

Service used to add or edit a table used in a schema.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.12.5 CONTROL_SCHEMA

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.12.6 DELETE_SCHEMA_FIELD

Service used to delete a field from a schema table.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schFieldName: The name of the field to be deleted.

4.12.7 DELETE_SCHEMA_RELATION

Service used to delete a relation from a schema.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schRelationName: The name of the relation to be deleted.

4.12.8 DELETE_SCHEMA_TABLE

Service used to delete a table from a schema.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schTableName: The name of the relation to be deleted.

4.12.9 DELETE_SCHEMA_VIEW

Service used to delete a view from a schema.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be deleted.

4.12.10 EDIT_SCHEMA_FIELD

Service used to edit a field in a schema.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schFieldName: The name of the field to be edited.

4.12.11 EDIT_SCHEMA_NODE

Service used to edit a node in a schema.

Location: *IdcHomeDir*/resources/core/tables/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.

4.12.12 EDIT_SCHEMA_RELATION

Service used to edit a schema relation.

Location: *IdcHomeDir*/resources/core/tables/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.

4.12.13 EDIT_SCHEMA_VIEW

Service used to edit a schema view.

Location: *IdcHomeDir*/resources/core/tables/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.

4.12.14 EDIT_SCHEMA_VIEW_VALUES

Service used to edit values in a schema view.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

4.12.15 GET_SCHEMA_FIELD_INFO

Service that returns information about a specific schema field.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schFieldName: The name of the field to be used.

4.12.16 GET_SCHEMA_FIELDS

Service that returns information about fields used in schemas.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.12.17 GET_SCHEMA_RELATIONS

Service that returns information about relations used with schemas.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.12.18 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.12.19 GET_SCHEMA_TABLE_INFO

Service that returns information about all tables used in schemas.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.12.20 GET_SCHEMA_TABLES

Service that returns information about the tables and information in the tables that are used in schemas.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.12.21 GET_SCHEMA_VIEW_EDIT_INFO

Service that returns the editable information for a schema view.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

4.12.22 GET_SCHEMA_VIEW_FRAGMENT

Service that returns a fragment of a Schema-based option list.

Location: *IdcHomeDir*/resources/core/tables/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.

4.12.23 GET_SCHEMA_VIEW_INFO

Service that returns table and other information for the specified view.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

4.12.24 GET_SCHEMA_VIEW_VALUES

Service that displays values in the specified view.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- schViewName: The name of the view to be used.

4.12.25 GET_SCHEMA_VIEWS

Service that retrieves all defined schema views.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.12.26 PUBLISH_SCHEMA

Service that initiates a publish of all schemas.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.13 Search Services (Core Content Server)

The Search Services are used to manage searching within the Content Server system. Frequently used services are marked with an asterisk (*) in the following list.

The following services are described in this section:

- APPEND_SEARCH_AUDIT_INFO
- GET_EXTERNAL_DOC_INFO
- GET_EXTERNAL_HIGHLIGHT_INFO
- GET_EXTERNAL_XML_HIGHLIGHT_INFO
- GET_HIGHLIGHT_INFO
- *GET_SEARCH_RESULTS
- GET_SEARCH_RESULTS_FORCELOGIN
- GET_XML_HIGHLIGHT_INFO
- PNE_GET_SEARCH_RESULTS
- VIEW_DOC

4.13.1 APPEND_SEARCH_AUDIT_INFO

SubService used by GET_SYSTEM_AUDIT_INFO to return information on the System Audit Info page.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.13.2 GET_EXTERNAL_DOC_INFO

Service that retrieves content information from an external Verity collection. Used with the LightlyManagedContent component.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.
- sCollectionID: The collection ID used by the Content Server system to locate the collection.

Example

```
IdcService=GET_EXTERNAL_DOC_INFO
dDocName=adminform113
```

4.13.3 GET_EXTERNAL_HIGHLIGHT_INFO

Service that returns PDF or HTML highlight information for a content item in an external collection.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.13.4 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: *IdcHomeDir*/resources/core/tables/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
```

4.13.5 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: *IdcHomeDir*`/resources/core/tables/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
```

4.13.6 GET_SEARCH_RESULTS

Service that returns a list of content items that match specific search criteria.

Access Level: Read (1)

Calls SubService: SUB

Location: *IdcHomeDir*`/resources/core/tables/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 `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.

- vcrContentType: The name of a searchable content type. The server modifies the query text of the search to limit the results to documents of that type. For example, if the content type specified is one describing a profile, then the query text is modified to limit the documents returned to those whose profile trigger value matches that of the profile.
- vcrAppendObjectClassInfo: When set to *true*, the server adds an additional column to the SearchResults ResultSet called vcrObjectClass. This column lists the content type associated with each document in the results. The default is *true*.

Example

```
IdcService=GET_SEARCH_RESULTS
QueryText=benefits
```

4.13.7 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: *IdcHomeDir*`/resources/core/tables/std_services.htm`

4.13.8 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: *IdcHomeDir*`/resources/core/tables/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
```

4.13.9 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 available to the GET_SEARCH_RESULTS service.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- QueryText: The search expression.

Example

```
IdcService=PNE_GET_SEARCH_RESULTS
QueryText=benefits
```

4.13.10 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. 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 INSTANCE.
- Usually, the VdkVgwKey is the dDocName. Verity gives a unique ID to each content item, and the Content Server system maps that Verity ID (VdkVgwKey) with dDocName (Content ID).
- Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- QueryText: The search expression.
- sCollectionID: The collection ID used by the Content Server system 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
```

4.14 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
- ADD_GROUP
- ADD_ROLE
- ADD_USER
- CHANGE_USER_AUTH_TYPE
- *CHECK_USER_CREDENTIALS
- DELETE_ALIAS
- DELETE_GROUP
- DELETE_ROLE
- DELETE_USER
- EDIT_ALIAS
- EDIT_GROUP
- EDIT_ROLE
- EDIT_USER
- *EDIT_USER_PROFILE
- GET_ALIASES
- GET_FILTER_ADMIN_PAGE
- GET_SELF_REGISTER_PAGE
- GET_USERDOCPROFILES

- *GET_USER_INFO
- GET_USERS
- LOAD_PNE_PORTAL
- *LOAD_USER_TOPIC
- QUERY_GROUP
- QUERY_USER_ATTRIBUTES
- REGISTER_USER
- *SAVE_USER_TOPICS
- UPDATE_FILTER_INFO
- UPDATE_USEROPTION_LIST

4.14.1 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: *IdcHomeDir*/resources/core/tables/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 result sets:

```
@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
```

4.14.2 ADD_GROUP

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

Location: *IdcHomeDir*/resources/core/tables/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 *Oracle Fusion Middleware Administering Oracle WebCenter Content*.

Example

```
IdcService=ADD_GROUP
dGroupName=NEW_GROUP
```

```
dPrivilege=15
dDescription=admin privileges
```

4.14.3 ADD_ROLE

Service that creates a new role. A role is a set of permissions (Read, Write, Delete, Admin) for each security group.

The service adds a row in the RoleDefinition table for every existing security group. The most likely error is when the role name is not unique.

Location: *IdcHomeDir*/resources/core/tables/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 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 *Oracle Fusion Middleware Administering Oracle WebCenter Content*.

Example

```
IdcService=ADD_ROLE
dRoleName=specialuser
dPrivilege=15
```

4.14.4 ADD_USER

Service used to create a new user. The most likely error is when the user name is not unique.

Location: *IdcHomeDir*/resources/core/tables/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 result set 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-delimited 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@example.com
@end
@ResultSet UserAttribInfo
2
dUserName
AttributeInfo
jsmith
role,admin,15,role,contributor,15
@end
```

4.14.5 CHANGE_USER_AUTH_TYPE

Service that changes the authentication type (global, local, or external) for users.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- curUserAuthType: Current user authentication type.
- dUserAuthType: New user authentication type.

4.14.6 CHECK_USER_CREDENTIALS

Service that checks the credentials of a user. You can use this service to get information about a user (for example, roles and accounts), or use it 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.

This service does not return user data for Microsoft Active Directory managed users when the Content Server instance is configured to use ADSI. It does return the user data for local users and external LDAP managed users.

Access Level: N/A (0)

Location: *IdcHomeDir*/resources/core/tables/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-delimited list of user's accounts. Only returned if optional parameter hasSecurityInfo is 0 (*false*).
  - roles: comma-delimited 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=\ndUserType=\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=
\nu11=\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=\ndUserType=\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=
\nu11=\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
```

4.14.7 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dAlias: The alias name.

Example

```
IdcService=DELETE_ALIAS
dAlias=admin_alias
```

4.14.8 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dGroupName: The security group name.

Example

```
IdcService=DELETE_GROUP
dGroupName=admingroup
```

4.14.9 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dRoleName: The role name.

Example

```
IdcService=DELETE_ROLE
dRoleName=test_role
```

4.14.10 DELETE_USER

Service that deletes an existing user. The most likely error is when the user has been assigned to an alias.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.14.11 EDIT_ALIAS

Service that modifies an existing alias. To edit and add users, the AliasUsersString parameter must be included.

Location: *IdcHomeDir*/resources/core/tables/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
watchedMonikers=
monitoredSubjects=usermetaoptlists,1008541017549,aliases,1008541017549,
userlist,1008541017549,users,1008541017549,metadata,1008541017549,config,100854
1017549,accounts,1008541017549
AliasUsersString=Gillian\nMonique\nsysadmin\n
dAlias=YourAlias
@end
```

- In HDA format, adding users with the AliasUserMap and Alias result sets:

```
@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,1008541017549,users,1008541017549,metadata,1008541017549,config,
1008541017549,accounts,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
```

4.14.12 EDIT_GROUP

Service that modifies an existing security group.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.14.13 EDIT_ROLE

Service that modifies an existing role. A role is a set of permissions (Read, Write, Delete, Admin) for each security group.

Location: *IdcHomeDir*/resources/core/tables/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. For more information about numbering of permissions, see *Oracle Fusion Middleware Administering Oracle WebCenter Content*.

Example

```
IdcService=EDIT_ROLE
dRoleName=test_role
dGroupName=MY_GROUP
dPrivilege=15
```

4.14.14 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: *IdcHomeDir*`/resources/core/tables/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.

Note: If the specified user does not exist, the Content Server adds the user.

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 result set 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-delimited 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@example.com
dUserType=MKT
dUserLocale=English-US
@end

@ResultSet UserAttribInfo
2
dUserName
AttributeInfo
jsmith
role,admin,15,role,contributor,15
@end
```

4.14.15 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: *IdcHomeDir*/resources/core/tables/std_services.htm

- Any of the fields in the Users table (except dName) can be updated by this service.
- You can use this service o 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 Content Server 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

- ResultSets: 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@example.com
dUserType=MKT
dUserLocale=English-US
@end
```

4.14.16 GET_ALIASES

Service that returns a list of all aliases.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.14.17 GET_FILTER_ADMIN_PAGE

Service that retrieves the Configure Web Server Filter page in a browser.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.14.18 GET_SELF_REGISTER_PAGE

Service that returns the self-registration page and loads the user information for a self-registered user.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.14.19 GET_USERDOCPROFILES

Service that returns the checkin and search profiles available to the current user. This service also includes the Standard Search /Checkin information if available to the user.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Results

- ResultSets:
  - SearchProfiles: Result set with search profile information. This result set contains five columns:
    - dpName: The profile name
    - dpDescription: The profile description
    - dpTriggerValue: The trigger value of the profile
    - dpDisplayLabel: The profile display label
    - dpdDocClass: The Document Class associated with the profile
  - CheckInProfiles: Result set with check-in profile information. This result set contains five columns:
    - dpName: The profile name
    - dpDescription: The profile description
    - dpTriggerValue: The trigger value of the profile
    - dpDisplayLabel: The profile display label
    - dpdDocClass: The Document Class associated with the profile

4.14.20 GET_USER_INFO

Service that returns the User Profile page for the current user. Use CHECK_USER_CREDENTIALS 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Results:

- ResultSets:
  - USER_INFO (fields from Users database table for current user)
    - dName: Name of the user
    - dFullName: Full name of the user
    - dEmail: E-mail of the user
    - dUserTimeZone: Time zone of the user
    - dUserLocale: User locale of the user
    - dUserLanguageId: User language ID of the user
  - UserLocaleLanguageMap (available locales for the user)
    - lcLocaleID: Locale ID
    - lcLanguageID: Language ID corresponding to the locale ID
  - 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

4.14.21 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: *IdcHomeDir*/resources/core/tables/std_services.htm

4.14.22 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Results

- ResultSets: 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

4.14.23 LOAD_USER_TOPIC

Loads personalization data based on the topic chosen. The topics correspond to HDA files in the *DomainHome*/ucm/cs/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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- userTopic: The name of the user topic, such as *pne_portal* or *wf_in_queue*.

Results

- ResultSets: All result sets in the specified topic file.
- Local Data: All LocalData properties in the specified topic file.

4.14.24 QUERY_GROUP

Service that returns the description of a security group.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dGroupName: The security group name.

Example

```
IdcService=QUERY_GROUP
dGroupName=Public
```

4.14.25 QUERY_USER_ATTRIBUTES

Service that returns user attributes for a specific user.

This service does not return user data for Microsoft Active Directory managed users when the Content Server instance is configured to use ADSI. It does return the user data for local users and external LDAP managed users.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dName: The user name.

Example

```
IdcService=QUERY_USER_ATTRIBUTES
dName=jsmith
```

4.14.26 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: *IdcHomeDir*/resources/core/tables/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
```

4.14.27 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Actions

The following actions are used with the topicString parameter:

- updateKeys: Updates a single LocalData key variable. The topicString parameter 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 as of Content Server version 10*g*R1. 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 parameter takes the 3 required values only.
- updateRows: This action is used to add or update one or more result set rows using a single topic string. The topicString parameter takes 5 values: the first 3 required values; a fourth that contains a comma-delimited 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, the 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 parameter takes 4 values: the first 3 required values and a fourth that contains a comma-delimited 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 parameter takes 4 values: the first 3 required values, and a fourth that contains a comma-delimited 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 parameter takes only 3 values, but in this case, the third value can be a comma-delimited 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 is not 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

- ResultSets: 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 the Content Server instance. This is a custom feature extension and is only supported through the purchase of Consulting Services or through the purchase of our Developer Support Token Program. As with all customization, 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 must 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.example.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='Company Home Page'>
<INPUT type='hidden' name='website1' value='http://www.example.com'>
<INPUT type='submit' value='Submit'>
</FORM>
```

4.14.28 UPDATE_FILTER_INFO

Service that updates the Web server filter settings.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.14.29 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 *DomainHome*/ucm/cs/data/users/config/ directory.

Location: *IdcHomeDir*/resources/core/tables/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
```

4.15 Collaboration Services (Core Content Server)

The following Collaboration Services ship with the core Content Server software:

- ADD_COLLABORATION
- ADD_COLLABORATION_FORM
- DELETE_COLLABORATION
- EDIT_CLBRA_ACCESS_LIST

- EDIT_CLBRA_ACCESS_LIST_FORM
- EDIT_COLLABORATION
- EDIT_COLLABORATION_FORM
- GET_CLBRA_DOCUMENTS
- GET_CLBRA_INFO
- GET_COLLABORATION_LIST
- GET_USER_CLBRA_LIST

Note: dClbraType is a required parameter for many services. At this time, *basic* is the only accepted value for this parameter.

4.15.1 ADD_COLLABORATION

Service called from ADD_COLLABORATION_FORM. This service calls ADD_COLLABORATION_SUB to add a collaboration project.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dClbraName: The project name.
- dClbraType: The type of collaboration project.

4.15.2 ADD_COLLABORATION_FORM

Service used to present a form to a user to add a Collaboration. After submitting this form, the ADD_COLLABORATION service is called.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.15.3 DELETE_COLLABORATION

Service used to delete a collaboration project.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dClbraName: The project name.

4.15.4 EDIT_CLBRA_ACCESS_LIST

Service used to edit the members in a collaboration project.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dClbraName: The project name.
- dClbraType: The type of collaboration project.

4.15.5 EDIT_CLBRA_ACCESS_LIST_FORM

Service that presents the form used to edit members in a collaboration project.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dClbraName: The project name.

4.15.6 EDIT_COLLABORATION

Service that calls EDIT_COLLABORATION_SUB to update existing project information.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dClbraName: The project name.
- dClbraType: The type of collaboration project.

4.15.7 EDIT_COLLABORATION_FORM

Service that presents the form used to edit a collaboration project.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dClbraName: The project name.

4.15.8 GET_CLBRA_DOCUMENTS

Service used to retrieve content in a collaboration project.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dClbraName: The project name.

4.15.9 GET_CLBRA_INFO

Service used to retrieve information about a specific collaboration project.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- dClbraName: The project name.

4.15.10 GET_COLLABORATION_LIST

Service that returns a list of all collaboration projects.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

4.15.11 GET_USER_CLBRA_LIST

Service that returns a list of users in all collaboration projects.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

5 Workflow Services

This chapter describes the services available when using and customizing Oracle WebCenter Content Server workflows, and other services which are stored in the workflow.htm file in the *IdcHomeDir*/resources/core/templates/ directory.

This chapter covers the following topics:

- About Workflow Services
- Doc and General Services (Workflows)
- Workflow Template Services
- Workflow Services

5.1 About Workflow Services

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific workflow services are listed within each individual service.

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

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.

5.2 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
- DELETE_PROBLEMREPORT

- GET_CRITERIA_WORKFLOWS_FOR_GROUP
- GET_DOCUMENT_PROBLEMREPORTS
- GET_PROBLEMREPORTS_SEARCH_FORM
- GET_PROBLEMREPORTS_SEARCH_RESULTS
- GET_UPDATE_PROBLEMREPORT_FORM
- GET_WORKFLOWDOCUMENTS
- GET_WORKFLOWS_FOR_ALL
- *LOAD_WORKFLOW_QUEUE
- NOTIFY_CONTRIBUTOR
- PROBLEMREPORT_INFO
- RESEND_PROBLEMREPORT
- REVIEW_WORKFLOW_DOC
- UPDATE_PROBLEMREPORT
- WORKFLOW_CHECKIN_SUB
- WORKFLOW_EDIT_REV
- WORKFLOW_NEW_REV
- WORKFLOW_REJECT_FORM

5.2.1 ADD_PROBLEMREPORT

Service that adds a problem report to a content item.

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: *IdcHomeDir*`/resources/core/templates/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*.

> **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.

Example

```
IdcService=ADD_PROBLEMREPORT
```

```
dPrCaption=Problem report for this content.
dDocName=PublicDoc1_ia3c488971
dID=67
dPrState=OPEN
dPrSeverity=MINOR
```

5.2.2 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report ID.

Example

```
IdcService=DELETE_PROBLEMREPORT
dPrID=1
```

5.2.3 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 result sets:

- 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: *IdcHomeDir*/resources/core/templates/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
```

5.2.4 GET_DOCUMENT_PROBLEMREPORTS

Service that returns all problem reports for a specific content item.

Service class: Doc Service

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dDocName: The Content ID for the content item.

Example

```
IdcService=GET_DOCUMENT_PROBLEMREPORTS
dDocName=security_000015
```

5.2.5 GET_PROBLEMREPORTS_SEARCH_FORM

Service that returns the problem report search form.

Service Class: Doc Service

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

5.2.6 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: *IdcHomeDir*/resources/core/templates/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.

5.2.7 GET_UPDATE_PROBLEMREPORT_FORM

Service that returns the update form for a problem report.

Service Class: Doc Service

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report ID.

Example

```
IdcService=GET_UPDATE_PROBLEMREPORT_FORM
dPrID=1
```

5.2.8 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- IdcService: Must be set to GET_WORKFLOWDOCUMENTS.
- dWfName: The workflow name.

Example

```
IdcService=GET_WORKFLOWDOCUMENTS
dWfName=mktg_review
```

5.2.9 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: *IdcHomeDir*/resources/core/templates/workflow.htm

5.2.10 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 *DomainHome*/ucm/cs/data/users/profiles directories.

This is intended to be a remote application service because it is a raw data service with no associated display template.

Service Class: User Service

Access Level: Read, Scriptable (33)

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Results

- Results Sets: WorkflowInQueue

Used By

- Other: SoapCustom:Wsdl:Workflow:Services

5.2.11 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

```
IdcService=NOTIFY_CONTRIBUTOR
dID=55
```

5.2.12 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report's unique ID.

Example

```
IdcService=PROBLEMREPORT_INFO
dPrID=1
```

5.2.13 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dPrID: The problem report ID.

Example

```
IdcService=RESEND_PROBLEMREPORT
dPrID=1
```

5.2.14 REVIEW_WORKFLOW_DOC

Service that generates the page allowing a user to review a document in a workflow.

Service Class: Doc Service

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

5.2.15 UPDATE_PROBLEMREPORT

Service that updates problem report information.

Location: *IdcHomeDir*/resources/core/templates/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
```

5.2.16 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: *IdcHomeDir*/resources/core/templates/workflow.htm

5.2.17 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: *IdcHomeDir*/resources/core/templates/workflow.htm

5.2.18 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: *IdcHomeDir*/resources/core/templates/workflow.htm

5.2.19 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 in the following subsection.

Service Class: Doc Service

Location: *IdcHomeDir*/resources/core/templates/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
```

5.3 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
- DELETE_WF_TEMPLATE
- EDIT_WF_TEMPLATE
- GET_WF_TEMPLATE
- GET_WF_TEMPLATES

5.3.1 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: *IdcHomeDir*/resources/core/templates/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
```

5.3.2 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dWfTemplateName: The workflow template name.

Example

```
IdcService=DELETE_WF_TEMPLATE
dWfTemplateName=wf_template_10
```

5.3.3 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: *IdcHomeDir*/resources/core/templates/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,
    1061386026216,metaoptlists,1061386026236,wftemplates,1061386026267,projects,
    1061386026586,users,1061386026216,workflows,1061386026296,aliases,
    1061386026266,wfscripts,1061386026266,usermetaoptlists,1061386026256,
    doctypes,1061386026216,accounts,1061386026216
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,
    1061386026216,metaoptlists,1061386026236,wftemplates,1061386026267,projects,
    1061386026586,users,1061386026216,workflows,1061386026296,aliases,
    1061386026266,wfscripts,1061386026266,usermetaoptlists,1061386026256,
    doctypes,1061386026216,accounts,1061386026216
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
    \nLOCAL\ntrue\n\@end\n\@ResultSet UserView:filter\n3\nfilterColumn
```

```
    \nfilterValue\nisEnabled\ndUserAuthType\nLOCAL\ntrue\n\@end\n\@ResultSet
    DocView:filter\n3\nfilterColumn\nfilterValue\nisEnabled\ndProcessingState
    \nM\nfalse\ndIsCheckedOut\n1\nfalse\ndStatus\nGENWWW\ntrue\ndDocTitle
    \nCanceled:%\nfalse\ndReleaseState\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
```

5.3.4 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dWfTemplateName: The workflow template name.

Example

```
IdcService=GET_WF_TEMPLATE
dWfTemplateName=servicestemplate
```

5.3.5 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: *IdcHomeDir*/resources/core/templates/workflow.htm

5.4 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
- ADD_WORKFLOW_SCRIPT
- ADD_WORKFLOW_TOKEN
- ADD_WORKFLOWALIASES
- ADD_WORKFLOWDOCUMENT
- ADD_WORKFLOWDOCUMENT_SUB

- ADD_WORKFLOWDOCUMENTS
- ADD_WORKFLOWSTEP
- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- CRITERIAWORKFLOW_ENABLE
- DELETE_WFCONTRIBUTORS
- DELETE_WORKFLOW
- DELETE_WORKFLOW_SCRIPT
- DELETE_WORKFLOW_TOKEN
- DELETE_WORKFLOWCRITERIA
- DELETE_WORKFLOWDOCUMENTS
- DELETE_WORKFLOWSTEP
- EDIT_WORKFLOW
- EDIT_WORKFLOW_SCRIPT
- EDIT_WORKFLOW_TOKEN
- EDIT_WORKFLOWCRITERIA
- EDIT_WORKFLOWSTEP
- *GET_ACTIVE_WORKFLOWS
- GET_ALL_WORKFLOWDOCREVISIONS
- GET_CRITERIA_WORKFLOWS_FOR_GROUP
- GET_WF_COMPANION_INFO
- GET_WORKFLOW
- *GET_WORKFLOW_INFO
- *GET_WORKFLOW_INFO_BYNAME
- GET_WORKFLOW_SCRIPT
- *GET_WORKFLOWDOCREVISIONS
- GET_WORKFLOW_INFO
- GET_WORKFLOWS
- TEST_WORKFLOW_SCRIPT
- WORKFLOW_APPROVE
- WORKFLOW_CANCEL
- *WORKFLOW_CHECKIN
- WORKFLOW_EDIT_APPROVE
- *WORKFLOW_REJECT
- WORKFLOW_START

5.4.1 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.2 ADD_WORKFLOW_SCRIPT

Service used to add a script template for a workflow.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- wfScriptName: Name of the script template to be used.
- wfScriptDescription: Description used for the script.

5.4.3 ADD_WORKFLOW_TOKEN

Service used to add tokens for workflows.

Service Class: Workflow Service

Location: *IdcHomeDir*/resources/core/templates/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.

5.4.4 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: *IdcHomeDir*/resources/core/templates/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 system automatically assigns dWfStepID for that contribution stage.

Example

```
IdcService=ADD_WORKFLOWALIASES
dWfName=mktg_review
dWfID=12
dAliasType=user
aliases=sammy
dWfStepID=26
```

5.4.5 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.6 ADD_WORKFLOWDOCUMENT_SUB

SubService called by ADD_WORKFLOWDOCUMENTS and ADD_WORKFLOWDOCUMENT to add a content item to the Basic workflow.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

5.4.7 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.8 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: *IdcHomeDir*/resources/core/templates/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 0 (*zero*) notifies 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: `<$if ConditionExpression$>`

5.4.9 CRITERIAWORKFLOW_DISABLE

Service that disables a Criteria workflow. The most likely error is when the specified workflow does not exist.

Location: *IdcHomeDir*`/resources/core/templates/workflow.htm`

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=CRITERIAWORKFLOW_DISABLE
dWfName=mktg_review
```

5.4.10 CRITERIAWORKFLOW_DISABLE_SUB

Service that disables a sub-workflow in a Criteria workflow.

Location: *IdcHomeDir*`/resources/core/templates/workflow.htm`

5.4.11 CRITERIAWORKFLOW_ENABLE

Service that enables a Criteria workflow. The most likely error is when the specified workflow does not exist.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

dWfName: The workflow name.

Example

```
IdcService=CRITERIAWORKFLOW_ENABLE
dWfName=mktg_review
```

5.4.12 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.13 DELETE_WORKFLOW

Service that deletes an existing Basic workflow. The most likely error is when the specified workflow does not exist.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=DELETE_WORKFLOW
dWfName=mktg_review
```

5.4.14 DELETE_WORKFLOW_SCRIPT

Service that deletes a workflow script.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- wfScriptName: Name of the script template to be used.
- wfScriptDescription: Description used for the script.

5.4.15 DELETE_WORKFLOW_TOKEN

Service that deletes a workflow token.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- wfTokenName: Name of the token to be deleted.

5.4.16 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=DELETE_WORKFLOWCRITERIA
dWfName=mktg_review
```

5.4.17 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.18 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.19 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.20 EDIT_WORKFLOW_SCRIPT

Service used to edit a workflow script.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- wfScriptName: Name of the script template to be used.
- wfScriptDescription: Description used for the script.

5.4.21 EDIT_WORKFLOW_TOKEN

Service used to edit a workflow token.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- wfTokenName: Name of the token to be edited.

5.4.22 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 must meet 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 the two types of workflows, 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: *IdcHomeDir*`/resources/core/templates/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
```

5.4.23 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: *IdcHomeDir*/resources/core/templates/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 0 (*zero*) notifies 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=
```

5.4.24 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Results

- ResultSets:
  - StdWorkflows (dWfName, dWfID, dWfDescription, dCompletionDate, dSecurityGroup, dWfStatus, dWfType, 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

5.4.25 GET_ALL_WORKFLOWDOCREVISIONS

Service that returns a result set of documents in a workflow and their revision information.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

5.4.26 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 result sets:

- 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.27 GET_WF_COMPANION_INFO

Returns the companion information file for a document in a workflow.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dDocName: The name of the document in the workflow.
- dWfName: The workflow name.

5.4.28 GET_WORKFLOW

Service that returns information about a specific workflow. The most likely error is a workflow name that does not exist.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=GET_WORKFLOW
dWfName=mktg_review
```

5.4.29 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: *IdcHomeDir*/resources/core/templates/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

- ResultSets:
  - 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
```

5.4.30 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dDocName: The name of a document in an active workflow.

Results

- ResultSets:
  - 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

5.4.31 GET_WORKFLOW_SCRIPT

Service used to retrieve a workflow script.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- wfScriptName: Name of the script template to be used.
- wfScriptDescription: Description used for the script.

5.4.32 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: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Results

- ResultSets:
  - 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:
  - dUser
  - dWfName
  - isCollaboration
- Response Template:
  - WORKFLOW_DOCS (workflow_docs_list.htm)

Used By

- Resource Includes:
  - 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
```

5.4.33 GET_WORKFLOWS

Service that retrieves a list of all workflows including their description, security group, status, and type.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

5.4.34 TEST_WORKFLOW_SCRIPT

Service used to check the validity of a workflow script.

Location: *IdcHomeDir*/resources/core/templates/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.

5.4.35 WORKFLOW_APPROVE

Service that approves a content item revision in a workflow.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dID: The generated content item revision ID.

Example

```
IdcService=WORKFLOW_APPROVE
dID=81
```

5.4.36 WORKFLOW_CANCEL

Service that cancels a Basic workflow.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

Additional Required Service Parameters

- dWfName: The workflow name.

Example

```
IdcService=WORKFLOW_CANCEL
dWfName=mktg_review
```

5.4.37 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: *IdcHomeDir*/resources/core/templates/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 instance. Use the 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, you can use a metafile in its place. Only one metafile can exist though for each content item (a primary AND alternate meta file cannot coexist).
  - 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
    - dIsCheckedOut
    - 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
```

5.4.38 WORKFLOW_EDIT_APPROVE

Service used for client applications (that is, not used by the core Content Server software) which need the ability to approve a document without passing through a checkin/checkout process.

Location: *IdcHomeDir*/resources/core/templates/workflow.htm

5.4.39 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 run 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: *IdcHomeDir*/resources/core/templates/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
```

5.4.40 WORKFLOW_START

Service Class: Workflow Service

Location: *IdcHomeDir*/resources/core/templates/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
```

6

Archive Services

This chapter describes the services available when using and customizing Oracle WebCenter Content Server archives.

This chapter covers the following topics:

- About Archive Services
- Archive Services

6.1 About Archive Services

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific archive services are listed within each individual service.

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

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.

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 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.

6.2 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
- ADD_COLLECTION
- ADD_PROXIEDCOLLECTION
- CANCEL_ARCHIVE
- *CHECKIN_ARCHIVE
- COPY_ARCHIVE
- DELETE_ARCHIVE
- DELETE_BATCH_FILE
- DELETE_BATCH_FILE_DOCUMENTS
- DELETE_BATCH_FILE_TABLES
- EDIT_ARCHIVE
- EDIT_ARCHIVEDATA
- EDIT_EXPORTERS
- EDIT_TRANSFEROPTIONS
- EXECUTE_BATCH
- EXPORT_ARCHIVE
- GET_ARCHIVECOLLECTIONS
- GET_ARCHIVETABLECONTENT
- GET_ARCHIVED_FILE
- GET_ARCHIVES
- GET_ARCHIVERELATIONQUERY
- GET_BATCH_FILE_DOCUMENTS
- GET_BATCH_PROPERTIES
- GET_BATCH_SCHEMA
- GET_BATCH_VALUES
- GET_BATCHFILES
- GET_PROXIED_ARCHIVECOLLECTIONS
- GET_PROXIEDSERVERS
- GET_REPLICATION_DATA
- GET_TABLECOLUMNLIST
- GET_TARGET_INFO
- GET_TARGET_TRANSFER_STATUS
- GET_TRANSFER_SOURCE_INFO

- IMPORT_ARCHIVE
- IMPORT_ARCHIVE_START_AUTOMATED
- IMPORT_BATCHFILE
- IMPORT_DOCUMENT
- IMPORT_TABLE_ENTRY
- INSERT_NATIVE
- INSERT_NEW
- REGISTER_IMPORTER
- REMOVE_COLLECTION
- REMOVE_EXPORTER
- REMOVE_IMPORTER
- REMOVE_PROXIEDTRANSFER
- REMOVE_QUEUED_IMPORT
- REMOVE_TRANSFER
- REQUEST_TRANSFER
- TRANSFER_ARCHIVE
- UPDATE_TARGET_TOTALS
- UPDATE_TRANSFER_STATUS
- UPLOAD_ARCHIVE_TRANSFER

6.2.1 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: *IdcHomeDir*`/resources/core/tables/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 instance, the required parameters will be:

```
IdcService=ADD_ARCHIVE
IDC_Name=Master_on_server01
aArchiveName=archive_test
aArchiveDescription=this is an archive test
```

6.2.2 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection.
- aCollectionLocation: The absolute path to the collection location. Use the slash as the file separator.
- aVaultDir: The absolute path to the *vault* directory. Use the slash as the file separator.
- aWeblayoutDir: The absolute path to the *weblayout* directory. Use the slash as the file separator.

Example

```
IdcService=ADD_COLLECTION
IDC_Name=new_collection
aCollectionLocation=c:/oracle/archives
aVaultDir=c:/oracle/vault
aWeblayoutDir=c:/oracle/weblayout
```

6.2.3 ADD_PROXIEDCOLLECTION

Service used to add an archive collection from another server.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- psIDC_Name: The name of the provider (proxied connection).
- IDC_Name: The name of the archive collection.

6.2.4 CANCEL_ARCHIVE

Service that cancels the current archive request.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

6.2.5 CHECKIN_ARCHIVE

Service that checks a content item revision into an archive.

Access Level: Admin (8)

Location: *IdcHomeDir*/resources/core/tables/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 instance.
- 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 earlier.

- 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 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, you can use a metafile in its place. Only one metafile can exist though for each content item (a primary AND alternate meta file cannot coexist).
  - 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 coexist.)
  - 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
```

6.2.6 COPY_ARCHIVE

Service that copies an archive into a collection.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.7 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: *IdcHomeDir*/resources/core/tables/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
```

6.2.8 DELETE_BATCH_FILE

Service that deletes a batch file from an archive.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.9 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: *IdcHomeDir*/resources/core/tables/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
```

6.2.10 DELETE_BATCH_FILE_TABLES

Service similar to DELETE_BATCH_FILE_DOCUMENTS except it is used on exported table data.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.11 EDIT_ARCHIVE

Service used to change the description of an archive after it has been created. To access the description, double click the archive in the archive list.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.12 EDIT_ARCHIVEDATA

Service used to edit the data as specified in the EditItems parameter.

Location: *IdcHomeDir*/resources/core/tables/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-delimited 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
```

6.2.13 EDIT_EXPORTERS

Service that edits the archive data for the specified archive. If specified, it registers or unregisters the archive.

Location: *IdcHomeDir*/resources/core/tables/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

6.2.14 EDIT_TRANSFEROPTIONS

Service used to edit transfer options for an archive process.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.15 EXECUTE_BATCH

Service that executes a batch operation for an archive.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- aBatchFile: The path and name of the batch file to be executed.

6.2.16 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 users or document configuration information should be exported (or both), and whether previous batch files should be deleted.
- The Content Server executes this service asynchronously. The server must 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. 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: *IdcHomeDir*/resources/core/tables/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
```

6.2.17 GET_ARCHIVECOLLECTIONS

Service that returns a list of all archive collections.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

6.2.18 GET_ARCHIVETABLECONTENT

Service used to preview the table data to be exported. It is used in the **Archiver/Export Data/Table/Preview** dialog.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.19 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 system assume that it is present, as do other features such as forms.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.20 GET_ARCHIVES

Service that returns a list of all archives in a collection.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.21 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: *IdcHomeDir*/resources/core/tables/std_services.htm

6.2.22 GET_BATCH_FILE_DOCUMENTS

Service that returns all batch file content items.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.23 GET_BATCH_PROPERTIES

Service that returns the properties of the specified batch file.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.24 GET_BATCH_SCHEMA

Service that returns the batch schema. Schema information describes the classes of objects that are stored in the database.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.25 GET_BATCH_VALUES

Service that returns the metadata field values for the batch.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.26 GET_BATCHFILES

Service that returns batch files for a specified archive.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.27 GET_PROXIED_ARCHIVECOLLECTIONS

Service that returns a list of all archive collections on a proxied Content Server instance.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- psIDC_Name: The name of the proxied Content Server instance.

Example

```
IdcService=GET_PROXIED_ARCHIVECOLLECTIONS
psIDC_Name=Proxied_2_on_test13
```

6.2.28 GET_PROXIEDSERVERS

Service that returns a list of outgoing providers.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

6.2.29 GET_REPLICATION_DATA

Service that returns replication data. The most likely error is an incorrect archive location.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.30 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: *IdcHomeDir*/resources/core/tables/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-delimited table names.

6.2.31 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: *IdcHomeDir*/resources/core/tables/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.

6.2.32 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: *IdcHomeDir*/resources/core/tables/std_services.htm

6.2.33 GET_TRANSFER_SOURCE_INFO

Service that is used during a pull transfer to establish the work that must be performed. Used when the source Archiver for the transfer resides on a proxied server and the transfer owner must determine what work, if any, must be done.

Not intended for external use.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.34 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 system executes this service asynchronously. The Content Server instance must 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. 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: *IdcHomeDir*/resources/core/tables/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 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
```

6.2.35 IMPORT_ARCHIVE_START_AUTOMATED

Service that registers a queued archive for import.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.36 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 instance.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.37 IMPORT_DOCUMENT

Service that imports a specified document, usually selected from the **View Batch File** dialog, into the Content Server instance.

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

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.38 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: *IdcHomeDir*/resources/core/tables/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.

6.2.39 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: *IdcHomeDir*/resources/core/tables/std_services.htm

To disallow the check in of empty files, set the following:
`ValidatePrimaryFileNotEmpty=1`

6.2.40 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 and during an archive import operation. It adds a new content item into the system and creates new IDs (such as dID, dRevClassID) for it.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

To disallow the check in of empty files, set the following:
`ValidatePrimaryFileNotEmpty=1`

6.2.41 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: *IdcHomeDir*/resources/core/tables/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 instance as importer:

```
IdcService=REGISTER_IMPORTER
IDC_Name=Master_on_server01
aArchiveName=JAN_22_02
IsRegister=1
EditItems=aRegisteredImporter,aImportLogonUser
```

6.2.42 REMOVE_COLLECTION

Service that removes an archive collection. You cannot remove the default archive collection.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- IDC_Name: The name of the collection.

Example

```
IdcService=REMOVE_COLLECTION
IDC_Name=new_collection
```

6.2.43 REMOVE_EXPORTER

Service that removes an exporter from an archive. The most likely error is an incorrect archive location.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.44 REMOVE_IMPORTER

Service that removes an importer from an archive. The most likely error is an incorrect archive location.

Location: *IdcHomeDir*/resources/core/tables/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
```

6.2.45 REMOVE_PROXIEDTRANSFER

Service that is called when removing the transfer settings for an archive but the Content Server instance performing the activity is not the Content Server instance 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: *IdcHomeDir*/resources/core/tables/std_services.htm

Additional Required Service Parameters

- aArchiveLocation: Location of the archive.
- IDC_Name: The name of the owner of the transfer.

6.2.46 REMOVE_QUEUED_IMPORT

Service that allows a user to delete a queued import. It is used from the Automation dialog, accessible by clicking **Archiver Options** then **View Automation** then Queued Automated Imports.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.47 REMOVE_TRANSFER

Service that allows a user to delete or stop an automated transfer. It is used from the Automation dialog, accessible by clicking **Archiver Options** then **View Automation**.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.48 REQUEST_TRANSFER

Service that initiates a transfer request.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.49 TRANSFER_ARCHIVE

Service that initiates a specified manual transfer.

Location: *IdcHomeDir*/resources/core/tables/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.

6.2.50 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 and updating the data on completion of any manual or automated transfers.

Location: *IdcHomeDir*/resources/core/tables/std_services.htm

6.2.51 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: *IdcHomeDir*/resources/core/tables/std_services.htm

6.2.52 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: *IdcHomeDir*/resources/core/tables/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 path name to the zip file of the transfer to be uploaded.

7

Contribution Folders Services

This chapter describes the Oracle WebCenter Content services available when using and customizing Contribution Folders services.

This chapter covers the following topics:

- About Contribution Folders Services
- Contribution Folders Services

7.1 About Contribution Folders Services

Contribution Folders is the name given to what in earlier releases of WebCenter Content was named the Folders_g feature. Contribution Folders is supported by the Folders_g component in WebCenter Content. Contribution Folders services **cannot** be used with the current Folders feature (see Chapter 8).

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations and service types for specific Contribution Folders services are listed within each individual service.

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

7.2 Contribution Folders Services

The following services are described in this section:

- COLLECTION_ADD
- COLLECTION_ADD_LINK
- COLLECTION_BROWSE
- COLLECTION_CHECKIN_NEW
- COLLECTION_CHECKIN_REVISION
- COLLECTION_CHECKIN_SEL_SUB
- COLLECTION_COPY_ALL
- COLLECTION_COPY_COLLECTION

- COLLECTION_COPY_ITEM
- COLLECTION_COPY_LOT
- COLLECTION_DELETE
- COLLECTION_DELETE_ALL
- COLLECTION_DELETE_COLLECTION
- COLLECTION_DELETE_ITEM
- COLLECTION_DELETE_LOT
- COLLECTION_DISPLAY
- COLLECTION_EDIT
- COLLECTION_GET_ADMIN_CONFIG
- COLLECTION_GET_ADMIN_INHERIT_CONFIG
- COLLECTION_GET_ADMIN_MARKED_CONFIG
- COLLECTION_GET_ADMIN_META_CONFIG
- COLLECTION_GET_ADMIN_METADATA_DEFAULTS
- COLLECTION_GET_ADMIN_MOUNTED_CONFIG
- COLLECTION_GET_ARCHIVE
- COLLECTION_GET_BRANCH
- COLLECTION_GET_COLLECTION
- COLLECTION_GET_COLLECTIONS
- COLLECTION_GET_CONTENT_FILE
- COLLECTION_GET_CONTENTS
- COLLECTION_GET_FILE
- COLLECTION_GET_INFO
- COLLECTION_GET_LINKS
- COLLECTION_GET_META_MAPPING
- COLLECTION_GET_PROFILE_METADATA_DEFAULTS
- COLLECTION_GET_PROFILE_METADATA_REVISION_DEFAULTS
- COLLECTION_GET_REFERENCE
- COLLECTION_GET_SEARCH_FORM
- COLLECTION_GET_SYSTEM_FILE
- COLLECTION_GET_USER_CONFIG
- COLLECTION_INFO
- COLLECTION_ISVALID_META
- COLLECTION_LOCK
- COLLECTION_MOVE_ALL
- COLLECTION_MOVE_COLLECTION
- COLLECTION_MOVE_ITEM

- COLLECTION_MOVE_LOT
- COLLECTION_NEW
- COLLECTION_PROFILE_UPDATE_COLUMNS
- COLLECTION_RESTORE_COLLECTION
- COLLECTION_RESTORE_ITEM
- COLLECTION_SEARCH_CONTENT
- COLLECTION_SEARCH_RESULTS
- COLLECTION_SET_ARCHIVE
- COLLECTION_SET_USER_CONFIG
- COLLECTION_UNLOCK
- COLLECTION_UPDATE
- COLLECTION_UPDATE_ADMIN_CONFIG
- COLLECTION_UPDATE_ADMIN_INHERIT_CONFIG
- COLLECTION_UPDATE_ADMIN_METADATA_DEFAULTS
- COLLECTION_UPDATE_ALL
- COLLECTION_UPDATE_ITEM
- COLLECTION_UPDATE_META
- COLLECTION_UPDATE_META_TABLE
- COLLECTION_UPDATE_PROFILE_METADATA_DEFAULTS
- COLLECTION_UPDATE_PROFILE_METADATA_REVISION_DEFAULTS
- COLLECTION_UPDATE_STRUCTURE
- COLLECTION_VERIFY_FOLDER_NAME
- FOLDERSLOCAL_BUILD_MOUNT
- FOLDERSLOCAL_CREATE_MOUNT
- FOLDERSLOCAL_DELETE_MOUNT
- FOLDERSLOCAL_UPDATE_MOUNT
- GET_FOLDER_HISTORY_REPORT
- GET_OPTION_LISTS
- GOTO_COLLECTION
- GOTO_ROOT_COLLECTION

7.2.1 COLLECTION_ADD

Service that creates a Contribution Folder. The most likely errors are mismatched parameters or the contribution folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*`/components/folders_g/resources/folders_service.htm`

Additional Required Service Parameters

- dCollectionName: The name of the Contribution Folder to create.

One of the following sets of parameters to identify the parent Contribution Folder:

- hasParentCollectionID: (Boolean) Specifies whether the dParentCollectionID parameter is used to identify the parent Contribution Folder ID.

  dParentCollectionID: The parent Contribution Folder ID.

- hasParentCollectionPath: (Boolean) Specifies whether the dParentCollectionPath parameter is used to identify the path to the parent Contribution Folder.

  dParentCollectionPath: The path to the parent Contribution Folder.

- hasParentCollection GUID: (Boolean) Specifies that the dParentCollectionGUID parameter is used to identify the GUID for the parent Contribution Folder.

  dParentCollectionGUID: The parent Contribution Folder GUID.

Optional Service Parameters

- dCollectionOwner: The Contribution Folder owner. If xForceFolderSecurity=TRUE, then the Contribution Folder owner is the same as the parent Contribution Folder's owner. Otherwise, the default is the current user.

  This setting overrides the user set as the collection owner. The owner can access and modify the Contribution Folder despite other set security criteria.

- dCollectionCreator: Overrides the user set as the Contribution Folder creator. The default value is the current user.
- dCollectionModifier: Overrides the user set as the last Contribution Folder modifier. The default value is the current user.
- ignoreMaxFolderLimit: Allows more subfolders to be added to the parent Contribution Folder than the currently set maximum limit. The default value is *false*.
- mark: Sets dCollectionMark for the Contribution Folder. Contribution Folders with dCollectionMark set are not modifiable without CollectionReadOnlyMarkedFolders being set to *false* (not the default case). These marked Contribution Folders and their enabled/disabled state can be viewed on the System Folder Configuration page. Contribution Folders and Trash are marked folders.
- force: Creates a Contribution Folder under the root Contribution Folder (dParentCollectionID=-1). The default value is *false*.
- *any system or custom metadata field*: The given field and value will be set for the Contribution Folder and used as default values for content and Contribution Folders created within the Contribution Folder.

Example

```
IdcService=COLLECTION_ADD
hasParentCollectionID=true
dCollectionName=Products
dParentCollectionID=1
dCollectionOwner=bsmith
```

7.2.2 COLLECTION_ADD_LINK

Service that adds a shortcut to a Contribution Folder or content within a Contribution Folder. The shortcut references the Contribution Folder using either the Contribution Folder ID, the actual path to the Contribution Folder, or the Contribution Folder GUID.

The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_services.htm

Additional Required Service Parameters

- One of the following sets of parameters to identify destination parent Contribution Folder for the created short cut:
  - tohasCollectionID: (Boolean) Specifies whether the todCollectionID parameter is used to identify the destination parent Contribution Folder ID.

    todCollectionID: The Contribution Folder ID that the shortcut references.
  - tohasCollectionPath: (Boolean) Specifies whether the todCollectionPath parameter is used to identify the path to the Contribution Folder.

    todCollectionPath: The path to the Contribution Folder that the shortcut references.
  - tohasCollectionGUID: (Boolean) Specifies whether the todCollectionGUID parameter is used to identify the Contribution Folder.

    todCollectionGUID: The GUID for the Contribution Folder that the shortcut references.
- dLinkName: The name of the shortcut being created.
- dLinkType: The type of shortcut being created. Must be either COLLECTION or CONTENT.
  - If COLLECTION is specified, then hasdCollectionID must be specified.

    hasdCollectionID: (Boolean) Specifies whether the Contribution Folder is referenced using a Collection ID.

    dCollectionID: The Contribution Folder ID of the parent Contribution Folder referenced by the shortcut.
  - If CONTENT is specified, then dRevClassID must be specified.

    dRevClassID: The revision class ID for the content item to which the shortcut points.

Example

```
IdcService=COLLECTION_ADD_LINK
tohasCollectionID=true
todCollectionID=999999999999000633
dLinkName=Shortcut
dLinkType=COLLECTION
hasdCollectionID=true
dCollectionID=999999999999004533
```

7.2.3 COLLECTION_BROWSE

Service that loads the defined Contribution Folder metadata and the Contribution Folder path. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_interface_service.htm

Additional Required Service Parameters

- dCollectionID: The Contribution Folder ID of the parent Contribution Folder referenced by the shortcut (for example, ="+id+").
- hasCollectionID: (Boolean) Specifies whether the Contribution Folder has an assigned collection ID.
- changeToUser: Reassigns the Contribution Folder to a different user.

Example

```
IdcService=COLLECTION_BROWSE
dCollectionID="+id+"
hasCollectionID=true
changeToUser=bsmith
```

7.2.4 COLLECTION_CHECKIN_NEW

Service that checks new content into a Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.5 COLLECTION_CHECKIN_REVISION

Service that checks in a revision to an existing Contribution 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 Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folder/resources/folders_service.htm

7.2.6 COLLECTION_CHECKIN_SEL_SUB

Service that checks in a revision to an existing Contribution Folder under certain conditions. Used when the Contribution 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 Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.7 COLLECTION_COPY_ALL

Service that copies all content residing in the specified Contribution Folder to the destination Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.8 COLLECTION_COPY_COLLECTION

Service that copies a specified Contribution Folder and places the copy into a different Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following sets of parameters to identify the destination parent Contribution Folder for the copied Contribution Folder:

- tohasCollectionID: (Boolean) Specifies whether the todCollectionID parameter is used to identify the destination parent Contribution Folder ID.

  todCollectionID: The destination parent Contribution Folder ID.

- tohasCollectionPath: (Boolean) Specifies whether the todCollectionPath parameter is used to identify the path to the destination parent Contribution Folder.

  todCollectionPath: The path to the destination parent Contribution Folder.

- tohasCollection GUID: (Boolean) Specifies whether the todCollectionGUID parameter is used to identify the destination parent Contribution Folder.

  todCollectionGUID: The destination parent Contribution Folder GUID.

One of the following sets of parameters to identify the source Contribution Folder to be copied:

- fromhasCollectionID: (Boolean) Specifies whether the fromdCollectionID parameter is used to identify the source Contribution Folder.

  fromdCollectionID: The source Contribution Folder ID.

- fromhasCollectionPath: (Boolean) Specifies whether the fromdCollectionPath parameter is used to identify the path to the source Contribution Folder.

fromdCollectionPath: The path to the source Contribution Folder.

- fromhasCollectionGUID: (Boolean) Specifies whether the fromdCollectionGUID parameter is used to identify the source Contribution Folder.

  fromdCollectionGUID: The source Contribution Folder GUID.

- fromisLink: (Boolean) Specifies whether the fromdLinkID parameter is used to identify the source Contribution Folder.

  fromdLinkID: The Contribution Folder shortcut ID for the source Contribution Folder

- fromCollectionislink: (Boolean) Specifies whether the fromdCollectionLinkID parameter is used to identify the source Contribution Folder.

  fromdCollectionLinkID: The collection shortcut ID for the source Contribution Folder.

Example

```
IdcService=COLLECTION_COPY_COLLECTION
tohasCollectionID=true
todCollectionID=999999999999000633
fromhasCollectionID=true
fromdCollectionID=999999999999000633
```

7.2.9 COLLECTION_COPY_ITEM

Service that copies a single content item in a Contribution Folder and places it in a destination Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: File Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

- dDocName: The content ID of the content item being copied.

One of the following sets of parameters to identify the destination Contribution Folder:

- tohasCollectionID: (Boolean) Specifies whether the todCollectionID parameter is used to identify the destination Contribution Folder.

  todCollectionID: The ID for the destination Contribution Folder.

- tohasCollectionPath: (Boolean) Specifies whether the todCollectionPath parameter is used to identify the destination Contribution Folder.

  todCollectionPath: The path to the destination Contribution Folder.

- tohasCollectionGUID: (Boolean) Specifies whether the todCollectionGUID parameter is used to identify the destination Contribution Folder.

  todCollectionGUID: The GUID for the destination Contribution Folder.

- tolevel0: Specifies the base value in the path for the destination parent Contribution Folder (for example, `/Contribution Folders`).

  tolevel1: Specifies the next level value in the path for the destination parent Contribution Folder (for example, `/t4`).

tolevel2: Specifies the next level value in the path for the destination parent Contribution Folder (for example, `/1`).

Example

```
IdcService=COLLECTION_COPY_ITEM
tohasCollectionPath=true
todCollectionPath=/Contribution Folders/t4/1
dDocName=000660
```

7.2.10 COLLECTION_COPY_LOT

Service that copies a set of items including content items, Contribution Folders, shortcuts to content items, and shortcuts to Contribution Folders. The copied items are placed into a single designated Contribution Folder.

The most likely errors are mismatched parameters or the designated Contribution Folder does not exist. No returned data.

Service Class: Service (general service)

Location: *IdcHomeDir*`/components/folders_g/resources/folders_service.htm`

Additional Required Service Parameters

One of the following sets of parameters to identify the destination parent Contribution Folder for the copied items:

- tohasCollectionID: (Boolean) Specifies whether the todCollectionID parameter is used to identify the destination parent Contribution Folder.

  todCollectionID: The destination parent Contribution Folder ID.

- tohasCollectionPath: (Boolean) Specifies whether the todCollectionPath parameter is used to identify the destination parent Contribution Folder.

  todCollectionPath: The path to the destination parent Contribution Folder.

- tohasCollection GUID: (Boolean) Specifies whether the todCollectionGUID parameter is used to identify the destination parent Contribution Folder.

  todCollectionGUID: The destination parent Contribution Folder GUID.

Parameters to designate content items (where {*n*} designates an integer used to distinguish parameters particular to one item to copy from other items to copy):

- contentselect{*n*}: (Boolean) Specifies whether to copy the content item. The value must be *true* to copy the content item.
- fromContentisLink{*n*}: (Boolean) Specifies whether this is a content item or a shortcut. The value must be `0` to specify a content item.
- fromdDocName{*n*}: The content ID of the content item to be copied.

Parameters to designate shortcuts to content items (where {*n*} designates an integer used to distinguish parameters particular to one item to copy from other items to copy):

- contentselect{*n*}: (Boolean) Specifies whether to copy the content item. The value must be *true* to copy the content item.
- fromContentisLink{*n*}: (Boolean) Specifies whether this is a content item or a shortcut. The value must be `1` to specify a shortcut.
- fromdDocName{*n*}: The content ID of the content item pointed to by the shortcut.

Parameters to designate Contribution Folders (where {*n*} designates an integer used to distinguish parameters particular to one item to copy from other items to copy):

- collectionselect{*n*}: (Boolean) Specifies whether to copy the Contribution Folder. The value must be *true* to copy the Contribution Folder.
- fromCollectionisLink{*n*}: (Boolean) Specifies whether this is a Contribution Folder or a shortcut. The value must be `0` to specify a Contribution Folder.
- fromdCollectionID{*n*}: The dCollectionID of the Contribution Folder being copied.
- fromhasCollectionID{*n*}: The value must be `1` to copy the Contribution Folder.

Parameters to designate shortcuts to Contribution Folders (where {*n*} designates an integer used to distinguish parameters particular to one item to copy from other items to copy):

- collectionselect{*n*}: (Boolean) Specifies whether to copy the shortcut to the Contribution Folder. The value must be *true* to copy the shortcut.
- fromCollectionisLink{*n*}: (Boolean) Specifies whether this is a Contribution Folder or a shortcut. The value must be `1` to specify a shortcut.
- fromCollectiondLinkID{*n*}: The link ID of the shortcut to be copied.
- fromdCollectionID{*n*}: The dCollectionID of the Contribution Folder pointed to by the shortcut.
- fromhasCollectionID{*n*}: The value must be `1` to copy the shortcut to the Contribution Folder.

Example

```
IdcService=COLLECTION_COPY_LOT
tohasdCollectionID=true
todCollectionID=999999999999000633
contentselect=true
fromContentisLink=0
fromdDocName=PHL11GX010033
```

7.2.11 COLLECTION_DELETE

Service that deletes Contribution Folders. If Trash is enabled, the Contribution Folder is moved to Trash and not directly deleted.

The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Doc Service

Location: *IdcHomeDir*`/components/folders_g/resources/folders_service.htm`

Additional Required Service Parameters

One of the following parameter sets is required to identify the Contribution Folder.

- hasCollectionID: (Boolean) Specifies that the dCollectionID parameter is used to identify the Contribution Folder.

  dCollectionID: The Contribution Folder ID of the Contribution Folder to delete.
- hasCollectionPath: (Boolean) Specifies that the dCollectionPath parameter is used to identify the Contribution Folder.

  dCollectionPath: The path of the Contribution Folder to delete.

- hasCollectionGUID: (Boolean) Specifies that the dCollectionGUID parameter is used to identify the Contribution Folder.

  dCollectionGUID: The GUID of the Contribution Folder to delete.

Example

```
IdcService=COLLECTION_DELETE
hasCollectionPath=true
dCollectionPath=/Contribution Folders/dept/hr
```

7.2.12 COLLECTION_DELETE_ALL

Service that deletes all content residing in a Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.13 COLLECTION_DELETE_COLLECTION

Service that deletes a specified Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.14 COLLECTION_DELETE_ITEM

Service that deletes a single content item in a Contribution Folder. If Trash is enabled, the item is moved to the Trash Contribution Folder and not directly deleted. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

- dDocName: The content ID of the content item to be deleted.

Example

```
IdcService=COLLECTION_DELETE_ITEM
dDocName=000660
```

7.2.15 COLLECTION_DELETE_LOT

Service that deletes a set of items including content items, Contribution Folders, shortcuts to content items, and shortcuts to Contribution Folders. If Trash is enabled, the items are moved to Trash and not directly deleted.

The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

Parameters to designate content items (where {*n*} designates an integer used to distinguish parameters particular to one item to delete from other items to delete):

- contentselect{*n*}: (Boolean) Specifies whether to delete the content item. The value must be *true* to copy the content item.
- fromContentisLink{*n*}: (Boolean) Specifies whether this is a content item or a shortcut. The value must be `0` to specify a content item.
- fromdDocName{*n*}: The content ID of the content item to delete.

Parameters to designate shortcuts to content items (where {*n*} designates an integer used to distinguish parameters particular to one item to delete from other items to delete):

- contentselect{*n*}: (Boolean) Specifies whether to delete the shortcut. The value must be *true* to delete the shortcut.
- fromContentisLink{*n*}: (Boolean) Specifies whether this is a content item or a shortcut. The value must be `1` to specify a shortcut.
- fromContentdLinkID{*n*}: The link ID of the shortcut.
- fromdDocName{*n*}: The content ID of the content item pointed to by the shortcut.

Parameters to designate Contribution Folders (where {*n*} designates an integer used to distinguish parameters particular to one item to delete from other items to delete):

- collectionselect{*n*}: (Boolean) Specifies whether to delete the Contribution Folder. The value must be *true* to delete the Contribution Folder.
- fromCollectionisLink{*n*}: (Boolean) Specifies whether this is a Contribution Folder or a shortcut. The value must be `0` to specify a Contribution Folder.
- fromdCollectionID{*n*}: The dCollectionID of the Contribution Folder to delete.
- fromhasCollectionID{*n*}: The value must be `1` to delete the Contribution Folder.

Parameters to designate shortcuts to Contribution Folders (where {*n*} designates an integer used to distinguish parameters particular to one item to delete from other items to delete):

- collectionselect{*n*}: (Boolean) Specifies whether to delete the shortcut to the Contribution Folder. The value must be *true* to delete the shortcut.
- fromCollectionisLink{*n*}: (Boolean) Specifies whether this is a Contribution Folder or a shortcut. The value must be `1` to specify a shortcut.
- fromCollectiondLinkID{*n*}: The link ID of the shortcut to delete.
- fromdCollectionID{*n*}: The dCollectionID of the Contribution Folder pointed to by the shortcut.
- fromhasCollectionID{*n*}: The value must be `1` to delete the shortcut to the Contribution Folder.

Example

```
IdcService=COLLECTION_DELETE_LOT
collectionselect=true
fromCollectionisLink=0
fromdCollectionID=999999999999004955
fromhasCollectionID=1
```

7.2.16 COLLECTION_DISPLAY

Service that returns a list of the items contained in a Contribution Folder, including content items, Contribution Folders and short cuts. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_interface_service.htm

Additional Required Service Parameters

One of the following parameter sets is required to identify the Contribution Folder.

- hasCollectionID: (Boolean) Specifies that the dCollectionID parameter is used to identify the Contribution Folder.

  dCollectionID: The collection ID of the Contribution Folder.

- hasCollectionPath: (Boolean) Specifies that the dCollectionPath parameter is used to identify the Contribution Folder.

  dCollectionPath: The path of the Contribution Folder.

- hasCollectionGUID: (Boolean) Specifies that the dCollectionIGUID parameter is used to identify the Contribution Folder.

  dCollectionGUID: The collection GUID for the Contribution Folder.

Optional Service Parameters

- CollectionDisplayResultSetSize: Limits the number of items returned in the COLLECTIONS and CONTENTS ResultSets and establishes the page size.
- showCollections: Flag that signals the web UI presentation to display the COLLECTIONS ResultSet. This is effective when using CollectionDisplayResultSetSize and PageNumber.
- showContent: Flag that signals the web UI presentation to display the CONTENTS ResultSet. This is effective when using CollectionDisplayResultSetSize and PageNumber.
- PageNumber: Integer value 1 or greater displays the correct page size subset of data from the CONTENTS or COLLECTIONS ResultSets. Page size is set by CollectionDisplayResultSetSize.
- CollectionReleasedOnly:
  - 0 (default): If you have access to a folder, you can view the documents that are checked into the folder, that is, both released and not released documents.
  - 1: If you have access to a folder, you can view the released documents or the documents for which you are the author. Pass the *CollectionReleasedOnlyIgnoreAuthor* flag along with CollectionReleasedOnly = 1 to display only the released content.

Results

- ResultSets: (contain a row with Contribution Folder metadata for each Contribution Folder item directly contained by the Contribution Folder, including shortcuts to Contribution Folders)
  - CONTENTS
  - COLLECTIONS

- METAMAPPING
- COLUMNS
- Fields
- METADATA_OVERRIDE
- PATH
- metadata

Example

```
IdcService=COLLECTION_DISPLAY
hasCollectionPath=true
dCollectionPath=%2fContent%20Server%20Folders%2f
```

7.2.17 COLLECTION_EDIT

Service that edits a Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service type: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

- hasCollectionID: (Boolean) Specifies whether the Contribution Folder has an assigned collection ID. If set to *true*, dCollectionID must be defined.
- dCollectionID: The Contribution Folder ID of the parent Contribution Folder.

7.2.18 COLLECTION_GET_ADMIN_CONFIG

Service that retrieves the current global values defined for the Contribution Folders hierarchy. No specific collection is passed. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.19 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 Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.20 COLLECTION_GET_ADMIN_MARKED_CONFIG

Service that retrieves 'special' Contribution Folders and their disabled/enabled status. A disabled Contribution Folder does not show up in the hierarchy. An enabled Contribution Folder is accessible. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.21 COLLECTION_GET_ADMIN_META_CONFIG

Service that retrieves the fields from the administration metadata ResultSet for the defined Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.22 COLLECTION_GET_ADMIN_METADATA_DEFAULTS

Service that retrieves the option lists for custom fields, default metadata, and the administration metadata of the Contribution Folder configuration (retrieves field defaults). The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.23 COLLECTION_GET_ADMIN_MOUNTED_CONFIG

Service that retrieves the Contribution Folders system configuration for the local Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_mounted_service.htm

7.2.24 COLLECTION_GET_ARCHIVE

Service that downloads the archived Contribution Folder structure of the specified Contribution Folder. This service does not download the contents of a Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: File Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.25 COLLECTION_GET_BRANCH

Service that retrieves the structure of the specified Contribution Folder and its subfolders. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.26 COLLECTION_GET_COLLECTION

Service that retrieves all the subfolders of the specified Contribution Folder. Compare COLLECTION_GET_COLLECTIONS. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.27 COLLECTION_GET_COLLECTIONS

Service that retrieves the Contribution Folder. Compare COLLECTION_GET_ COLLECTION. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Example

To retrieve a listing of hierarchical collections, the required parameters are:

```
hasCollectionID=1
dCollectionID=collectionnumber
```

7.2.28 COLLECTION_GET_CONTENT_FILE

Service that downloads the first content item of the specified Contribution Folders collection that matches the metadata of the item passed in (excluding revisions). The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: File Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.29 COLLECTION_GET_CONTENTS

Service that retrieves information for the content items contained directly in the specified Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following parameter sets is required to identify the Contribution Folder.

- hasCollectionID: (Boolean) Specifies that the dCollectionID parameter is used to identify the Contribution Folder.

  dCollectionID: The Contribution Folder ID of the Contribution Folder.

- hasCollectionPath: (Boolean) Specifies that the dCollectionPath parameter is used to identify the Contribution Folder.

  dCollectionPath: The path of the Contribution Folder.

- hasCollectionGUID: (Boolean) Specifies that the dCollectionGUID parameter is used to identify the Contribution Folder.

  dCollectionGUID: The GUID of the Contribution Folder.

Results

- ResultSets:
  - CONTENTS (contains a row with content metadata for each content item contained in the Contribution Folder)

Example

```
IdcService=COLLECTION_GET_CONTENTS
hasCollectionID=true
dCollectionID=999999999999000633
```

7.2.30 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 Contribution Folder does not exist.

Service Class: File Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.31 COLLECTION_GET_INFO

Service that retrieves Contribution folder information for the local Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following parameter sets is required to identify the Contribution Folder.

- hasCollectionID: (Boolean) Specifies whether the dCollectionID parameter is used to identify the Contribution Folder.

  dCollectionID: The Contribution Folder ID of the Contribution Folder.

- hasCollectionPath: (Boolean) Specifies whether the dCollectionPath parameter is used to identify the Contribution Folder.

  dCollectionPath: The path of the Contribution Folder.

- hasCollectionGUID: (Boolean) Specifies that the dCollectionGUID parameter is used to identify the Contribution Folder.

  dCollectionGUID: The GUID of the Contribution Folder.

Results

- ResultSets:
  - PATH (contains a row with Contribution Folder metadata for each Contribution Folder in the Contribution Folder's path)

Example

```
IdcService=COLLECTION_GET_INFO
hasCollectionID=true
dCollectionID=999999999999000633
```

7.2.32 COLLECTION_GET_LINKS

Service that retrieves information about the specified shortcut (dLinkID) to a content item or Contribution Folder. The most likely errors are mismatched parameters or the item or Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

- dLinkID: The link ID for a shortcut to a content item or Contribution Folder.

Optional Service Parameters

- resultSetName: Sets the name of the returned ResultSet. The default value is LINKS.

Results

- ResultSets:
  - $*resultSetName*$ (the ResultSet containing shortcut information for a content item or a Contribution Folder)

7.2.33 COLLECTION_GET_META_MAPPING

Service used to retrieve the WebDAV equivalent properties of the specified Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.34 COLLECTION_GET_PROFILE_METADATA_DEFAULTS

Service used to retrieve the Default Information Field Configuration for a specific user. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.35 COLLECTION_GET_PROFILE_METADATA_REVISION_DEFAULTS

Service that retrieves the Revision Information Field Configuration values for a specific user. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.36 COLLECTION_GET_REFERENCE

Service used to determine the existence of an item within the Contribution Folder hierarchy, and if the item exists, whether it is a Contribution Folder, content item, or a shortcut. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following parameter sets is required to identify the path.

- hasCollectionPath: (Boolean) Specifies whether the dCollectionPath parameter is used to identify the parent Contribution Folder.

  dCollectionPath: The path of the parent Contribution Folder.

- level0: The primary level of the parent Contribution Folder.

  level1: The secondary level of the parent Contribution Folder.

  level2: The content item within the Contribution Folder.

Optional Service Parameters

- RevisionHistory: (Boolean) Specifies whether to include the REVISION_HISTORY result set if a content item is selected. The default value is *false*.

Results

- isCollection: (Boolean) Is *true* if the referenced path represents a Contribution Folder.
- isContent: (Boolean) Is *true* if the referenced path represents a content item. If both isCollection and isContent are *false*, then no item exists with the given path.
- isLink: (Boolean) Is *true* if the referenced path represents a shortcut. If missing, the value is assumed *false*.
- ResultSets:
  - REVISION_HISTORY (only included if the RevisionHistory and isContent parameters are *true*)

Partial item information is returned for the referenced item in the binder's local data (Doc Info or Contribution Folder Info).

7.2.37 COLLECTION_GET_SEARCH_FORM

Service that retrieves the Contribution Folder specific search form. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Search Type: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.38 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 Contribution Folder does not exist.

Service Class: File Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.39 COLLECTION_GET_USER_CONFIG

Service that retrieves user specific configuration for the display and behavior of their Contribution Folder structure. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location; *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.40 COLLECTION_INFO

Service that retrieves the metadata associated with this particular Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following sets of parameters to identify the Contribution Folder:

- hasCollectionID: (Boolean) Specifies whether the dCollectionID parameter is used to identify the Contribution Folder.

  dCollectionID: The Contribution Folder ID of the Contribution Folder referenced by the shortcut.

- hasCollectionPath: (Boolean) Specifies whether the dCollectionPath parameter is used to identify the Contribution Folder.

  dCollectionPath: The path of the Contribution Folder.

- hasCollectionGUID: (Boolean) Specifies whether the dCollectionGUID parameter is used to identify the Contribution Folder.

  dCollectionGUID: The GUID of the Contribution Folder.

Results

- canReadCollection: (Boolean) Is *true* if the requesting user has read privilege for the Contribution Folder.
- canWriteCollection: (Boolean) Is *true* if the requesting user has write privilege for the Contribution Folder.
- canAdminCollection: (Boolean) Is *true* if the requesting user has admin privilege for the Contribution Folder.
- ResultSets:
  - PATH (contains a row with Contribution Folder information for every Contribution Folder in the Contribution Folder's path)
  - DocFormats
  - ClbraProjectsAccessLists
  - DocTypes
  - metadata

7.2.41 COLLECTION_ISVALID_META

Service that verifies that the Contribution Folder can be added by the item of the specified metadata. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.42 COLLECTION_LOCK

Service that checks out a Contribution Folder by name (dDocName) and locks the Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.43 COLLECTION_MOVE_ALL

Service that moves all of the content in a specified Contribution Folder into a different Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following sets of parameters to identify the current source Contribution Folder of items to move:

- fromhasCollectionID: (Boolean) Specifies whether the fromdCollectionID parameter is used to identify the source Contribution Folder.

  fromdCollectionID: The source Contribution Folder ID.

- fromhasCollectionPath: (Boolean) Specifies whether the fromdCollectionPath parameter is used to identify the source Contribution Folder.

  fromdCollectionPath: The path to the source Contribution Folder.

- fromhasCollectionGUID: (Boolean) Specifies whether the fromdCollectionGUID parameter is used to identify the source Contribution Folder.

  fromdCollectionGUID: The source Contribution Folder GUID.

One of the following sets of parameters to identify the destination Contribution Folder:

- tohasCollectionID: (Boolean) Specifies whether the todCollectionID parameter is used to identify the destination parent Contribution Folder.

  todCollectionID: The ID for the destination parent Contribution Folder.

- tohasCollectionPath: (Boolean) Specifies whether the todCollectionPath parameter is used to identify the destination parent Contribution Folder.

  todCollectionPath: The path to the destination parent Contribution Folder.

- tohasCollectionGUID: (Boolean) Specifies whether the todCollectionGUID parameter is used to identify the destination parent Contribution Folder.

  todCollectionGUID: The GUID for the destination parent Contribution Folder.

- tolevel0: Specifies the base value in the path for the destination parent Contribution Folder (for example, `/Contribution Folders`).

  tolevel1: Specifies the next level value in the path for the destination parent Contribution Folder (for example, `/t4`).

  tolevel2: Specifies the next level value in the path for the destination parent Contribution Folder (for example, `/1`).

Example

```
IdcService=COLLECTION_MOVE_ALL
fromhasCollectionPath=true
fromdCollectionPath=/Contribution Folders/B/1
tohasCollectionPath=true
todCollectionPath=/Contribution Folders/t4/1
```

7.2.44 COLLECTION_MOVE_COLLECTION

Service that moves a folder into a different Contribution Folder. Compare COLLECTION_MOVE_ALL. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following sets of parameters to identify the Contribution Folder to move:

- fromhasCollectionID: (Boolean) Specifies that the fromdCollectionID parameter is used to identify the source Contribution Folder.

  fromdCollectionID: The source Contribution Folder ID.

- fromhasCollectionPath: (Boolean) Specifies whether the fromdCollectionPath parameter is used to identify the source Contribution Folder.

  fromdCollectionPath: The path to the source Contribution Folder.

- fromhasCollectionGUID: (Boolean) Specifies whether the fromdCollectionGUID parameter is used to identify the source Contribution Folder.

  fromdCollectionGUID: The source Contribution Folder GUID.

One of the following sets of parameters to identify the destination parent Contribution Folder:

- tohasCollectionID: (Boolean) Specifies whether the todCollectionID parameter is used to identify the destination parent Contribution Folder.

  todCollectionID: The ID for the destination parent Contribution Folder.

- tohasCollectionPath: (Boolean) Specifies whether the todCollectionPath parameter is used to identify the destination parent Contribution Folder.

  todCollectionPath: The path to the destination parent Contribution Folder.

- tohasCollectionGUID: (Boolean) Specifies whether the todCollectionGUID parameter is used to identify the destination parent Contribution Folder.

  todCollectionGUID: The GUID for the destination parent Contribution Folder.

- tolevel0: Specifies the base value in the path for the destination parent Contribution Folder (for example, `/Contribution Folders`).

  tolevel1: Specifies the next level value in the path for the destination parent Contribution Folder (for example, `/t4`).

  tolevel2: Specifies the next level value in the path for the destination parent Contribution Folder (for example, `/1`).

Example

```
IdcService=COLLECTION_MOVE_COLLECTION
```

```
fromhasCollectionID=true
fromdCollectionID=257
tohasCollectionID=true
todCollectionID=218
```

7.2.45 COLLECTION_MOVE_ITEM

Service that moves a single content item (or link) from the specified Contribution Folder to the destination Contribution Folder. If an item with the same file name exists in the target Contribution Folder, the move fails. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

- One of the following sets of parameters to identify the destination Contribution Folder:
  - tohasCollectionID: (Boolean) Specifies whether the todCollectionID parameter is used to identify the destination Contribution Folder.

    todCollectionID: The ID for the destination Contribution Folder.
  - tohasCollectionPath: (Boolean) Specifies whether the todCollectionPath parameter is used to identify the destination Contribution Folder.

    todCollectionPath: The path to the destination Contribution Folder.
  - tohasCollectionGUID: (Boolean) Specifies whether the todCollectionGUIID parameter is used to identify the destination Contribution Folder.

    todCollectionGUID: The GUID for the destination Contribution Folder.
  - tolevel0: Specifies the first value in the path for the destination Contribution Folder (for example, `/Contribution Folders`).

    tolevel1: Specifies the next level value in the path for the destination Contribution Folder (for example, `/t4`).

    tolevel2: Specifies the next level value in the path for the destination Contribution Folder (for example, `/1`).
- One of the following sets of parameters to identify the content item to move:
  - dDocName: The Content ID for the content item.
  - fromhasCollectionPath: (Boolean) Specifies whether the fromhasCollectionPath parameter is used to identify the Contribution Folder for the content item.

    fromdCollectionPath: The path to the Contribution Folder for the content item.
  - isLink: (Boolean) Specifies whether the dLinkID parameter is used to identify the shortcut to the content item.

    dLinkID: The Link ID for the shortcut to the content item.

Optional Service Parameters

- toContentName: New file name for the moved item.
- destMustExist: Defines (true/false) whether the destination for the item or link to move must exist. If this parameter is *false* or missing, and no destination

Contribution Folder is specified, then the item is removed from its current Contribution Folder (xCollectionID is set to 0).

Example

```
IdcService=COLLECTION_MOVE_ITEM
tohasCollectionID=true
todCollectionID=218
dDocName=TG000570
toContentName=moved_name.txt
```

7.2.46 COLLECTION_MOVE_LOT

Service that moves a set of items including Contribution Folders, content items, and shortcuts to Contribution Folders and content items to a single destination Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following sets of parameters to identify the destination parent Contribution Folder for the moved items.

- tohasCollectionID: (Boolean) Specifies whether the todCollectionID parameter is used to identify the destination parent Contribution Folder.

  todCollectionID: The Contribution Folder collection ID.

- tohasCollectionPath: (Boolean) Specifies whether the todCollectionPath parameter is used to identify the path to the destination Contribution Folder.

  todCollectionPath: The path to the destination Contribution Folder.

- tohasCollectionGUID: (Boolean) Specifies whether the todCollectionGUID parameter is used to identify the destination Contribution Folder.

  todCollectionGUID: The GUID for the destination Contribution Folder.

Parameters required to designate content items. {*n*} designates an integer used to distinguish parameters particular to one item to move from other items to move.

- contentselect{*n*}: (Boolean) Specifies whether to move the content item. To move the item, the value must be *true*.
- fromContentisLink{*n*}: (Boolean) If the item is a content item and not a shortcut, the value must be 0.
- fromdDocName{*n*}: The content ID of the item to move.

Parameters required to designate shortcuts to content items. {*n*} designates an integer used to distinguish parameters particular to one item to move from other items to move.

- contentselect{*n*}: (Boolean) Specifies whether to move the shortcut. To move the shortcut, the value must be *true*.
- fromContentisLink{*n*}: (Boolean) Specifies whether this is a shortcut and not a content item. If the item is a shortcut, the value must be 1.
- fromContentdLinkID{*n*}: The link ID of the shortcut to move.

- fromdDocName{*n*}: The content ID of the content item pointed to by the shortcut.

Parameters required to designate Contribution Folders. {*n*} designates an integer used to distinguish parameters particular to one item to move from other items to move.

- collectionselect{*n*}: (Boolean) Specifies whether to move the Contribution Folder. To move the shortcut, the value must be *true*.
- fromCollectionisLink{*n*}: (Boolean) Specifies whether this is a Contribution Folder and not a shortcut. If this is a Contribution Folder, the value must be 0.
- fromdCollectionID{*n*}: The dCollectionID of the Contribution Folder to move.
- fromhasCollectionID{*n*}: To move the Contribution Folder, the value must be 1.

Parameters required to designate shortcuts to Contribution Folders. {*n*} designates an integer used to distinguish parameters particular to one item to move from other items to move.

- collectionselect{*n*}: (Boolean) Specifies whether to move the shortcut. To move the shortcut, the value must be *true*.
- fromCollectionisLink{*n*}: (Boolean) Specifies whether this is a shortcut and not a Contribution Folder. If this is a shortcut, the value must be 1.
- fromdCollectiondLinkID{*n*}: = The link ID of the shortcut to move.
- fromdCollectionID{*n*}: The dCollectionID of the Contribution Folder pointed to by the shortcut.
- fromhasCollectionID{*n*}: To move the Contribution Folder shortcut, the value must be 1.

Example

```
IdcService=COLLECTION_MOVE_LOT
tohasCollectionID=true
docCollectionID=999999999999000633
contentselect1=true
fromContentisLink1=0
fromdDocName1=PHL11GX010033
```

7.2.47 COLLECTION_NEW

Service used to create a new Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

- hasParentCollectionID: (Boolean) Specifies whether the Contribution Folder being added as a parent Contribution Folder. If set to *true*, dParentCollectionID must be defined.
- dParentCollectionID: (Boolean) Specifies whether the Contribution Folder ID for the parent Contribution Folder. Used when hasParentCollectionID is set to *true*.
- dCollectionInherit: (Boolean) Specifies whether to inherit Contribution Folder metadata.

Example

```
IdcService=COLLECTION_NEW
hasParentCollectionID=true
dParentCollectionID=1
dCollectionInherit=0
```

7.2.48 COLLECTION_PROFILE_GET_COLUMNS

Service that return user hierarchy column information and metadata field information. There are no parameters.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Returned Data

- ResultSets:
  - COLUMNS (User-specific column set.) If the requesting user has not selected columns, the default columns returned are:
    - dFileSize
    - dDocType
    - dDocAuthor
    - dSecurityGroup
  - metadata (information about system and custom metadata fields)

7.2.49 COLLECTION_PROFILE_UPDATE_COLUMNS

Service that updates user hierarchy columns. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.50 COLLECTION_RESTORE_COLLECTION

Service that restores a Contribution Folder that is currently in the Trash folder to its original location. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.51 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 Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.52 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 Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.53 COLLECTION_SEARCH_RESULTS

Service that displays the search results of a search for Contribution Folders that match search criteria. Fields used for the search operands are those in the COLMETA and COLLECTIONS table. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Optional Service Parameters

- ResultCount: Limits the number of items returned in the SearchResuls result set. The default is 25.
- SortField: Field used as a sort key. The default is dCollectionName.
- SortOrder: The sort direction. The default is ASC (ascending).
- *fieldname*: Name of the field to be used as part of a string search condition. Example: `dSecurityGroup=true`
- op*fieldname*: Operator, relative to the name field, to be used in a string search condition. Example: `opdSecurityGroup=hasAsSubstring`
- compare*fieldname*: Value to be use in a string search condition, relative to named field. Example: `comparedSecurityGroup=public`
- *fieldname*LE or *fieldname*GE: Field and value to be used in a date search condition. Example: `dCreateDatedateLE=2008-03-01 00:01:00`

Results

- ResultSets:
  - SearchResults (list of Contribution Folders, with metadata, matching the search conditions)

Examples

- startsWith search:

```
dCollectionName=true&opdCollectionName=beginsWith&comparedCollectionName=a
```

- endsWith search:

```
dCollectionName=true&opdCollectionName=endsWith&comparedCollectionName=12
```

- exact string match search:

```
dCollectionName=true&opdCollectionName=equals&comparedCollectionName=A-EZ-9
```

- field not empty search:

```
dDocTitle=true&opdDocTitle=hasAsSubstring&comparedDocTitle=%
```

- boolean field search:

  ```
  xBooleanTestField=true&opxBooleanTestField=equals&comparexBooleanTestField=1
  ```

- later than search:

  ```
  dCreateDatedateGE=2009-10-01 00:01:00
  ```

- earlier than search

  ```
  dCreateDatedateLE=2008-03-01 00:01:00
  ```

7.2.54 COLLECTION_SET_ARCHIVE

Service that takes the uploaded archive and applies it to the system. This service deletes any Contribution Folders that are in conflict with the archive. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/olders_service.htm

7.2.55 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 Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.56 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 Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.57 COLLECTION_UPDATE

Service that modifies field data for a Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following sets of parameters to identify the Contribution Folder:

- hasCollectionID: (Boolean) Specifies whether the dCollectionID parameter is used to identify the Contribution Folder.

  dCollectionID: The Contribution Folder ID.

- hasCollectionPath: (Boolean) Specifies whether the dCollectionPath parameter is used to identify the Contribution Folder.

dCollectionPath: The path of the Contribution Folder.

- hasCollectionGUID: (Boolean) Specifies whether the dCollectionGUID parameter is used to identify the Contribution Folder.

  dCollectionGUID: The GUID of the Contribution Folder.

Optional Service Parameters

- dCollectionName: Collection ID for the Contribution Folder.
- dCollectionMark:
- dCollectionEnabled:
- dPromptForMetadata:
- dCreateDate: Date the content item was created.
- dInDate: Content release date.
- dOutDate: Content expiration date.
- dReleaseDate: Release state (used to indicate the Web state of the revision).
- dLastModifiedDate: Date of the last modification to the content item.
- dDocName: Content ID for the content item.
- dDocType: Content item type.
- dDocTitle: Content item title.
- dDocAuthor: Content item author.
- dRevLabel: Revision label for the content item. If set, the label is used to locate the specified revision.
- dSecurityGroup: Security group such as *Public* or *Secure*.
- dDocAccount: Account for the content item. Required only if accounts are enabled.
- dCollectionOwner: Contribution Folder owner.
- dCollectionCreator: Overrides the user set as the Contribution Folder creator. The default value is the current user.
- dCollectionModifier: Overrides the user set as the last Contribution Folder modifier. The default value is the current user.
- x?????: Any of the fields as defined in the ColMeta table, which must exactly match the custom metadata fields in DocMeta.

Example

```
IdcService=COLLECTION_UPDATE
dCollectionName=Products
dParentCollectionID=1
dCollectionOwner=bsmith
```

7.2.58 COLLECTION_UPDATE_ADMIN_CONFIG

Service that updates the Contribution Folders system configuration settings. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.59 COLLECTION_UPDATE_ADMIN_INHERIT_CONFIG

Service that updates the Contribution Folders system configuration setting by inheriting the metadata from an existing Contribution Folder (from which metadata is propagated). The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.60 COLLECTION_UPDATE_ADMIN_METADATA_DEFAULTS

Service that updates the Contribution Folders system metadata field default system settings. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.61 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 Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.62 COLLECTION_UPDATE_ITEM

Service that updates a specific item in a collection and changes the defined metadata. Contribution Folders related service (table Folders_Services). The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.63 COLLECTION_UPDATE_META

Service that propagates metadata values from a Contribution Folder to all underlying Contribution Folders and content. The most likely errors are mismatched parameters or the Contribution Folder does not exist. No returned data.

Service Class: Meta Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following sets of parameters to identify the Contribution Folder:

- hasCollectionID: (Boolean) Specifies that the dCollectionID parameter is used to identify the Contribution Folder.

  dCollectionID: The Contribution Folder ID of the parent Contribution Folder.

- hasCollectionPath: (Boolean) Specifies that the dCollectionPath parameter is used to identify the Contribution Folder.

  dCollectionPath: The path of the parent Contribution Folder.

- hasCollectionGUID: (Boolean) Specifies that the dCollectionGUID parameter is used to identify the Contribution Folder.

  dCollectionGUID: The GUID of the parent Contribution Folder.

Example

```
IdcService=COLLECTION_UPDATE_META
hasCollectionGUID=true
dCollectionGUID=5A2E5617-7F95-7356-D923-F6DA894D489C
```

7.2.64 COLLECTION_UPDATE_META_TABLE

Service that updates a Contribution Folder metadata table. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Meta Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.65 COLLECTION_UPDATE_PROFILE_METADATA_DEFAULTS

Service that updates the profile metadata field defaults. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.66 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 Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources//folders_service.htm

7.2.67 COLLECTION_UPDATE_STRUCTURE

Service that refreshes the Contribution Folder cache. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Meta Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.68 COLLECTION_VERIFY_FOLDER_NAME

Service that validates that the given Contribution Folder name is a valid name for a Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

One of the following sets of parameters to identify the Contribution Folder:

- dCollectionName: The Collection ID for the Contribution Folder.
- hasCollectionPath: (Boolean) Specifies that the dCollectionPath parameter is used to identify the Contribution Folder.

  dCollectionPath: The path of the parent Contribution Folder.

Optional Service Parameters

- CollectionSkipIllegalFolderValidation: Allows all validation to be skipped except csCollectionFolderHasNoName. The default is *false*.

Configuration Parameters

- CollectionIllegalFolderCharacters: String containing any illegal characters to use in a folder name. The default is `/\:*?"<>¦`
- dCollectionNameSize: Maximum number of characters for a Contribution Folder name. The default is `100`.
- CollectionIllegalFolderPattern_1: A regular expression. If the expression matches the Contribution Folder name, the verification will fail. The default is `\\..*`
- CollectionIllegalFolderPattern_2: A regular expression. If the expression matches the Contribution Folder name, the verification will fail. The default is `.*\\.`
- CollectionIllegalFolderPattern_3: A regular expression. If the expression matches the Contribution Folder name, the verification will fail. The default is `com[\\d]`
- CollectionIllegalFolderPattern_4: A regular expression. If the expression matches the Contribution Folder name, the verification will fail. The default is l`pt[\\d]`
- CollectionIllegalFolderPattern_5: A regular expression. If the expression matches the Contribution Folder name, the verification will fail. The default is `con`
- CollectionIllegalFolderPattern_6: A regular expression. If the expression matches the Contribution Folder name, the verification will fail. The default is `nul`
- CollectionIllegalFolderPattern_7: A regular expression. If the expression matches the Contribution Folder name, the verification will fail. The default is `prn`

Additional CollectionIllegalFolderPattern_{*n*} variables can be added. They are evaluated by the service as long as they are in sequence.

Results

- ResultSets:
  - SUCCESS
  - FAILED (csCollectionFolderHasNoName)
  - FAILED (csCollectionInvalidCharacter:pos=*position of first bad char*)
  - FAILED (csCollectionNameTooLong:*folder name*,*number of extra characters*)
  - FAILED (csCollectionIllegalPatternMatch:*folder name*,*illegal folder pattern*)

Example

```
IdcService=COLLECTION_VERIFY_FOLDER_NAME
hasCollectionPath=true
dCollectionPath=/Contribution Folders/dept/hr
CollectionSkipIllegalFolderValidation=true
```

```
dCollectionNameSize=90
```

7.2.69 FOLDERSLOCAL_BUILD_MOUNT

Service that rewrites the files of the specified local Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_mounted_service.htm

7.2.70 FOLDERSLOCAL_CREATE_MOUNT

Service that maps a Contribution Folder to a local file system. Whenever the Contribution Folder changes, the mapped directory is updated.

There is no security on this directory.

The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: SearchService

Location: *IdcHomeDir*/components/Folders/resources/folders_mounted_service.htm

7.2.71 FOLDERSLOCAL_DELETE_MOUNT

Service that removes the specified local Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_mounted_service.htm

7.2.72 FOLDERSLOCAL_UPDATE_MOUNT

Service that updates the files of the specified local Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Search Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_mounted_service.htm

7.2.73 GET_FOLDER_HISTORY_REPORT

Service that retrieves history data for Contribution Folders. The requesting user must have admin role.

Service Class: DocService

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

Additional Required Service Parameters

- actionDateGreaterThan: A date in ODBC format. All history data from two minutes before this time and later is returned. The two minutes is a margin for load-balanced servers. The default value is five minutes before the current time.

Results

- ResultSets
  - HistoryReport: Each row contains information about a Contribution Folder history event: rename, update, move, or delete.

Example

```
IdcService=GET_FOLDER_HISTORY_REPORT
actionDateGreaterThan={ts '2010-09-09 18:04:06.293'}
```

7.2.74 GET_OPTION_LISTS

Contribution Folders-related service that retrieves the option lists. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Service (general service)

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.75 GOTO_COLLECTION

Service that displays a Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

7.2.76 GOTO_ROOT_COLLECTION

Service that displays the root Contribution Folder. The most likely errors are mismatched parameters or the Contribution Folder does not exist.

Service Class: Doc Service

Location: *IdcHomeDir*/components/folders_g/resources/folders_service.htm

8

Folders Services

This chapter describes the Oracle WebCenter Content services available when using and customizing Folders services.

This chapter covers the following topics:

- About Folders Services
- Folders Services

8.1 About Folders Services

Folders is the name given to the current WebCenter Content feature that provides a hierarchical folder interface, similar to a conventional file system, for organizing and locating some or all of the content in the Content Server repository. This feature is supported by the FrameworkFolders component in WebCenter Content. For information about services used with the earlier version of Folders named *Contribution Folders*, supported by the Folders_g component in WebCenter Content, see Chapter 7.

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific Folders services are listed within each individual service.

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

8.2 Folders Services

Individual Folders service types are noted in each service description. The following services are described in this section:

- FLD_BROWSE
- FLD_BROWSE_POPUP
- FLD_COPY
- FLD_CREATE_FILE
- FLD_CREATE_FILE_FORM
- FLD_CREATE_FOLDER

- FLD_CREATE_FOLDER_FORM
- FLD_DELETE
- FLD_EDIT_FILE
- FLD_EDIT_FILE_FORM
- FLD_EDIT_FOLDER
- FLD_EDIT_FOLDER_FORM
- FLD_EDIT_METADATA_RULES
- FLD_EDIT_METADATA_RULES_FORM
- FLD_FOLDER_MIGRATION_STATUS
- FLD_FOLDER_SEARCH
- FLD_FOLDER_SEARCH_FORM
- FLD_GET_CHOOSE_DESTINATION_DIALOG
- FLD_GET_CREATE_LINK_DIALOG
- FLD_GET_CREATE_SHORTCUT_DIALOG
- FLD_GET_RENAME_FILE_DIALOG
- FLD_GET_RENAME_FOLDER_DIALOG
- FLD_INFO
- FLD_LOAD_SOFT_LINKS_FOR_DOCUMENT
- FLD_MIGRATION_FOLDER_DATA
- FLD_MOVE
- FLD_PRE_CHECKIN
- FLD_PROPAGATE
- FLD_PROPAGATE_FORM
- FLD_REINDEX_FOLDER_CONTENTS
- FLD_RETRIEVE_CHILD_FILES
- FLD_RETRIEVE_CHILD_FOLDERS
- FLD_SUBSCRIBE
- FLD_TEST_USER_CAPABILITIES
- FLD_UNFILE
- FLD_UNSUBSCRIBE

8.2.1 FLD_BROWSE

Service that is used to browse through the folder structure. It can be used to paginate through the folder structure in these browse modes:

- Paging through folders and files separately (default behavior)
- Paging through folders and files combined (must set `doCombinedBrowse=true`)

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- item: (Required if no fFolderGUID, no path, and no items are specified.) The item to browse. Specify in the form *path*:*path to item* or fFolderGUID is specified, *fFolderGUID*:*GUID of item*. Can be used in both browse modes.
- items: (Required if no fFolderGUID, no path, and no item are specified.) The items to browse. Specify in the form of a comma-delinated list of identifiers in the form *path*:*path to item* or *fFolderGUID*:*GUID of item*. Can be used in both browse modes.
- path: (Required if no lFolderGUID is specified.) The full path to the current folder. Can be used in both browse modes.
- fFolderGUID: (Required if no path is specified.) Can be used instead of the path to reference the current folder. Can be used in both browse modes.

Optional Service Parameters

- fldapp: Specifies the Folders Application of the location to which the user is browsing. Can be used in both browse modes.
- doCombinedBrowse: (Integer) If set, performs the service in a combined pagination mode. Default is 0 (zero).
- foldersFirst: (Integer) If set, lists folders before files in the combined pagination mode. Default is 1.
- folderCount: (Integer) The number of folders to return. Default is 50. Can be used in separate browse modes.
- folderStartRow: (Integer) The row number at which to start returning data. Used for pagination. Default is 0 (zero). Can be used in separate browse modes.
- fileCount: (Integer) The number of files to return. Default is 50. Can be used in separate browse modes.
- fileStartRow: (Integer) The row number at which to start returning data. Used for pagination. Default is 0 (zero). Can be used in separate browse modes.
- combinedCount: (Integer) The number of items (folders plus files) to return. Default is 100. Can be used in combined browse modes.
- combinedStartRow: (Integer) The row number at which to start returning data. Used for pagination. Default is 0 (zero). Can be used in combined browse modes.
- doRetrieveTargetInfo: (Integer) If set, returns target folder's information for all shortcuts retrieved in the ChildTargetFolders ResultSet. Default is 0 (zero). Can be used in both browse modes.
- doMarkFavorites: (Integer) If set, adds a fIsFavorite field in the ResultSet to indicate if the folder or file is favorite. It is a favorite if it has a shortcut in the Favorites folder. Default is 0 (zero). Can be used in both browse modes.
- doRetrieveDocumentURL: (Integer) If set, adds a URL field in the files ResultSet that is the absolute Web location of the document. Default is 0 (zero). Can be used in both browse modes.
- doRetrieveUniqueLinks: (Integer) If set, post processes the results to return only unique links. For example, if folder's shortcut and folder itself are in the ResultSet, only folder itself will be returned. Same for files. Default is 0 (zero). Can be used in both browse modes.

- foldersFilterParams: The comma-delinated list of filter parameters for the browse. For example, `foldersFilterParams=fIsContribution&fIsContribution=1` will return folders with `fIsContribution=1`. Can be used in both browse modes.
- foldersFilterQuery: Standard query for filtering out the folders. This query is in DATABASE engine format in the FolderFolders table. For example, `foldersFilterQuery=fFolderName<substring>`test``. Can be used in both browse modes.
- filesFilterQuery: Standard query for filtering out the files. This query is in DATABASE engine format in the FolderFiles table. For example, `filesFilterQuery=fFileName<substring>`test``. Can be used in both browse modes.
- filesSortField: Field name from the FolderFiles table on which to sort the records. Can be used in both browse modes.
- filesSortOrder: Specify `Asc` or `Desc` for an ascending or descending sort. Can be used in both browse modes.
- foldersSortField: Field name from the FolderFolders table on which to sort the records. Can be used in both browse modes.
- foldersSortOrder: Specify `Asc` or `Desc` for an ascending or descending sort. Can be used in both browse modes.
- $fieldName: The value for the specified field in the foldersFilterParams and filesFilterParams parameters. Can be used in both browse modes.
- filesFilterParams: The comma-delinated list of filter parameters on the browse. For example, `filesFilterParams=fOwner&fOwner=sysadmin` will return folders with `fOwner=sysadmin`. Can be used in both browse modes.
- filterFavorites: If set to `Folders`, this filters results to only include folders that are favorites. If set to `Files`, this filters only files that are favorites. If set to `1`, this filters both folders and files that are favorites. By default, this filters none. Can be used in both browse modes.
- filterFollows: If set to `Folders`, this filters results to only include folders that are followed. If set to `Files`, this filters only files that are followed. If set to `1`, this filters both folders and files that are followed. By default, it filters none. Can be used in both browse modes.
- combinedSortField: If you use this parameter to list the contents, then its value is assigned to sortField, overriding any value set for sortField in the binder. If you use this parameter to list the contents of system libraries and query folders, the mapping converts the fields in the FolderFiles table to the corresponding field values of core tables.
- combinedSortOrder: If you use this parameter to list the contents, then its value is assigned to sortOrder, overriding any value set for sortOrder in the binder.

Results

- ResultSets:
  - FolderInfo: Information about the folder the user is currently browsing.
  - ChildFolders: Information about all of the folders that exist within this particular folder.
  - ChildTargetFolders: Information about all of the target folders of shortcuts in the ChildFolders ResultSet.

  - ChildFiles: Information about all of the files that exist within this particular folder.
- numFolders: (string) The number of folders in the ChildFolders ResultSet.
- hasMoreChildFolders: (Boolean) This is *true* if the request did not return all of the child folders. This occurs when the folderCount value is reached and additional folders could have been returned.
- numFiles: (Integer) The number of files in the ChildFiles ResultSet.
- TotalChildFoldersCount: (Integer) The total number of items (folders and files) in the parent folder.
- TotalChildFilesCount: (Integer) The total number of files in the parent folder.
- TotalChildItemsCount: (Integer) The total number of items (folders and files) in the parent folder.
- hasMoreChildFiles: (Boolean) This is *true* if the request did not return all of the child files. This occurs when the fileCount value is reached and additional documents could have been returned.
- hasMoreChildItems: (Boolean) This is *true* if the request did not return all of the child items (folders and files). This occurs when count is reached and there are additional items that could have been returned.

8.2.2 FLD_BROWSE_POPUP

Services that prepares a small pop-up containing a tree control which can be used to browse through the Folders structure and then select either documents or folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- target: The entity type the user is targeting with this pop-up. Can be set to *file* or *folder* depending on what the user is allowed to select.

Results

- ResultSets:
  - ChildFolders: The child folders of the root item.
  - ChildFiles: The child files of the root item. Will be null if the target is not set to *file*.

8.2.3 FLD_COPY

Service that allows a user to copy items from one location to another in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- destination: The destination to which all of the items will be copied. This identifier must be in the form *path:$PATH'* or *fFolderGUID:$FOLDER_GUID*.

- item1: The item to be copied. If more than one item is being copied in this operation, you can specify *item2*, *item3*, and so forth. Specify these items either as IDs or as full paths. The items must be in the form of path:$*PATH*, fFolderGUID:$*FOLDER_GUID*, or fFileGUID:$*FILE_GUID*.

Optional Service Parameters

- overwrite: Set to 1 to overwrite any destinations that already exist in the case of naming conflicts. Items that are overwritten are moved to the trash.
- copyOwnerFilesToNewFiles: Set to 1 to copy all files of type `owner` to a new owner file. This action causes a new checkin to occur for each owner file being copied. By default, the copy of an owner file is a "soft" file that points to the source document.
- copySoftFilesToNewFiles: Set to 1 to create new content items for all soft files being copied. By default, a copy of a soft file is another soft file pointing to the same content item.
- constructDialog : Set this to 1 to make the server automatically construct an HTML dialog with the results of the copy action or information about why the copy action may have failed.

Results

- ResultSets:
  - ItemsRequiringOverwrite: If the overwrite flag is not set, and if any conflicting items exist in the destination, this ResultSet lists what conflicts exist. If there are any conflicts, none of the items are moved. This return allows for a quick server response on potential conflicts, enabling the client application to prompt the user for overwrite.
  - TaskList: Information about the tasks performed and which tasks were successful
- didBackgroundTask: (string) Set this parameter to 1 if some or all of the task was backgrounded.
- mainTasksComplete: (string) Set this parameter to 1 if the main tasks were completed, even if the service was backgrounded. It is useful to know that items have been copied, even if the post-copy checks have not yet finished.
- dialogMarkup: (string) HTML for the dialog.
- dialogScript: (string) JavaScript required to display the dialog.

8.2.4 FLD_CREATE_FILE

Service that creates a link to a document in Folders.

Service Class: intradoc.folders.FoldersService

Location: *`IdcHomeDir`*`/components/frameworkfolders/resources/frameworkfolders_service.htm`

Required Service Parameters

- fParentGUID: The GUID of the parent folder in which the new link will be created.

Optional Service Parameters

- $fileMeta: Metadata to be assigned to the link.

File Name Restrictions

- A file name cannot start or end with a space.
- A file name cannot be longer than 250 characters.
- Restricted characters:

Table 8–1 Restricted File Name Characters

Character	Description
/	Slash
\	Back slash
*	Asterisk, or star
"	Quote
<	Less than
>	Greater than
|	Vertical bar, or OR
?	Question mark
:	Colon

- Restricted strings:

Table 8–2 Restricted File Name Strings

String
Users
/
_shortcuts
_REAL_ITEMS
(space or consecutive space)
.
..
CON
PRN
AUX
CLOCK
NUL
NULL
COM0
COM1
COM2
COM3
COM4
COM5
COM6

Table 8–2 (Cont.) Restricted File Name Strings

String
COM7
COM8
COM9
LPT0
LPT1
LPT2
LPT3
LPT4
LPT5
LPT6
LPT7
LPT8
LPT9

8.2.5 FLD_CREATE_FILE_FORM

Service that displays a Folders form for creating a new link object which points to a document.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fParentGUID: The GUID of the parent folder in which the new link will be created.
- fFileType: The type of link to create.
- dDocName: The document name of the target of the link.

Optional Service Parameters

- fApplication: The Folders Application creating the folder. The default is *framework*.
- $fileMeta: Default metadata values for the link.

Results

- ResultSets:
  - ParentInfo: Information about the parent folder.
- parentPath: (string) The full path to the parent folder.

8.2.6 FLD_CREATE_FOLDER

Service that creates a folder in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fParentGUID: The GUID of the parent folder in which the new folder will be created.

Optional Service Parameters

- $folderMeta: Metadata to be assigned to the folder.

Folder Name Restrictions

- A folder name cannot start or end with a space.
- A folder name cannot be longer than 250 characters.
- Restricted characters:

Table 8–3 Restricted Folder Name Characters

Character	Description
/	Slash
\	Back slash
*	Asterisk, or star
"	Quote
<	Less than
>	Greater than
|	Vertical bar, or OR
?	Question mark
:	Colon

- Restricted strings:

Table 8–4 Restricted Folder Name Strings

String
Users
/
_shortcuts
_REAL_ITEMS
(space or consecutive space)
.
..
CON
PRN
AUX
CLOCK
NUL

Table 8–4 (Cont.) Restricted Folder Name Strings

String
NULL
COM0
COM1
COM2
COM3
COM4
COM5
COM6
COM7
COM8
COM9
LPT0
LPT1
LPT2
LPT3
LPT4
LPT5
LPT6
LPT7
LPT8
LPT9

8.2.7 FLD_CREATE_FOLDER_FORM

Service that displays a Folders form for creating new folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fParentGUID: The GUID of the parent folder in which the new folder will be created.

Optional Service Parameters

- fApplication: The Folders Application creating the folder. The default is *framework*.
- fFolderType: The type of folder to create. The default is *owner*.
- $folderMeta: Default metadata values for the folder.
- fTargetGUID: If the fFolderType value is *soft*, this must be the GUID of the target folder.

Results

- ResultSets:
  - ParentInfo: Information about the parent folder.
  - TargetInfo: Information about the target folder, if there is a target folder.
- parentPath: (string) The full path to the parent folder.
- targetPath: (string) The full path to the target, if there is a target.

8.2.8 FLD_DELETE

Service that allows a user to delete one or more items from the Framerwork Folders hierarchy. This immediately deletes the items, even if the items are not already in the trash.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*`/components/frameworkfolders/resources/frameworkfolders_service.htm`

Required Service Parameters

- item1: The item to be deleted. If more than one item is being deleted in this operation, then you can specify *item2*, *item3*, and so on. These items can be specified either as IDs or as full paths, and they must be specified in one of these forms:
  - *path*:$*PATH*
  - fFolderGUID:$*FOLDER_GUID*
  - fFileGUID:$*FILE_GUID*

Optional Service Parameters

- constructDialog: (Boolean) Set this parameter to 1 if you want the server to automatically construct an HTML dialog with the results of the FLD_DELETE operation or information about why the operation may have failed.

Results

- ResultSets:
  - TaskList: Information about the tasks performed and which were successful.
- didBackgroundTask: (Boolean) Set to 1 if some or all of the task was backgrounded.
- dialogMarkup: (string) HTML for the dialog.
- dialogScript: (string) JavaScript necessary to display the dialog.
- mainTasksComplete: (Boolean) Set to 1 if the main tasks were completed, even if the service was backgrounded. It is useful to know that items have been deleted, even if the post-delete checks have not yet finished.

8.2.9 FLD_EDIT_FILE

Service that edits a link to a document in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFileGUID: The ID of the link being edited.

Optional Service Parameters

- $fileMeta: Metadata to be assigned to the link being edited.

8.2.10 FLD_EDIT_FILE_FORM

Service that displays a form for editing links to documents in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFileGUID: The GUID of the link being edited.

Results

- $fileMeta: Metadata values currently assigned to the link being edited.
- folderPath: (string) The full path to the link being edited.

8.2.11 FLD_EDIT_FOLDER

Service that edits a folder in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFolderGUID: The ID of the folder being edited.

Optional Service Parameters

- $folderMeta: Metadata to be assigned to the folder being edited.

8.2.12 FLD_EDIT_FOLDER_FORM

Service that displays a form for editing folders in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFolderGUID: The GUID of the folder being edited.

Results

- ResultSets:
  - $folderMeta: Metadata values currently assigned to the folder being edited.

- folderPath: (string) The full path to the folder being edited.
- targetPath: (string) The full path to the target if the folder being edited is a shortcut.
- TargetInfo: Information about the target if the folder being edited is a shortcut.

8.2.13 FLD_EDIT_METADATA_RULES

Service that edits metadata rules assigned to a folder in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFolderGUID: The GUID of the folder being edited.
- $fieldName: The value for the specified field.

Optional Service Parameters

- $fieldName:isDefault: (Boolean) Set to 1 if the specified field is forced upon all children of the folder being edited.
- $fieldName:isForced: (Boolean) Set to 1 if the specified field is forced upon all children of the folder being edited.
- $fieldName:isRecursiveForced: (Boolean) Set to 1 if the specified field is forced upon all children of the folder being edited and upon its children's children.
- $fieldName:inhibitPropagation: (Boolean) Set to 1 if the specified field can be propagated to children of the folder being edited.

8.2.14 FLD_EDIT_METADATA_RULES_FORM

Service that displays a form for editing content metadata defaults assigned to a folder. in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFolderGUID: The GUID of the folder being edited.

Results

- ResultSets:
  - FolderInfo: Information about the folder that was edited.
- $fieldName: (string) The value for the specified field.
- $fieldName:isDefault: (Boolean) Set to 1 if the specified field has a default value associated with it.
- $fieldName:isForced: (Boolean) Set to 1 if the specified field is forced upon all children of the folder.

- $fieldName:isRecursiveForced: (Boolean) Set to 1 if the specified field is forced upon all children of the folder and upon its children's children.
- $fieldName:inhibitPropagation: (Boolean) Set to 1 if the specified field can be propagated to children of the folder.
- folderPath: (string) The full path to the folder.
- dpTriggerField: (string) The profile trigger field, if one exists.
- dpTriggerValue: (string) The current profile trigger value, if one exists.
- dpDisplayLabel: (string) The current profile display label, if one exists.

8.2.15 FLD_FOLDER_MIGRATION_STATUS

Service that displays a dialog for migrating Folders_g legacy folder data into FrameworkFolders data structures. A history of migrations and status of any ongoing migration is displayed.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*`/components/frameworkfolders/resources/frameworkfolders_service.htm`

8.2.16 FLD_FOLDER_SEARCH

Service that search for folders in Folders and returns the result.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*`/components/frameworkfolders/resources/frameworkfolders_service.htm`

Required Service Parameters

- QueryTest: The text of the search query.
- SortField: The field on which to sort the search results.
- SortOrder: The order in which to sort the search results. Use one of these:
  - `asc`
  - `desc`

Results

- ResultSets:
  - SEARCH_RESULTS: The results of the search.

8.2.17 FLD_FOLDER_SEARCH_FORM

Service that displays a form for searching folders in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*`/components/frameworkfolders/resources/frameworkfolders_service.htm`

8.2.18 FLD_GET_CHOOSE_DESTINATION_DIALOG

Service that displays a dialog for choosing a destination during a move or copy operation in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- action: Set to `move` or `copy` depending on the action being performed.
- item1: The item to be copied. If more than one item is being touched in this operation, then *item5*, *item3*, and so on can be specified. These items can be specified either as IDs or as full paths, and they must be in the form of one of these:
  - path:$*PATH*
  - fFolderGUID:$*FOLDER_GUID*
  - fFileGUID:$*FILE_GUID*

Results

- dialogMarkup: (string) HTML for the dialog.
- dialogScript: (string) JavaScript required to display the dialog.

8.2.19 FLD_GET_CREATE_LINK_DIALOG

Service that displays a dialog for creating a link to an existing content item in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fParentGUID: The GUID of the folder in which to create the link. Either this or dDocName must be set.
- dDocName: The target document of the new link. Either this or fParentGUID must be set.
- fFileName: The type of link to create.

Results

- dialogMarkup: (string) HTML for the dialog.
- dialogScript: (string) JavaScript required to display the dialog.

8.2.20 FLD_GET_CREATE_SHORTCUT_DIALOG

Service that displays a dialog for creating a shortcut to an existing folder in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fTargetGUID: The target document ID of the shortcut.
- fParentGUID: The parent folder ID where the shortcut will be created.

Results

- dialogMarkup: (string) HTML for the dialog.
- dialogScript: (string) JavaScript required to display the dialog.

8.2.21 FLD_GET_RENAME_FILE_DIALOG

Service that retrieves a dialog used to rename a file in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFileGUID: The GUID of the file to rename.

Results

- dialogMarkup: (string) HTML for the dialog.
- dialogScript: (string) JavaScript required to display the dialog.

8.2.22 FLD_GET_RENAME_FOLDER_DIALOG

Service that retrieves a dialog used to rename a folder in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFolderGUID: The GUID of the folder to rename.

Results

- dialogMarkup: (string) HTML for the dialog.
- dialogScript: (string) JavaScript required to display the dialog.

8.2.23 FLD_INFO

Service that returns information about a particular folder or file in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- path: The path to the folder or file whose information is being requested. This parameter must be specified if no fFolderGUID or fFileGUID parameter is specified.
- fFolderGUID: The GUID of the folder whose information is being requested. This parameter must be specified if no path or fFileGUID parameter is specified.
- fFileGUID: The GUID of the file whose information is being requested. This parameter must be specified if no path or fFolderGUID parameter is specified.

Results

- ResultSets:
  - FileInfo: Information about the link to the document (file). Also the metadata associated with the document itself.
  - FolderInfo: Information about the folder.
  - TargetInfo: Information about the target (shortcut).
- filePath: (string) The full path to the item if the item is a document link.
- folderPath: (string) The full path to the item if the item is a folder.
- targetPath: (string) The full path to the target item if the item is a shortcut.

8.2.24 FLD_LOAD_SOFT_LINKS_FOR_DOCUMENT

Service that loads all soft links which reference a particular content item in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- dDocName: The name of the content item.
- dID: The identifier for the content item.

Optional Service Parameters

- constructSoftLinkTableRows: (Boolean) Whether or not to construct HTML for the soft link table rows. The default is 1 (true).

Results

- ResultSets:
  - softLinks: The soft links for the content item.
- softLinkTableRows: (string) HTML to display the soft link table rows.

8.2.25 FLD_MIGRATION_FOLDER_DATA

Service that migrates Folders_g legacy folder data into FrameworkFolders data structures.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

8.2.26 FLD_MOVE

Service that moves one or more content items from one location to another in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- destination: The destination where all of the items should be moved. If the destination is an existing folder, then all of the items being moved will be moved into that folder. If you are moving a single file, the destination may point to the location (including the new filename) where the file should be moved. This identifier should be in the form path:$*PATH* or fFolderGUID:$*FOLDER_GUID*.
- item1: The item to be moved. If more than one item is being moved in this operation, the *item2*, *item3*, and so on can be specified. These items can be specified either as IDs or as full paths. They must be in one of the following forms:
  - path:$*PATH*
  - fFolderGUID:$*FOLDER_GUID*
  - fFileGUID:$*FILE_GUID*

Optional Service Parameters

- overwrite: (Boolean) Set to 1 to overwrite any destinations that already exist in the case of naming conflicts. Items that are overwritten are moved to the trash.
- constructDialog: (Boolean) Set to 1 to make the server automatically construct an HTML dialog with the results of the move operation or information about why the move may have failed.

Results

- ResultSets:
  - ItemsRequiringOverwrite: If the overwrite flag is not set, and if any conflicting items exist in the destination, this ResultSet is returned listing which conflicts exist. If there are any conflicts, none of the items are moved. This allows for a quick server response on potential conflicts, and allows the client application to prompt the user for overwrite.
  - TaskList: Information about the tasks performed and which were successful.
- dialogMarkup: (string) HTML for the dialog.
- dialogScript: (string) JavaScript required to display the dialog.
- didBackgroundTask: (string) This is set to 1 if some or all of the task was backgrounded.
- mainTasksComplete: (string) This is set to 1 if the main tasks were completed, even if the service was backgrounded. It is useful to know that items have been moved, even if the post move sanity check has not yet finished.

8.2.27 FLD_PRE_CHECKIN

Service that provides information so a requesting client can determine if a metadata entry dialog should be presented to a user before checking in a content item.

Service Class: intradoc.folders.FoldersService

Location: `IdcHomeDir/components/frameworkfolders/resources/frameworkfolders_service.htm`

Required Service Parameters

- fFolderGUID: The base folder ID for items in CONTENTS. This parameter must be used if fFolderPATH is not set.

- fFolderPath: The base folder path for items in CONTENTS. This parameter must be used if fFolderGUID is not set.
- CONTENTS: (ResultSet) Set of paths for folders and content items relative to the base folder. If the specified path ends with a "/" (slash) character, the path is assumed to be a folder, otherwise it is assumed to be a content item.
- EXTRA_INFO_FIELDS: (ResultSet) A list of extra information fields to be returned for each item. Extra information field data is only populated for content items that already exist. Valid fields that can be requested are any of the columns from the REVISIONS, DOCMETA or DOCUMENTS tables as well as these fields (as in WebDAV): getcontentlength, getcontenttype, getlastmodified, creationdate.

Optional Service Parameters

- allowPromptForExistingItem: (Boolean) If set to 1, enables client requests to allow prompting for second revisions. Normally prompts occurs only for first revisions.

Results

- ResultSet: REQUIRES_METADATA_PROMPT: For each row in CONTENTS, this returned ResultSet contains these fields:
  - FLAG field: Two flag characters: {*flag1*}{*flag2*}. For *flag1*, if set to 1 the specified path already exists, otherwise it does not. For *flag2*, if set to 1 the client is directed to present a prompt dialog for entering metadata for this item.
  - DP_TRIGGER_VALUE: The document profile that was used for this item. It shows one column for each field specified in the EXTRA_INFO_FIELDS parameter.

8.2.28 FLD_PROPAGATE

Service that propagates metadata down through the folder structure in Folders.

Service Class: intradoc.folders.FoldersService

Location: `IdcHomeDir/components/frameworkfolders/resources/frameworkfolders_service.htm`

Required Service Parameters

- $fieldName:isSelected: Set to 1 to propagate the specified field.
- $fieldName: The value of the field to propagate

Optional Service Parameters

- propagateThroughSoftLinks: (Boolean) Set to 1 to propagate metadata to documents pointed to by soft links.
- FldAllowPropagatingEmptyValue: Set it to *true* to propagate empty values to Access Control List (ACL) fields and custom fields. But, you cannot set certain standard fields to empty value, such as DID, XCOLLECTIONID, XIDCPROFILE, XLIBRARYGUID, XCHECKSUM, XESIGHASELECTRONICSIGNATURES, and XCOMMENTS.

8.2.29 FLD_PROPAGATE_FORM

Service that displays a form to allow people to propagate metadata to all children of a folder in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFolderGUID: The ID of the folder.

Results

- ResultSets:
  - FolderInfo: Information about the folder.
- folderPath: The full path to the folder.
- $fieldName: The default value of the field to propagate. This value is taken from metadata rules or the metadata of the folder itself.
- dpTriggerField: The profile trigger field, if one exists.

8.2.30 FLD_REINDEX_FOLDER_CONTENTS

Service that reindexes all currently indexed documents within a particular folder in Folders. Only a system administrator can call this service.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- fFolderGUID: The GUID of the folder.

8.2.31 FLD_RETRIEVE_CHILD_FILES

Service that enables easy pagination of child documents returned by the FLD_BROWSE service in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- path: The full path to the current folder. This parameter must be used if no folderGUID parameter is specified.
- fFolderGUID: The GUID of the folder. This parameter an be specified instead of the path to reference the current folder.

Optional Service Parameters

- fldapp: The Folders Application of the location to be browsed.
- folderCount: The number of folders to return. The default is 50.
- folderStartRow: The row number at which to start returning data. Used for paging responses. The default is 0.

- constructListingMarkup: Specify whether or not the markup for the additional folders should be returned. This parameter is useful when performing this request in an AJAX environment. The default is 1 (true).

Results

- ResultSets:
  - FolderInfo: Information about the folder currently being browsed.
  - ChildFolders: Information about all of the folders that exist within this particular folder.
  - FileListingHTML: HTML that can be used to display the additional documents. Only returned when constructListingMarkup is set to 1 (true).
  - FileListingScript: JavaScript that must be run to display the additional documents. Only returned when constructListingMarkup is set to 1 (true).
- numFolders: The number of folders in the ChildFolders ResultSet.
- hasMoreChildFiles: This is 1 (true) if the request did not return all of the child files. This occurs when fileCount is reached and there are additional documents that could have been returned.

8.2.32 FLD_RETRIEVE_CHILD_FOLDERS

Service that enables easy pagination of child folders returned by the FLD_BROWSE service in Folders.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- path: The full path to the current folder. This parameter must be used if no folderGUID parameter is specified.
- fFolderGUID: The GUID of the folder. This parameter an be specified instead of the path to reference the current folder.

Optional Service Parameters

- fldapp: The Folders Application of the location to be browsed.
- folderCount: The number of folders to return. The default is 50.
- folderStartRow: The row number at which to start returning data. Used for paging responses. The default is 0.
- constructListingMarkup: Specify whether or not the markup for the additional folders should be returned. This parameter is useful when performing this request in an AJAX environment. The default is 1 (true).

Results

- ResultSets:
  - FolderInfo: Information about the folder currently being browsed.
  - ChildFolders: Information about all of the folders that exist within this particular folder.

  - FolderListingHTML: HTML that can be used to display the additional folders. Only returned when constructListingMarkup is set to 1 (true).
  - FolderListingScript: JavaScript that must be run to display the additional folders. Only returned when constructListingMarkup is set to 1 (true).
- numFolders: The number of folders in the ChildFolders ResultSet.
- hasMoreChildFolders: This is 1 (true) if the request did not return all of the child folders. This occurs when fileCount is reached and there are additional folders that could have been returned.

8.2.33 FLD_SUBSCRIBE

Service that allows a user to subscribe to a folder. The subscription notification is sent to the user's e-mail as specified in the profile. If neither the profile e-mail address or the dSubscriptionEmail setting is provided, Content Server sets the flag `isNoSubscriptionNotificationSent=1` in the response binder.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*`/resources/frameworkfolders_service.htm`

Required Service Parameter

- fParentGUID: The ID of the folder to which the user will subscribe.

Optional Service Parameters

- dSubscriptionEmail: The subscriber's e-mail address. If a non-empty value is provided, the service uses the value to update the user's e-mail value in the profile.
- constructDialog: (boolean) Set this to 1 (true) if you want Content Server to automatically construct an HTML dialog with the results of the subscription or information about why the subscription may have failed.

Results

- dialogMarkup: HTML for the dialog.
- dialogScript: JavaScript needed to display the dialog.

8.2.34 FLD_TEST_USER_CAPABILITIES

Service that tests the capability of a user to perform an operation on a document, folder, or library, based on:

- user's authorization
- item's access control policy
- item's state
- enabled WebCenter Content components

Service Class: intradoc.folders.FoldersServices

Location: *IdcHomeDir*`/resources/frameworkfolders_service.htm`

Required Service Parameters

- testedItems: A list of items to be tested. The type and ID of each item should be specified in the string `testedItems` with the format *Type*1:*ID*1,*Type*2:*ID*2,*Type*3:*ID*3

  Type is:

  - fFolderGUID for Folder
  - fFileGUID for File
  - dDocName for Document
  - dID for Document Revision

  ID is the value of those fields.

  For example:

  ```
  fFolderGUID:folder001,dDocName:doc001,fFolderGUID:folder002,
  fFileGUID:file001,dID:docRevision001
  ```

- testedCapabilities: A list of capabilities separated by a comma; for example, `MOVE,COPY,DELETE`. The full list of all capabilities that can be tested is:
  - CREATE_CHILD_FOLDER
  - CREATE_CHILD_DOCUMENT
  - FILE_DOCUMENT
  - UNFILE_DOCUMENT
  - UPDATE_METADATA
  - UPDATE_ACCESS_CONTROL
  - LATEST_REVISION
  - MOVE
  - COPY
  - DELETE
  - PROPAGATE_METADATA
  - CHECK_OUT
  - CANCEL_CHECK_OUT
  - CHECK_IN_NEW_REVISION
  - DELETE_REVISION
  - RESTORE_REVISION
  - SHARE_PERSONAL_FOLDER
  - FOLLOW
  - UNFOLLOW

Capabilities Supported for Testing

Table 8–5

Capability	Folder owner link (fFolderGUID)	Folder soft link (fFolderGUID)	File owner link (fFileGUID)	File soft link (fFileGUID)	Document (dDocName)	Revision (dID)	Description
CREATE_CHILD_FOLDER	TEST						Can the user create a new subfolder in this folder?
CREATE_CHILD_DOCUMENT	TEST						Can the user create a new document in this folder.
FILE_DOCUMENT					TEST		Can the user create an owner link to this document?
UNFILE_DOCUMENT			TEST		TEST(2)		Can the user delete this owner link (without deleting the document)?
UPDATE_METADATA	TEST	TEST	TEST	TEST		TEST	Can the user update the metadata of this item?
UPDATE_ACCESS_CONTROL	TEST					TEST	Can the user change the access control fields of this folder or file?
LATEST_REVISION						TEST	Is this the latest revision of the document?
MOVE	TEST	TEST	TEST	TEST	TEST(2)		Can the user move this folder, document, or shortcut?
COPY	TEST	TEST	TEST	TEST	TEST(2)		Can the user copy this folder, document, or shortcut?
DELETE	TEST	TEST	TEST(1)	TEST	TEST		Can the user delete this folder, document, or shortcut?
PROPAGATE_METADATA	TEST						Can the user propagate the metadata fields of this folder? It returns false for personal library/folders (LibraryType=4).
CHECK_OUT					TEST		Can the user check out this document? It returns false for documents in personal library/folder.
CANCEL_CHECK_OUT					TEST		Can the user cancel the check out of this document?

Table 8–5 (Cont.)

Capability	Folder owner link (fFolderGUID)	Folder soft link (fFolderGUID)	File owner link (fFileGUID)	File soft link (fFileGUID)	Document (dDocName)	Revision (dID)	Description
CHECK_IN_NEW_ REVISION					TEST		Can the user check in a new revision of this document? It returns TRUE for a document that is checked out by the current user, or is not checked out and the user has permissions to do an auto-checkout and checkin cycle.
DELETE_REVISION						TEST	Can the user delete this revision of this document?
RESTORE_REVISION						TEST	Can the user create a new revision of this document as a copy of this revision?
SHARE_PERSONAL_ FOLDER	TEST(3)						Can the user share this personal library or personal folder?
FOLLOW	TEST				TEST		Can the user follow this folder, document, or revision? A document or folder is considered "followed" if it is subscribed to directly.
UNFOLLOW	TEST				TEST		Can the user "unfollow" this folder, document, or revision?

Notes

(1) Equivalent to testing DELETE on the target Revision.

(2) Need to be able to test capabilities without knowing the fFileGUID.

(3) A personal library or personal folder can be shared only if each of the following conditions hold: (a) the ancestor personal library is not shared (for a personal folder); (b) no ancestor personal folder is shared (for a personal folder); and (c) no descendant personal folder is shared. If one or more of these conditions is not true, the user does not have the capability, and the FLD_TEST_USER_CAPABILITIES service should return the fFolderGUIDs of the library or folder that prevent the personal library or folder from being shared.

Results

ResultSet:

- TestResults: The test results in which:
  - "1" means "True" and the current user has the capability
  - "0" means "False"

- "-1" means "N/A" (not applicable)
- a string of shared ancestor/descendant folders will be returned for the failed test of the SHARE_PERSONAL_FOLDER capability because of existing shared ancestor/descendants
- if SHARE_PERSONAL_FOLDER fails because the user doesn't have the share permission on the item, the service returns "0"

8.2.35 FLD_UNSUBSCRIBE

Service that allows a user to unsubscribe a folder.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/resources/frameworkfolders_service.htm

Required Service Parameter

- fParentGUID: The ID of the folder to be unsubscribed.

Optional Service Parameter

- constructDialog: (boolean) Set this to 1 (true) to have Content Server automatically construct an HTML dialog with the results of the unsubscription or information about why the unsubscription may have failed.

Results

- dialogMarkup: HTML for the dialog.
- dialogScript: JavaScript needed to display the dialog.

8.2.36 FLD_UNFILE

Service that allows a user to unfile one or more items from the Folders hierarchy. This immediately deletes the link object without affecting the target document in any way.

Service Class: intradoc.folders.FoldersService

Location: *IdcHomeDir*/components/frameworkfolders/resources/frameworkfolders_service.htm

Required Service Parameters

- item1: The item to be unfiled. If more than one item is being unfiled in this operation, then *item2*, *item3*, and so on can be specified. These items can be specified either as IDs or as full paths. They must be in the form of path:$*PATH* or fFileGUID:$*FILE_GUID*.

Optional Service Parameters

- constructDialog: Set this to 1 if you want the server to automatically construct an HTML dialog with the results of the unfile or information about why the unfile may have failed.

Results

- ResultSets:
  - TaskList: Information about the tasks performed and which were successful.
- didBackgroundTask: This is set to 1 if some or all of the task was backgrounded.

- mainTasksComplete: This is set to 1 if the main tasks were completed, even if parts of the service were backgrounded. It is useful to know that items have been unfiled, even if the post unfile checks have not yet finished.
- dialogMarkup: HTML for the dialog.
- dialogScript: JavaScript required to display the dialog.

9 Electronic Signature Services

This chapter describes the services available when using and customizing electronic signatures in Oracle WebCenter Content.

This chapter covers the following topics:

- Section 9.1, "About Electronic Signature Services"
- Section 9.2, "Electronic Signature Services"

9.1 About Electronic Signature Services

Electronic signature services are used to configure, get, and sign unique identifiers that Oracle WebCenter Content Server generates and stores with revision metadata for content items. Electronic signatures can be used with general content items, content items in workflows, and with PDF Watermark.

Information about what is a WebCenter Content service and how services can be used is provided in Chapter 2. Information about basic services structure, attributes, actions, and a service example is provided in Chapter 3. You should be familiar with this information before customizing current services or creating new services.

The locations for specific Electronic Signature services are listed within each individual service.

For information about using Electronic Signature, see *Oracle Fusion Middleware Using Oracle WebCenter Content*.

9.2 Electronic Signature Services

The following services can be used when the Electronic Signature component is enabled in Oracle WebCenter Content:

- Section 9.2.1, "ESIG_GET_ADMIN_PAGE"
- Section 9.2.2, "ESIG_GET_CONTENT_SIGNATURES"
- Section 9.2.3, "ESIG_GET_CONTENT_SIGNATURES_AND_INFO"
- Section 9.2.4, "ESIG_GET_SIGNATURE_DETAILS"
- Section 9.2.5, "ESIG_GET_SIGN_CONTENT_FORM"
- Section 9.2.6, "ESIG_GET_SIGN_WORKFLOW_FORM"
- Section 9.2.7, "ESIG_GET_USER_DEFINED_FIELDS"
- Section 9.2.8, "ESIG_GET_VALIDATE_FILE_FORM"

- Section 9.2.9, "ESIG_SET_CONFIG"
- Section 9.2.10, "ESIG_SIGN_CONTENT"
- Section 9.2.11, "ESIG_VALIDATE_FILE"
- Section 9.2.12, "ESIG_VALIDATE_FILE_GLOBAL"
- Section 9.2.13, "ESIG_WORKFLOW_SIGN_AND_APPROVE"

9.2.1 ESIG_GET_ADMIN_PAGE

Service that loads the electronic signatures administration page.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Results

- ResultSets:
  - ESigUserDefinedSignatureFields: Information about custom electronic signature fields.

9.2.2 ESIG_GET_CONTENT_SIGNATURES

Service that gets the electronic signatures of a document.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Additional Required Service Parameters

- ESigViewSignaturesDocID: The ID of the document.
- ESigViewSignaturesRevisionID: The revision ID of the document.
- ESigViewSignaturesDocName: The content ID of the document.

Additional Optional Service Parameters

- ESigGetSignaturesAllRevisions: If this parameter is set, the service returns electronic signatures of all revisions of the document.
- ESigCallOrigin: The call origin key.

Results

- ResultSets:
  - DOC_INFO: Information about the document.
  - ESigViewSignaturesResultSet: Information about the signatures.

9.2.3 ESIG_GET_CONTENT_SIGNATURES_AND_INFO

Service that gets the electronic signatures of a document.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Additional Required Service Parameters

- dID: The ID of the document.

Additional Optional Service Parameters

- dDocName: The content ID of the document.
- ESigGetSignaturesAllRevisions: If this parameter is set, the service returns electronic signatures of all revisions of the document.
- ESigCallOrigin: The call origin key.

Results

- ResultSets:
  - DOC_INFO: Information about the document.
  - ESigViewSignaturesResultSet: Information about the signatures.

9.2.4 ESIG_GET_SIGNATURE_DETAILS

Service that gets detailed information about an electronic signature.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Additional Required Service Parameters

- ESigSignatureSequence: The sequence ID of the electronic signature.

Results

- ResultSets:
  - ESigUserDefinedFieldValues: Information from the custom electronic signature fields.
  - UserAttribInfo: Electronic signer's user attributes.

9.2.5 ESIG_GET_SIGN_CONTENT_FORM

Service that loads the electronic signature content form.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Additional Required Service Parameters

- dID: The ID of the document to sign.

- isESigUIAction: This checks for session and cookie according to the new design of electronic signature user reauthentication, which redirects the user to the WebLogic Server login page for authentication. Always set this to `1` (one), which indicates it's a native 11*g* user interface action.
- ESigCallOrigin: The call origin key that determines the redirected page on the native 11*g* user interface after the signing. Set this to `DOC_INFO`.

Results

- ResultSets:
  - DOC_INFO: Information about the document.
  - ESigUserDefinedSignatureFields: Information about the custom electronic signature fields.

9.2.6 ESIG_GET_SIGN_WORKFLOW_FORM

Service that loads the Workflow electronic signature content form.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*`/components/ElectronicSignatures/resources/esig_service.htm`

Additional Required Service Parameters

- dID: The ID of the document.
- isESigUIAction: This checks for session and cookie according to the new design of electronic signature user reauthentication, which redirects the user to the WebLogic Server login page for authentication. Always set this to `1` (one), which indicates it's a native 11*g* user interface action.
- ESigCallOrigin: The call origin key that determines the redirected page on the native 11*g* user interface after the signing. Set this to `WORKFLOW`.

Additional Optional Service Parameters

- ESigWorkflowName: The workflow name.
- ESigWorkflowStepname; The workflow step name.

Results

- ResultSets:
  - DOC_INFO: Information about the document.
  - ESigUserDefinedSignatureFields: Information about custom electronic signature fields.

9.2.7 ESIG_GET_USER_DEFINED_FIELDS

Service that gets the information about custom electronic signature fields.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*`/components/ElectronicSignatures/resources/esig_service.htm`

Results

- ResultSets:
  - ESigUserDefinedSignatureFields: Information about the custom electronic signature fields.

9.2.8 ESIG_GET_VALIDATE_FILE_FORM

Service that loads the validate file form.

Service Class: ESigHandler

Access Level: READ

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Additional Required Service Parameters

- ESigDocName: The content ID of the document.
- ESigContext: Set to `specific` to indicate the search is on a specific content item, or set to `global` to indicate the search is on all the items in system.

Additional Optional Service Parameters

- ESigDocTitle: Title of the electronic signature document.

9.2.9 ESIG_SET_CONFIG

Service that saves the configuration of custom electronic signatures fields.

Service Class: ESigHandler

Access Level: READ

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Additional Required Service Parameters

- fieldName_index: The custom electronic signature field name. If multiple fields are being configured, they are distinguished by index. index starts at 0 and increments.
- displayLabel_*index*: The custom field display label.
- dataType_*index*: The custom field data type.
- hasChoiceList_*index*: If set to `true`, the custom field is a choice list.
- listOfValues_*index*: If the custom field is a choice list, specifies the list of values.
- isCheckbox_*index*: If set to `true`, the custom field is a check box.
- isRequired_*index*: If set to `true`, the custom field is a required field.

Additional Optional Service Parameters

- esigIsAdminUpdate: If set to true, the configuration will be updated.

Results

- ResultSets:

  - ESigUserDefinedSignatureFields: Information about custom electronic signature fields.

9.2.10 ESIG_SIGN_CONTENT

Service that signs content electronically.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Additional Required Service Parameters

- ESigSignContentDocID: The ID of the document to sign.
- ESigSignContentPassword: The user's password for reauthentication.
- ESigSignContentDocRevisionID: The revision ID of the document to sign.
- ESigSignContentDocName: The content ID of the document to sign.
- $customField: The ID of the document to sign. This field is required if a custom field is required.

Additional Optional Service Parameters

- isESigUIAction: If set, the sign service checks for session and cookie information according to the electronic signature user reauthentication design, which redirects the user to the Oracle WebLogic Server login page for authentication.
- ESigCallOrigin: The call origin key that determines the redirected page on the native 11g user interface after the document is signed.

Results

- ResultSets:
  - DOC_INFO: Information about the document.

9.2.11 ESIG_VALIDATE_FILE

Service that searches for a local file in the repository by matching checksum against the list of available revisions for a specified content item. This service must pass the file content as part of a request to validate. A browser request passes the file content as part of form-data, whereas RIDC requires an instance of `oracle.stellent.ridc.model.TransferFile` class to send the file content to the server.

Service Class: ESigHandler

Access Level: None

Location: *IdcHomeDir*/components/ElectronicSignatures/resources/esig_service.htm

Additional Required Service Parameters

- ESigDocName: The content ID of the document.
- ESigValidationFile: The instance of `oracle.stellent.ridc.model.TransferFile` class.

Additional Optional Service Parameters

- ESigDocTitle: Title of the electronic signature document.
- ESigContext: To indicate the search is on a specific content item, set to `specific`.

Results

- ResultSets:
  - ESigValidateFileDocInfoResultSet: Information about the matching content item revision(s). `ESigValidateFileStatus` in the LocalData section indicates the status of the search. Possible values are `OK` and `NO_MATCH`.

9.2.12 ESIG_VALIDATE_FILE_GLOBAL

Service that searches for a local file in the repository by matching checksum against all the items in the system. This service must pass the file content as part of a request to validate. A browser request passes the file content as part of form-data, whereas RIDC requires an instance of `oracle.stellent.ridc.model.TransferFile` class to send the file content to the server.

Service Class: ESigHandler

Access Level: None

Location: *`IdcHomeDir`*`/components/ElectronicSignatures/resources/esig_service.htm`

Additional Required Service Parameters

- ESigValidationFile: The instance of `oracle.stellent.ridc.model.TransferFile` class.

Additional Optional Service Parameters

- ESigContext: To indicate the search is on a specific content item, set to `specific`.

Results

- ResultSets:
  - ESigGlobalValidationResultSe: Information about the matching content item(s). `ESigValidateFileStatus` in the LocalData section indicates the status of the search. Possible values are `OK` and `NO_MATCH`.

9.2.13 ESIG_WORKFLOW_SIGN_AND_APPROVE

Service that signs and approves content in a workflow.

Service Class: ESigHandler

Access Level: None

Location: *`IdcHomeDir`*`/components/ElectronicSignatures/resources/esig_service.htm`

Additional Required Service Parameters

- dID: The generated content item revision ID.
- ESigSignContentDocID: The ID of the document to sign.
- ESigSignContentPassword: The user's password for reauthentication.

- ESigSignContentDocRevisionID: The revision ID of the document to sign.
- ESigSignContentDocName: The content ID of the document to sign.
- $customField: The ID of the document to sign. This field is required if a custom field is required.

Additional Optional Service Parameters

- ESigWorkflowName: The name of the workflow.
- ESigWorkflowStepName: The name of the workflow step.
- ESigCallOrigin: The call origin key that determines the redirected page on the native 11g user interface after the document is signed.
- isESigUIAction: If set, the sign service checks for session and cookie information according to the electronic signature user reauthentication design, which redirects the user to the Oracle WebLogic Server login page for authentication.

10 Records Services

This chapter describes the services available when using and customizing Oracle WebCenter Content: Records services.

This chapter covers the following topics:

- About Records Services
- Records Services

10.1 About Records Services

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific Records services are listed within each individual service.

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

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.

10.2 Records Services

The following services are used in Records functions:

- ACTIVATE_SERVICE
- APPROVE_DELETE_SERVICE
- BROWSE_CATEGORY_FORM
- BROWSE_FOLDER_FORM
- BROWSE_SERIES_FORM
- CHECKIN_NEW_REVISION_SERVICE
- CHECKIN_SIMILAR_FORM

- CLEAR_FOLDER_CANCELLED_DATE
- CLEAR_FOLDER_EXPIRATION_DATE
- CLEAR_FOLDER_OSBOLETE_DATE
- CLEAR_FOLDER_RESCINDED_DATE
- CLEAR_FOLDER_REVIEW_DATE
- CLEAR_RECORD_CANCELLED_DATE
- CLEAR_RECORD_EXPIRATION_DATE
- CLEAR_RECORD_OBSOLETE_DATE
- CLOSE_FOLDER
- CREATE_FOLDER
- CREATE_FOLDER_FORM
- DELETE_ALL_BUT_LAST_N_REVISIONS_ SERVICE
- DELETE_FOLDER
- DELETE_REVISION_SERVICE
- EDIT_FOLDER
- EDIT_FOLDER_FORM
- FREEZE_FOLDER
- FREEZE_RECORD
- INFO_CATEGORY_FORM
- INFO_FOLDER_FORM
- INFO_FOLDER_LIFECYCLE
- INFO_FOLDER_METADATA_HISTORY
- INFO_FOLDER_REVIEW_HISTORY
- INFO_RECORD_LIFECYCLE
- INFO_RECORD_METADATA_HISTORY
- INFO_RECORD_REVIEW_HISTORY
- INFO_SERIES_FORM
- MARK_FOLDER_ACTIVATION_DATE
- MARK_FOLDER_CANCELLED_DATE
- MARK_FOLDER_EXPIRATION_DATE
- MARK_FOLDER_OBSOLETE_DATE
- MARK_FOLDER_RESCINDED_DATE
- MARK_FOLDER_REVIEW_DATE
- MARK_FOLDER_REVIEW_DATE_ RECURSIVE
- MARK_RECORD_CANCELLED_DATE
- MARK_RECORD_EXPIRATION_DATE
- MARK_RECORD_OBSOLETE_DATE

- MARK_RECORD_RESCINDED_DATE
- MARK_RECORD_REVIEW_DATE
- MOVE_FOLDER
- PREVIEW_RECORD_LIFECYCLE
- RMA_CLOSE_SERVICE
- RMA_CUTOFF_SERVICE
- RMA_DESTROY_SERVICE
- RMA_EXPORT_ARCHIVE_SERVICE
- RMA_EXPORT_SERVICE
- RMA_MARK_COMPLETED
- RMA_NO_ACTION_SERVICE
- RMA_OBSOLETE_SERVICE
- RMA_SCRUB_SERVICE
- RMA_SUPERSEDE_SERVICE
- UNCLOSE_FOLDER
- UNFREEZE_FOLDER
- UNFREEZE_RECORD

10.2.1 ACTIVATE_SERVICE

This service is used to set an Activate disposition action date.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action to be processed.

10.2.2 APPROVE_DELETE_SERVICE

This service is used to an Approve Delete disposition action date.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action to be processed.

10.2.3 BROWSE_CATEGORY_FORM

This service is used to retrieve a page used to browse the contents of a retention category in the retention schedule.

Additional Required Service Parameters

- dCategoryID: The unique identifier of the retention category to be retrieved.

10.2.4 BROWSE_FOLDER_FORM

This service is used to retrieve a page used to browse the contents of a record folder in the retention schedule.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be retrieved.

10.2.5 BROWSE_SERIES_FORM

This service is used to retrieve a page used to browse the contents of a series in the retention schedule.

Additional Required Service Parameters

- dSeriesID: The unique identifier of the series to be retrieved.

10.2.6 CHECKIN_NEW_REVISION_SERVICE

This service is used to check in the latest revision of a content item as a new revision. This is usually used to trigger a workflow.

10.2.7 CHECKIN_SIMILAR_FORM

This service is used to retrieve the Check In Similar page. This is a check-in page with metadata fields already filled in based on the previous content item checked in.

Additional Required Service Parameters

- dID: The unique content ID for the item already checked in.

10.2.8 CLEAR_FOLDER_CANCELLED_DATE

This service is used to clear the date for a canceled record folder.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.

10.2.9 CLEAR_FOLDER_EXPIRATION_DATE

This service is used to clear the expiration date for a canceled record folder.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.

10.2.10 CLEAR_FOLDER_OSBOLETE_DATE

This service is used to clear the date at which a record folder becomes obsolete.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.

10.2.11 CLEAR_FOLDER_RESCINDED_DATE

This service is used to clear the date at which a record folder is rescinded.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.

10.2.12 CLEAR_FOLDER_REVIEW_DATE

This service is used to clear the date at which a record folder was last reviewed.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.

10.2.13 CLEAR_RECORD_CANCELLED_DATE

This service is used to clear the date at which a content item was canceled.

Additional Required Service Parameters

- dID: The unique identifier of the content to be processed.

10.2.14 CLEAR_RECORD_EXPIRATION_DATE

This service is used to clear the date at which an item was expired.

Additional Required Service Parameters

- dID: The unique identifier of the item to be processed.

10.2.15 CLEAR_RECORD_OBSOLETE_DATE

This service is used to clear the date at which an item becomes obsolete.

Additional Required Service Parameters

- dID: The unique identifier of the item to be processed.

10.2.16 CLOSE_FOLDER

This service is used to close an existing record folder in the retention schedule. No further content items can be checked into the closed record 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 unique identifier of the record folder to be processed.

10.2.17 CREATE_FOLDER

This service is used to create a new record folder in the retention schedule. This service is usually called from the CREATE_FOLDER_FORM page.

Additional Required Service Parameters

- dFolderID: A unique identifier of the new record folder.
- dFolderName: A name for the new record folder.
- dSecurityGroup: The security group to be associated with the new record folder.
- dDocAuthor: The person creating the new record folder.
- dCategoryID: The unique identifier of the retention category to be associated with the record folder.

10.2.18 CREATE_FOLDER_FORM

This service is used to call CREATE_FOLDER to create a new record folder in the retention schedule.

10.2.19 DELETE_ALL_BUT_LAST_N_REVISIONS_ SERVICE

This service is used to process a Delete All But Last *N* Revisions disposition action. The administrator sets *n*, which is a variable.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action to be processed.

10.2.20 DELETE_FOLDER

This service is used to delete an existing record folder from the retention schedule.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.

10.2.21 DELETE_REVISION_SERVICE

This service is used to delete the latest revision of a content item.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action to be processed.

10.2.22 EDIT_FOLDER

This service is called by EDIT_FOLDER_FORM to update the (modified) properties of an existing record folder in the retention schedule.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be edited.
- dFolderName: The name of the record folder to be edited.
- dSecurityGroup: The security group associated with the record folder.
- dDocAuthor: The owner of the record folder.

10.2.23 EDIT_FOLDER_FORM

This service is used to call EDIT_FOLDER to edit a folder in the retention schedule.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be edited.

10.2.24 FREEZE_FOLDER

This service is used to freeze an existing record folder in the retention schedule. Freezing a folder pauses any processing of disposition rules associated with the folder until the UNFREEZE_FOLDER service is called.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be frozen.
- dCategoryID: The unique identifier of the retention category associated with the record folder.
- auditComments: The freeze name.
- auditComments2: The freeze reason.

10.2.25 FREEZE_RECORD

This service is used to freeze an existing content item. Freezing a content item pauses any processing of disposition rules associated with the content item until the UNFREEZE_RECORD service is called.

Additional Required Service Parameters

- dID: The content ID of the content item to be frozen.
- auditComments: The freeze name.
- auditComments2: The freeze reason.

10.2.26 INFO_CATEGORY_FORM

This service is used to retrieve the information page of a retention category.

Additional Required Service Parameters

- dCategoryID: The unique identifier of the retention category to be retrieved.

10.2.27 INFO_FOLDER_FORM

This service is used to retrieve the metadata information page and related content for a record folder in the retention schedule.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be retrieved.

10.2.28 INFO_FOLDER_LIFECYCLE

This service is used to retrieve the lifecycle information page for a record folder.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder.

10.2.29 INFO_FOLDER_METADATA_HISTORY

This service is used to retrieve the metadata history of a record folder.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder.

10.2.30 INFO_FOLDER_REVIEW_HISTORY

This service is used to retrieve the review history of a record folder.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder.

10.2.31 INFO_RECORD_LIFECYCLE

This service is used to retrieve the lifecycle information page for a content item.

Additional Required Service Parameters

- dID: The content ID of the item.
- dCategoryID: The unique identifier of the retention category associated with the item.

10.2.32 INFO_RECORD_METADATA_HISTORY

This service is used to retrieve the metadata history of a content item.

Additional Required Service Parameters

- dID: The content ID of the content item.

10.2.33 INFO_RECORD_REVIEW_HISTORY

This service is used to retrieve the review history of a content item.

Additional Required Service Parameters

- dID: The content ID of the content item.

10.2.34 INFO_SERIES_FORM

This service is used to retrieve the information page for a series in the retention schedule.

Additional Required Service Parameters

- dSeriesID: The unique identifier of the series.

10.2.35 MARK_FOLDER_ACTIVATION_DATE

This service is used to set the current date for when a record folder will be activated.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.
- currentDate: The date to be used for activation.

10.2.36 MARK_FOLDER_CANCELLED_DATE

This service is used to set the current date for when a record folder will be canceled.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.
- currentDate: The date to be used for cancellation.

10.2.37 MARK_FOLDER_EXPIRATION_DATE

This service is used to set the current date for the expiration date for a record folder.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.
- currentDate: The date to be used for expiration.

10.2.38 MARK_FOLDER_OBSOLETE_DATE

This service is used to set the current date for when a record folder becomes obsolete.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.
- currentDate: The date to be used as the obsolete date.

10.2.39 MARK_FOLDER_RESCINDED_DATE

This service is used to set the current date for when a record folder will be rescinded.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.
- currentDate: The date to be used as the rescinded date.

10.2.40 MARK_FOLDER_REVIEW_DATE

This service is used to set the current date for when a record folder will be reviewed.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.
- currentDate: The date to be used as the review date.

10.2.41 MARK_FOLDER_REVIEW_DATE_ RECURSIVE

This service is used to set the current date for when a record folder, including all of its child folders, will be reviewed.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.
- currentDate: The date to be used as the review date.

10.2.42 MARK_RECORD_CANCELLED_DATE

This service is used to set the current date for when a content item will be canceled.

Additional Required Service Parameters

- dID: The content ID of the content item to be processed.
- currentDate: The date to be used as the cancellation date.

10.2.43 MARK_RECORD_EXPIRATION_DATE

This service is used to set the current date for when an item will expire.

Additional Required Service Parameters

- dID: The content ID of the item to be processed.
- currentDate: The date to be used as the expiration date.

10.2.44 MARK_RECORD_OBSOLETE_DATE

This service is used to set the current date to be used to make an item obsolete.

Additional Required Service Parameters

- dID: The content ID of the item to be processed.
- currentDate: The date to be used as the obsolete date.

10.2.45 MARK_RECORD_RESCINDED_DATE

This service is used to set the current date for when an item will be rescinded.

Additional Required Service Parameters

- dID: The content ID of the item to be processed.
- currentDate: The date to be used as the rescinded date.

10.2.46 MARK_RECORD_REVIEW_DATE

This service is used to set the current date for when a content item will be reviewed.

Additional Required Service Parameters

- dID: The content ID of the content item to be processed.
- currentDate: The date to be used as the review date.

10.2.47 MOVE_FOLDER

This service is used to move a record folder from one location in the retention schedule to another.

Additional Required Service Parameters

- dFolderID: The folder ID of the folder to be processed.
- dCategoryID: The folder category.

10.2.48 PREVIEW_RECORD_LIFECYCLE

This service is used to retrieve the lifecycle preview of a content item.

Additional Required Service Parameters

- dID: The content ID of the content item to be processed.

10.2.49 RMA_CLOSE_SERVICE

This service is used to process a Close disposition action on a folder. This service calls the CLOSE_FOLDER service.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.50 RMA_CUTOFF_SERVICE

This service is used to process a Cutoff disposition action.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.51 RMA_DESTROY_SERVICE

This service is used to process a Destroy disposition action. All revisions of content or folders are destroyed unless other disposition rules take precedence.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.52 RMA_EXPORT_ARCHIVE_SERVICE

This service is used to add a Records archive to a Content Server archive.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.53 RMA_EXPORT_SERVICE

This service is used to create a zip archive.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.54 RMA_MARK_COMPLETED

This service is used to mark the completion of an action.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.55 RMA_NO_ACTION_SERVICE

This service is used to process a No Action disposition action.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.56 RMA_OBSOLETE_SERVICE

This service is used to process an Obsolete disposition action. This service calls MARK_FOLDER_OBSOLETE_DATE or MARK_RECORD_OBSOLETE_DATE.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.57 RMA_SCRUB_SERVICE

This service is used to securely delete record folders or content item so they cannot be recovered.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.58 RMA_SUPERSEDE_SERVICE

This service is used to process a Supersede disposition action.

Additional Required Service Parameters

- dDispositionID: The unique identifier of the disposition action.

10.2.59 UNCLOSE_FOLDER

This service is used to revoke the closed status of an existing record folder in the retention schedule. Unclosing a folder allows content to be checked into the closed record folder or its subfolders.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.

10.2.60 UNFREEZE_FOLDER

This service is used to unfreeze an existing record folder in the retention schedule. Unfreezing a folder releases the folder for processing of its associated disposition rules.

Additional Required Service Parameters

- dFolderID: The unique identifier of the record folder to be processed.
- auditComments: The unfreeze name.
- auditComments2: The unfreeze reason.

10.2.61 UNFREEZE_RECORD

This service is used to unfreeze an existing content item. Unfreezing releases the content item for processing of its associated disposition rules.

Additional Required Service Parameters

- dFolderID: The unique identifier of the content item to be processed.
- auditComments: The unfreeze name.
- auditComments2: The unfreeze reason.

11

Physical Content Management Services

This chapter describes the services available when using and customizing Physical Content Management (PCM), which is associated with Oracle WebCenter Content: Records.

This chapter covers the following topics:

- About Physical Content Management Services
- Physical Content Management Services

11.1 About Physical Content Management Services

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific PCM services are listed within each individual service.

For more information about PCM, see *Oracle Fusion Middleware Managing Oracle WebCenter Content*.

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

11.2 Physical Content Management Services

The following services are used in PCM functions:

- ADD_FUNCTION_BARCODE
- ADJUST_INVOICE
- ADJUST_INVOICE_FORM
- BARCODE_PROCESS
- BATCH_STORAGE_CREATION_FORM
- BLOCK_STORAGE
- BROWSE_CHARGE_INVOICES_FORM
- BROWSE_CHARGE_TRANSACTIONS_BY_INVOICE_FORM
- BROWSE_CHARGE_TRANSACTIONS_FORM

- BROWSE_CHARGE_TRANSACTIONS_WITH_NO_INVOICE_FORM
- BROWSE_RESERVATION_ITEMS_FORM
- BROWSE_STORAGE_FORM
- CHECKIN_EXTERNAL_ITEM
- CHECKOUT_EXTERNAL_ITEM
- CLEAN_UP_STORAGE
- CONFIG_CHARGE_BACKS
- CONFIG_CHARGE_BACKS_FORM
- CONFIGURE_CHARGE_BILLERS_FORM
- CONFIGURE_CHARGE_PAYMENT_TYPES_FORM
- CONFIGURE_CHARGE_TYPES_FORM
- CREATE_CHARGE_BILLER
- CREATE_CHARGE_BILLER_FORM
- CREATE_CHARGE_PAYMENT_TYPECREATE_CHARGE_PAYMENT_TYPE_FORM
- CREATE_CHARGE_TYPE
- CREATE_CHARGE_TYPE_FORM
- CREATE_CHARGE_TRANSACTION
- CREATE_CHARGE_TRANSACTION_FORM
- CREATE_CHARGE_TRANSACTION_MULTIPLE
- CREATE_EXTERNAL_ITEM
- CREATE_EXTERNAL_ITEM_FORM
- CREATE_EXTERNAL_ITEM_SIMILAR_FORM
- CREATE_MEDIA_TYPE
- CREATE_MEDIA_TYPE_FORM
- CREATE_OBJECT_TYPE
- CREATE_OBJECT_TYPE_FORM
- CREATE_RESERVATION
- CREATE_RESERVATION_FORM
- CREATE_STORAGE
- CREATE_STORAGE_FORM
- CREATE_STORAGE_TYPE
- CREATE_STORAGE_TYPE_FORM
- DELETE_CHARGE_BILLER
- DELETE_CHARGE_INVOICE
- DELETE_CHARGE_PAYMENT_TYPE
- DELETE_CHARGE_TRANSACTION

- DELETE_CHARGE_TYPE
- DELETE_COMPLETED_RESERVATIONS
- DELETE_EXTERNAL_ITEM
- DELETE_MEDIA_TYPE
- DELETE_OBJECT_TYPE
- DELETE_RESERVATION
- DELETE_RESERVATION_ITEM
- DELETE_STORAGE
- DELETE_STORAGE_TYPE
- EDIT_CHARGE_BILLER
- EDIT_CHARGE_BILLER_FORM
- EDIT_CHARGE_PAYMENT_TYPE
- EDIT_CHARGE_TYPE_FORM
- EDIT_EXTERNAL_ITEM
- EDIT_EXTERNAL_ITEM_FORM
- EDIT_MEDIA_TYPE
- EDIT_MEDIA_TYPE_FORM
- EDIT_OBJECT_TYPE
- EDIT_OBJECT_TYPE_FORM
- EDIT_OBJECT_TYPE_RELATIONSHIPS
- EDIT_OBJECT_TYPE_RELATIONSHIPS_FORM
- EDIT_RESERVATION
- EDIT_RESERVATION_FORM
- EDIT_RESERVATION_ITEM
- EDIT_RESERVATION_ITEM_FORM
- EDIT_RESERVATION_ITEM_STATUS
- EDIT_STORAGE
- EDIT_STORAGE_FORM
- EDIT_STORAGE_TYPE
- EDIT_STORAGE_TYPE_FORM
- GENERATE_CHARGE_INVOICE
- GET_EXPORT_INVOICES
- GET_EXTERNAL_ITEM_SEARCH_RESULTS
- GET_RELATED_CONTENT
- INFO_CHARGE_BILLER_FORM
- INFO_CHARGE_INVOICE_FORM
- INFO_CHARGE_PAYMENT_TYPE_FORM

- INFO_CHARGE_TRANSACTION_FORM
- INFO_CHARGE_TYPE_FORM
- INFO_EXTERNAL_ITEM_FORM
- INFO_MEDIA_TYPE_FORM
- INFO_OBJECT_TYPE_FORM
- INFO_RESERVATION_FORM
- INFO_RESERVATION_ITEM_FORM
- INFO_STORAGE_FORM
- INFO_STORAGE_TYPE_FORM
- MARK_INVOICE_PAID
- MARK_INVOICE_PAID_FORM
- MOVE_STORAGE
- NOTIFY_OVERDUE_USERS
- PROCESS_OVERDUE_REQUESTS
- PROCESS_STORAGE_SPACE_COUNTS
- RESERVE_STORAGE
- RUN_CHARGEBACK_REPORT
- RUN_CHARGEBACK_REPORT_MULTIPLE
- RUN_FUNCTION_BARCODE_LABEL
- RUN_RESERVATION_REPORT
- RUN_RESERVATION_SEARCH_RESULTS_REPORT
- RUN_STORAGE_BARCODE_LABEL
- RUN_STORAGE_REPORT
- RUN_USER_BARCODE_LABEL
- SCREEN_CHARGE_BACKS_ADVANCED_FORM
- SCREEN_CHARGE_BACKS_FORM
- SEARCH_EXTERNAL_ITEM_FORM
- UNRESERVE_STORAGE
- UPDATE_STORAGE_DEPTHS
- UPDATE_USERS_WITH_NO_BARCODES

11.2.1 ADD_FUNCTION_BARCODE

This service is used to create new function barcodes.

Location: *IdcHomeDir*`/components/PhysicalContentManager/resources/pcm_service.htm`

Additional Required Service Parameters

- dFunctionValue: The unique identifier of the function barcode to be created.

- dFunctionName: The name for of the function barcode to be created

11.2.2 ADJUST_INVOICE

This service is used to adjust an invoice.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dInvoiceID: The unique identifier of the invoice to be adjusted.
- dAmount: The amount to add/subtract to the invoice.

11.2.3 ADJUST_INVOICE_FORM

This service is used to retrieve a page used to adjust an invoice.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dInvoiceID: The unique identifier of the invoice to be adjusted.
- dInvoiceAmount: The current of the invoice.

11.2.4 BARCODE_PROCESS

This service is used to process the uploaded barcode file.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- barcodeFile: The path to the file to be processed.

11.2.5 BATCH_STORAGE_CREATION_FORM

This service is used to paint the Storage Creation Utility form.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.6 BLOCK_STORAGE

This service is used to block a storage location

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier of the storage location to be blocked

11.2.7 BROWSE_CHARGE_INVOICES_FORM

This service is used to retrieve a page used to browse transactions invoices.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.8 BROWSE_CHARGE_TRANSACTIONS_BY_INVOICE_FORM

This service is used to retrieve a page used to browse transactions belonging to the specified invoice.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dInvoiceID: The unique identifier of the invoice for transactions to be retrieved.

11.2.9 BROWSE_CHARGE_TRANSACTIONS_FORM

This service is used to retrieve a page used to browse charge transactions.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.10 BROWSE_CHARGE_TRANSACTIONS_WITH_NO_INVOICE_FORM

This service is used to retrieve a page used to browse transactions that have not been invoiced.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.11 BROWSE_RESERVATION_ITEMS_FORM

This service is used to retrieve a page used to browse items belonging to a the specified request.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the request for items to be retrieved.

11.2.12 BROWSE_STORAGE_FORM

This service is used to retrieve a page used to browse Storage Items.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier of the storage item to be retrieved. Set to 0 if you want the root.

11.2.13 CHECKIN_EXTERNAL_ITEM

This service is used to check in an external item in a request.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the request.
- dID: The unique identifier of the physical item to check in.

11.2.14 CHECKOUT_EXTERNAL_ITEM

This service is used to check out an external item in a request.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the request.
- dID: The unique identifier of the physical item to check out.

11.2.15 CLEAN_UP_STORAGE

This service is used to clean up the storage table location values.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

11.2.16 CONFIG_CHARGE_BACKS

This service is used save Chargeback settings.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.17 CONFIG_CHARGE_BACKS_FORM

This service is used to retrieve a page used to configure Chargeback settings.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.18 CONFIGURE_CHARGE_BILLERS_FORM

This service is used to retrieve a page used to browse customers.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.19 CONFIGURE_CHARGE_PAYMENT_TYPES_FORM

This service is used to retrieve a page used to browse payment types.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.20 CONFIGURE_CHARGE_TYPES_FORM

This service is used to retrieve a page used to browse charge types.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.21 CREATE_CHARGE_BILLER

This service is used to create a payment type customer.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dBillerTypeID: The unique identifier of the customer to be created.
- dBillerName: The name of the customer to be created.

11.2.22 CREATE_CHARGE_BILLER_FORM

This service is used to retrieve a page used to create a customer.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.23 CREATE_CHARGE_PAYMENT_TYPE

This service is used to create a payment type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dPaymentTypeID: The unique identifier of the payment type to be created.
- dPaymentTypeDesc: The name of the payment type to be created.

11.2.24 CREATE_CHARGE_PAYMENT_TYPE_FORM

This service is used to retrieve a page used to create a payment type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.25 CREATE_CHARGE_TYPE

This service is used to create a charge type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dChargeTypeID: The unique identifier of the charge type to be created.
- dChargeTypeDesc: The name of the charge type to be created.
- dChargeAction: The action assigned to the charge type.
- dChargeAmount: The amount for the action for the charge type.
- dExtObjectTypes: The object types associated with the charge type.

11.2.26 CREATE_CHARGE_TYPE_FORM

This service is used to retrieve a page used to create a charge type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.27 CREATE_CHARGE_TRANSACTION

This service is used to create a transaction.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dChargeTypeID: The unique identifier of the charge type to be used for the transaction.
- dChargeAction: The type of charge action for the transaction.
- dID, dDocName: The unique identifier or name of the charge external item if this is not a storage charge action.
- dChargeAmount: The amount of the transaction.

11.2.28 CREATE_CHARGE_TRANSACTION_FORM

This service is used to retrieve a page used to create a transaction.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.29 CREATE_CHARGE_TRANSACTION_MULTIPLE

This service is used to create a transaction.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dIDList: The comma-separated list of dIDs for the items to have transactions created for them.
- dChargeTypeID: The unique identifier of the charge type to be used for the transaction.
- dChargeAction: The type of charge action for the transaction.
- dChargeAmount: The amount of the transaction.

11.2.30 CREATE_EXTERNAL_ITEM

This service is used to retrieve to create physical items.

Location: *IdcHomeDir*/components/ExternalHelper/resources/erm_service.htm

Additional Required Service Parameters

- dSource: Source for which the external item is created; for example, Physical.
- dDocTitle: The title of the external item to be created.
- *Other*: Any field that is set to be required by the user.

11.2.31 CREATE_EXTERNAL_ITEM_FORM

This service is used to retrieve a page used to create physical items.

Location: *IdcHomeDir*/components/ExternalHelper/resources/erm_service.htm

Additional Required Service Parameters

- dSource: Source in which the external item is created; for example, Physical.

11.2.32 CREATE_EXTERNAL_ITEM_SIMILAR_FORM

This service is used to retrieve a page used to create a physical item similar to another external item.

Location: *IdcHomeDir*/components/ExternalHelper/resources/erm_service.htm

Additional Required Service Parameters

- dSource: Source in which the external item is created; for example, Physical.
- dID: The unique identifier of the external item from which to create a similar item.

11.2.33 CREATE_MEDIA_TYPE

This service is used to create a media type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dMediaTypeID: The unique identifier of the Media Type to be created.
- dMediaTypeName: The name of the Media Type item.

11.2.34 CREATE_MEDIA_TYPE_FORM

This service is used to retrieve a page used to create a media type.

11.2.35 CREATE_OBJECT_TYPE

This service is used to create an object type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectTypeID: The unique identifier of the Object Type to be created.
- dObjectTypeName: The name of the Object Type item.

11.2.36 CREATE_OBJECT_TYPE_FORM

This service is used to retrieve a page used to create an object type.

11.2.37 CREATE_RESERVATION

This service is used to create a reservation.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the Request to be created.
- dRequestName: Name of the Request.
- dRequestDate: Date of the Request.
- dRequestor: User for whom the request is made.
- dSecurityGroup: Security Group of the request.
- dTransferMethod: Transfer Method of the request.
- dRequestPriority: Delivery Priority for the request.
- dIDList: Comma-separated list of dID values for physical items to request.

11.2.38 CREATE_RESERVATION_FORM

This service is used to retrieve a page used to create a reservation.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dIDList: Comma-separated list of dID values for physical items to request.

11.2.39 CREATE_STORAGE

This service is used to create a storage location.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dParentID: The unique identifier of the parent storage location for the storage location to be created.
- dObjectName: The name of the storage location to be created.
- dStorageType: The unique identifier of the location type assigned to the storage location.

11.2.40 CREATE_STORAGE_FORM

This service is used to retrieve a page used to create a storage location.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dParentID: The unique identifier of the parent storage location for the storage location to be created.

11.2.41 CREATE_STORAGE_TYPE

This service is used to create a location type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dStorageTypeID: The unique identifier of the location type to be created.
- dStorageTypeName: The name of the location type to be created.
- dOrder: The order the location type is located within the location types.
- dStorageImage: The name of the image to be associated with the location type

11.2.42 CREATE_STORAGE_TYPE_FORM

This service is used to retrieve a page used to create a location type.

11.2.43 DELETE_CHARGE_BILLER

This service is used to delete a customer.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dBillerID: The unique identifier of the customer to be deleted.

11.2.44 DELETE_CHARGE_INVOICE

This service is used to delete an invoice.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dInvoiceID: The unique identifier of the invoice to be deleted.

11.2.45 DELETE_CHARGE_PAYMENT_TYPE

This service is used to delete a payment type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dPaymentTypeID: The unique identifier of the payment type to be deleted.

11.2.46 DELETE_CHARGE_TRANSACTION

This service is used to delete a transaction.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dTransactionID: The unique identifier of the transaction to be deleted.

11.2.47 DELETE_CHARGE_TYPE

This service is used to delete a charge type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dChargeTypeID: The unique identifier of the charge type to be deleted.

11.2.48 DELETE_COMPLETED_RESERVATIONS

This service is used to remove completed reservations from the system.

11.2.49 DELETE_EXTERNAL_ITEM

This service is used to delete physical items.

Location: *IdcHomeDir*/components/ExternalHelper/resources/erm_service.htm

Additional Required Service Parameters

- dSource: Source from which the external item is deleted; for example, `Physical`.
- dID: The unique identifier of the external item.

11.2.50 DELETE_MEDIA_TYPE

This service is used to delete a media type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dMediaTypeID: The unique identifier of the Media Type to be deleted.

11.2.51 DELETE_OBJECT_TYPE

This service is used to delete a object type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectTypeID: The unique identifier of the Object Type to be deleted.

11.2.52 DELETE_RESERVATION

This service is used to delete a Reservation.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the Reservation to be deleted.

11.2.53 DELETE_RESERVATION_ITEM

This service is used to delete a request item from a request.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique request identifier of the requested item to be deleted.
- dID: The unique identifier of the requested item to be deleted.

11.2.54 DELETE_STORAGE

This service is used to delete a storage item.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier of the Storage item to be deleted.

11.2.55 DELETE_STORAGE_TYPE

This service is used to delete a location type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dStorageTypeID: The unique identifier of the Location Type to be deleted.

11.2.56 EDIT_CHARGE_BILLER

This service is used to edit a customer.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dBillerID: The unique identifier of the customer to be created.
- dBillerName: The name of the customer to be created.

11.2.57 EDIT_CHARGE_BILLER_FORM

This service is used to retrieve a page used to edit a customer.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.58 EDIT_CHARGE_PAYMENT_TYPE

This service is used to edit a payment type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dPaymentTypeID: The unique identifier of the payment type to be created.
- dPaymentTypeDesc: The name of the payment type to be created.

11.2.59 EDIT_CHARGE_TYPE_FORM

This service is used to retrieve a page used to edit a charge type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.60 EDIT_EXTERNAL_ITEM

This service is used to edit physical items.

Location: *IdcHomeDir*/components/ExternalHelper/resources/erm_service.htm

Additional Required Service Parameters

- dSource: Source for which the external item is edited; for example, `Physical`.
- dID: The unique identifier of the external item.

11.2.61 EDIT_EXTERNAL_ITEM_FORM

This service is used to retrieve a page used to edit a physical item.

Location: *IdcHomeDir*/components/ExternalHelper/resources/erm_service.htm

Additional Required Service Parameters

- dSource: Source in which the external item is edited; for example, `Physical`.
- dID: The unique identifier of the external item.

11.2.62 EDIT_MEDIA_TYPE

This service is used to edit a media type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dMediaTypeID: The unique identifier of the Media Type to be edited.
- dMediaTypeName: The name of the Media Type item.

11.2.63 EDIT_MEDIA_TYPE_FORM

This service is used to retrieve a page used to edit a media type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dMediaTypeID: The unique identifier of the Media Type to be edited.

11.2.64 EDIT_OBJECT_TYPE

This service is used to edit an object type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectTypeID: The unique identifier of the Object Type to be created.
- dObjectTypeName: The name of the Object Type item.

11.2.65 EDIT_OBJECT_TYPE_FORM

This service is used to retrieve a page used to edit an object type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectTypeID: The unique identifier of the Object Type to be edited.

11.2.66 EDIT_OBJECT_TYPE_RELATIONSHIPS

This service is used to edit object type relationships.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectTypeID: The unique identifier of the Object Type to be created.

- SelectListValues: List of Object Types that the Object Type can contain.

11.2.67 EDIT_OBJECT_TYPE_RELATIONSHIPS_FORM

This service is used to retrieve a page used to edit object type relationships.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectTypeID: The unique identifier of the Object Type Relationship to be edited.

11.2.68 EDIT_RESERVATION

This service is used to edit the request data.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the Request to be edited.
- dRequestName: Name of the Request.
- dRequestDate: Date of the Request.
- dRequestor: User for whom the request is made.
- dSecurityGroup: Security Group of the request.
- dTransferMethod: Transfer Method of the request.
- dRequestPriority: Delivery Priority for the request.

11.2.69 EDIT_RESERVATION_FORM

This service is used to retrieve a page used to edit a request.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the Request to be edited.

11.2.70 EDIT_RESERVATION_ITEM

This service is used to edit the requested item data.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the request to be edited.
- dID: The unique identifier of the requested item to be edited.
- dRequestDate: Date of the Request.
- dRequestStatus: Current status or the requested item.

11.2.71 EDIT_RESERVATION_ITEM_FORM

This service is used to retrieve a page used to edit a specific requested item.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the Request to be edited.
- dID: The unique identifier of the Physical Item to be edited.

11.2.72 EDIT_RESERVATION_ITEM_STATUS

This service is used to edit the requested items status.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the request to be edited.
- dID: The unique identifier of the requested item to be edited.
- dRequestStatusNew: Current status or the requested item.

11.2.73 EDIT_STORAGE

This service is used to edit a storage location.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dParentID: The unique identifier of the parent storage location for the storage location to be created.
- dObjectID: The unique identifier of the storage location to be edit.
- dObjectName: The name of the storage location to be created.
- dStorageType: The unique identifier of the location type assigned to the storage location.

11.2.74 EDIT_STORAGE_FORM

This service is used to retrieve a page used to create a storage location.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier of the storage location to be edit.

11.2.75 EDIT_STORAGE_TYPE

This service is used to edit a location type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dStorageTypeID: The unique identifier of the location type to be edited.
- dStorageTypeName: The name of the location type to be edited.
- dOrder: The order the location type is located within the location types.
- dStorageImage: The name of the image to be associated with the location type.

11.2.76 EDIT_STORAGE_TYPE_FORM

This service is used to retrieve a page used to edit a location type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dStorageTypeID: The unique identifier of the location type to be edited.

11.2.77 GENERATE_CHARGE_INVOICE

This service is used to generate invoices.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.78 GET_EXPORT_INVOICES

This service is used to retrieve a export file for invoices.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.79 GET_EXTERNAL_ITEM_SEARCH_RESULTS

This service retrieves a page used to search physical items.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dSource: source being used for the search (for example, "Physical").
- QueryText: the text used for the search.
- ErmSearchTable: the source table name. This should be EXTERNAL_SOURCE for a Physical source.
- SearchEngineName: the search engine to use. Default is DATABASE.
- SearchQueryFormat: the search query format to use. Default is UNIVERSAL.

11.2.80 GET_RELATED_CONTENT

This service retrieves a page used to show Related Links for the specified content.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dSource: source being used for the search (for example, "Physical").
- dID: the unique identifier of the external item.
- dLinkTypeID: the unique identifier for the related content type link.

11.2.81 INFO_CHARGE_BILLER_FORM

This service is used to retrieve a page used to display information for a specified storage type customer.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dBillerID: The unique identifier of the storage type to be created.

11.2.82 INFO_CHARGE_INVOICE_FORM

This service is used to retrieve a page used to display information for a specified Invoice.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dInvoiceID: The unique identifier of the invoice to be retrieved.

11.2.83 INFO_CHARGE_PAYMENT_TYPE_FORM

This service is used to retrieve a page used to display information for a specified payment type.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dPaymentTypeID: The unique identifier of the payment type to be retrieved.

11.2.84 INFO_CHARGE_TRANSACTION_FORM

This service is used to retrieve a page used to display information for a specified Transaction.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dTransactionID: The unique identifier of the transaction to be retrieved.

11.2.85 INFO_CHARGE_TYPE_FORM

This service is used to retrieve a page used to display information for a specified charge type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dChargeTypeID: The unique identifier of the charge type to be retrieved.

11.2.86 INFO_EXTERNAL_ITEM_FORM

This service is used to retrieve an information page for a physical item.

Location: *IdcHomeDir*/components/ExternalHelper/resources/erm_service.htm

Additional Required Service Parameters

- dSource: Source of the external item; for example, Physical.
- dID: The unique identifier of the external item.

11.2.87 INFO_MEDIA_TYPE_FORM

This service is used to retrieve a page used to display information for a specified media type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dMediaTypeID: The unique identifier of the media type to be retrieved.

11.2.88 INFO_OBJECT_TYPE_FORM

This service is used to retrieve a page used to display information for a specified object type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectTypeID: The unique identifier of the object type to be retrieved.

11.2.89 INFO_RESERVATION_FORM

This service is used to retrieve a page used to display information for a specified request.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier of the request to be retrieved.
- dID: The unique identifier of the physical item to be retrieved.

11.2.90 INFO_RESERVATION_ITEM_FORM

This service is used to retrieve a page used to display information for the specified requested physical item.

Additional Required Service Parameters

- dRequestID: The unique identifier of the request to be retrieved.

11.2.91 INFO_STORAGE_FORM

This service is used to retrieve a page used to display information for a specified storage item.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier of the storage item to be retrieved.

11.2.92 INFO_STORAGE_TYPE_FORM

This service is used to retrieve a page used to display information for a specified storage type.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dStorageTypeID: The unique identifier of the storage type to be retrieved.

11.2.93 MARK_INVOICE_PAID

This service is used to mark an invoice paid.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dInvoiceID: The unique identifier of the invoice to be marked paid.
- dPaymentType: The unique identifier of the payment type.
- dPaymentNumber: The number of the credit card or check, or a miscellaneous number.
- dExpirationDate: The expiration date of the credit card if using one or just blank.
- dAmount: The amount paid.

11.2.94 MARK_INVOICE_PAID_FORM

This service is used to retrieve a page used to mark an invoice paid.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- dInvoiceID: The unique identifier of the invoice so be marked paid.
- dInvoiceAmount: The current amount of the invoice.

11.2.95 MOVE_STORAGE

This service is used to move one storage item into another.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier of the Storage Item to be move.
- newParentID: The unique identifier of the Storage Item to move to (Parent Storage Item).

11.2.96 NOTIFY_OVERDUE_USERS

This service is used to notify users of overdue requests.

11.2.97 PROCESS_OVERDUE_REQUESTS

This service is used to update requests that have become overdue.

11.2.98 PROCESS_STORAGE_SPACE_COUNTS

This service is used to update the storage counts, that is, it updates the number of spaces used at each levels.

11.2.99 RESERVE_STORAGE

This service is used to reserve a storage location.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier of the storage location to be reserved.
- dRequestor: The User ID of the person reserving the storage.

11.2.100 RUN_CHARGEBACK_REPORT

This service is used to run a invoice report.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- rpReportTemplate: The Report Template to use for the report.

- rpReportType: The report type of the report to run.

11.2.101 RUN_CHARGEBACK_REPORT_MULTIPLE

This service is used to run an invoice report.

Location:
IdcHomeDir/components/PhysicalContentManager/resources/chargebacks_service.htm

Additional Required Service Parameters

- rpReportTemplate: The Report Template to use for the report.
- rpReportType: The report type of the report to run.
- selectedDocList: The list of comma-separated invoice IDs to report on.

11.2.102 RUN_FUNCTION_BARCODE_LABEL

This service is used to run a function barcode report.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

11.2.103 RUN_RESERVATION_REPORT

This service is used to run a reservation report.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dRequestID: The unique identifier for the request to run a report for.

11.2.104 RUN_RESERVATION_SEARCH_RESULTS_REPORT

This service is used to run the reservation search results report.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- QueryText: The query to run to get the reservations.

11.2.105 RUN_STORAGE_BARCODE_LABEL

This service is used to run a storage label report.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier for the storage item for which the report is run.

11.2.106 RUN_STORAGE_REPORT

This service is used to run a storage report

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier for the storage item for which the report is run.

11.2.107 RUN_USER_BARCODE_LABEL

This service is used to run a user report.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dSelectUser: The unique identifier of the user for which to run a report.
- dPrintAllUsers: Flag to set if you want to run for all users not a specific one.

11.2.108 SCREEN_CHARGE_BACKS_ADVANCED_FORM

This service is used to retrieve a page used to generate invoices using advanced page.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.109 SCREEN_CHARGE_BACKS_FORM

This service is used to retrieve a page used to generate invoices

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/chargebacks_service.htm

11.2.110 SEARCH_EXTERNAL_ITEM_FORM

This service is used to retrieve a page used to search physical items.

Location: *IdcHomeDir*/components/ExternalHelper/resources/erm_service.htm

Additional Required Service Parameters

- dSource: Source in which the external item is searched; for example, `Physical`.

11.2.111 UNRESERVE_STORAGE

This service is used to unreserve a storage location.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

Additional Required Service Parameters

- dObjectID: The unique identifier of the storage location to be unreserved.

11.2.112 UPDATE_STORAGE_DEPTHS

This service is used to update the dDepth value for storage items.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

11.2.113 UPDATE_USERS_WITH_NO_BARCODES

This service is used to update users that do not have barcode values.

Location: *IdcHomeDir*/components/PhysicalContentManager/resources/pcm_service.htm

12

Extended User Attributes Services

This chapter describes the Oracle WebCenter Content services available when using and customizing the Extended User Attributes component.

This chapter covers the following topics:

- About Extended User Attributes Services
- Extended User Attributes Services

12.1 About Extended User Attributes Services

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific Extended User Attributes services are listed within each individual service.

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

12.2 Extended User Attributes Services

The following Extended User Attributes component services are installed when the Content Server instance is installed:

- ADD_EXTENDED_USER_ATTRIBUTES
- EDIT_EXTENDED_USER_ATTRIBUTES
- DELETE_EXTENDED_USER_ATTRIBUTES
- DELETE_EXTENDED_ATTRIBUTES_BY_APPLICATION
- DELETE_ALL_EXTENDED_ATTRIBUTES_FOR_USER
- DELETE_ALL_EXTENDED_ATTRIBUTES_FOR_APPLICATION
- QUERY_EXTENDED_USER_ATTRIBUTES
- QUERY_EXTENDED_ATTRIBUTE_MAPPINGS
- EC_SET_PROPERTY
- EC_DELETE_PROPERTY

- EC_GET_PROPERTY
- EC_GET_PROPERTY_BY_KEY
- SET_DEFAULT_ATTRIBUTES
- DELETE_DEFAULT_ATTRIBUTES
- GET_DEFAULT_ATTRIBUTES
- SET_EXTENDED_ATTRIBUTE_MAPPINGS
- DELETED_EXTENDED_ATTRIBUTE_MAPPINGS

The following services are extended to provide support for the Extended User Attributes component:

- ADD_USER
- EDIT_USER
- DELETE_USER
- QUERY_USER_ATTRIBUTES

12.2.1 ADD_EXTENDED_USER_ATTRIBUTES

Service that adds extended attributes to a user. The user does not have to exist before using the service. If this service is asked to add extended attributes which already exist for the user, the service modifies the previous entries.

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- dName: The user name.
- ExtUserAttribInfo Resultset: A result set containing extended user attribution information.

Optional Service Parameters

- CheckUserMustExist: If set to true, a limited check is performed to verify that the user exists before extended attributes are added. This check only works for local users; it is *not* recommended to set this value for typical service calls.

Example

The following example from `add_user_attr.txt` illustrates the use of this service:

```
# Add a security group and an account to the user's extended attributes
# with different applications
@Properties LocalData
IdcService=ADD_EXTENDED_USER_ATTRIBUTES
dName=jsmith
@end
@ResultSet ExtUserAttribInfo
3
dUserName
dApplication
AttributeInfo
jsmith
appl
```

```
account,abc/def,15
jsmith
app2
role,contributor,15
@end
```

12.2.2 EDIT_EXTENDED_USER_ATTRIBUTES

Service that edits extended user attributes for a user. The user does not have to exist before using the service. All previously extended attributes are replaced by the new extended attributes provided.

Location:

IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- dName: The user name.
- ExtUserAttribInfo ResultSet: A result set containing extended user attribution information.

Optional Service Parameters

- CheckUserMustExist: If set to *true*, a limited check is performed to verify that the user exists before extended attributes are added. This check only works for local users; it is *not* recommended to set this value for typical service calls.

Example

The following example from `edit_user_attr.txt` illustrates the use of this service:

```
# Change the user to have all permissions with #all
@Properties LocalData
IdcService=EDIT_EXTENDED_USER_ATTRIBUTES
dName=jsmith
@end
@ResultSet ExtUserAttribInfo
3
dUserName
dApplication
AttributeInfo
jsmith
appl
account,\#all,15
@end
```

12.2.3 DELETE_EXTENDED_USER_ATTRIBUTES

Service that deletes specific extended user attributes for a user.

Location:

IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- dName: The user name.

- ExtUserAttribInfo ResultSet: A result set containing extended user attribution information.
- allowMissingAttributes: If set to *true*, the service suppresses errors when trying to delete attributes that do not exist (the default is *false*).

Optional Service Parameters

- CheckUserMustExist: If set to *true*, a limited check is performed to verify that the user exists before extended attributes are added. This check only works for local users; it is *not* recommended to set this value for typical service calls.

Example

```
# Delete the #all account from extended attributes for jsmith
@Properties LocalData
IdcService=DELETE_EXTENDED_USER_ATTRIBUTES
dName=jsmith
@end
@ResultSet ExtUserAttribInfo
3
dUserName
dApplication
AttributeInfo
jsmith
appl
account,\#all,15
@end
```

12.2.4 DELETE_EXTENDED_ATTRIBUTES_BY_APPLICATION

Service that deletes all extended attributes for a particular role/account for a specified application.

Location:

IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- dAttributeType: The type of attribute to delete.
- dAttributeName: The name of the attribute to delete.
- dApplication: The application from which to delete the attribute.

Example

The following example from `delete_attr_by_app.txt` illustrates the use of this service:

```
# Delete the xyz account from extended attributes for appl
@Properties LocalData
IdcService=DELETE_EXTENDED_ATTRIBUTES_BY_APPLICATION
dAttributeName=xyz
dAttributeType=account
dApplication=appl
@end
```

12.2.5 DELETE_ALL_EXTENDED_ATTRIBUTES_FOR_USER

Service that deletes all extended attributes for a user.

Additional Required Service Parameters

- dName: The user name.

Optional Service Parameters

- CheckUserMustExist: If set to *true*, a limited check is performed to verify that the user exists before extended attributes are added. This check only works for local users; it is *not* recommended to set this value for typical service calls.

Example

The following example from `delete_user_attr_all.txt` illustrates the use of this service:

```
# Add roles
@Properties LocalData
IdcService=DELETE_ALL_EXTENDED_ATTRIBUTES_FOR_USER
dName=jsmith
@end
```

12.2.6 DELETE_ALL_EXTENDED_ATTRIBUTES_FOR_APPLICATION

Service that deletes all extended users attributes for an application.

Location:
IdcHomeDir`/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm`

Additional Required Service Parameters

- dApplication: The application.

Example

The following example from `del_all_attr_app.txt` illustrates the use of this service:

```
# Delete all extended attribute information for appl
@Properties LocalData
IdcService=DELETE_ALL_EXTENDED_ATTRIBUTES_FOR_APPLICATION
dApplication=appl
@end
```

12.2.7 QUERY_EXTENDED_USER_ATTRIBUTES

Service that places all of a user's extended attributes into a result set in the binder. All of the data is put into the ExtUserAttribInfo ResultSet.

Location:
IdcHomeDir`/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm`

Additional Required Service Parameters

- dName: The user name.

Optional Service Parameters

- dApplication: The application name for the attribute.
- dAttributeType: The type of attribute.
- dAttributeName: The name of the attribute.

Example

Adding one or more of the optional parameters enables greater flexibility in narrowing the query. The following example from `query_user_attr_ext.txt` illustrates the use of this service:

```
# Query the extended attributes of a user
@Properties LocalData
IdcService=QUERY_EXTENDED_USER_ATTRIBUTES
dName=jsmith
@end
```

12.2.8 QUERY_EXTENDED_ATTRIBUTE_MAPPINGS

Service that places all mapped extended attributes from a particular attribute into a result set in the binder. All of the data is put into the ExtUserAttribInfo ResultSet.

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- dMappedAttributeName: The name of the attribute to be mapped.

Optional Service Parameters

- dApplication: The application name for the attribute.
- dAttributeType: The type of attribute.
- dAttributeName: The name of the attribute.

Example

Adding one or more of the optional parameters enables greater flexibility in narrowing the query. The following example from `query_attr_map_ext.txt` illustrates the use of this service:

```
<?hda version="11gR1-dev" jcharset=Cp1252 encoding=iso-8859-1?>

# Query the attributes mapped from role 'guest'
@Properties LocalData
IdcService=QUERY_EXTENDED_ATTRIBUTE_MAPPINGS
dMappedAttributeName=guest
@end
```

12.2.9 EC_SET_PROPERTY

Service that sets an extended configuration property. It can be used to either add a new property or modify an existing property.

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- dECPropType: The type of property.
- dECPropKey: A key (generally relating to dECPropType).
- dECPropValue: The property value.

Optional Service Parameters

- dECPropSubKey: (Recommended) A second key; for example, the key can be a feature of what is defined by dECPropType and dEDPropKey.

Example

The following example from `set_property.txt` illustrates the use of this service:

```
# Set an extended configuration property
@Properties LocalData
IdcService=EC_SET_PROPERTY
dECPropType=account
dECPropKey=#abc
dECPropSubKey=URL
dECPropValue=http://www.example.com
@end
```

12.2.10 EC_DELETE_PROPERTY

Service that deletes an extended configuration property. It is important that the exact key be specified for deletion. This service does not automatically delete all subkeys under a key.

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- dECPropType: The type of property.
- dECPropKey: A key (generally relating to dECPropType).

Optional Service Parameters

- dECPropSubKey: (Recommended) A second key; for example, the key can be a feature of what is defined by dECPropType and dEDPropKey.

Example

The following example from `del_property.txt` illustrates the use of this service:

```
# Delete an extended configuration property
@Properties LocalData
IdcService=EC_DELETE_PROPERTY
dECPropType=account
dECPropKey=#abc
dECPropSubKey=URL
@end
```

12.2.11 EC_GET_PROPERTY

Service that returns a specified extended configuration property in the ResultSet PROPERTY_LIST. It is important that the exact key be specified; if you want to search based on just the key (not subkey) use EC_GET_PROPERTY_BY_KEY.

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- dECPropType: The type of property.
- dECPropKey: A key (generally relating to dECPropType).

Optional Service Parameters

- dECPropSubKey: (Recommended) A second key; for example, the key can be a feature of what is defined by dECPropType and dEDPropKey.

Example

The following example from `get_property.txt` illustrates the use of this service:

```
# Retrieve an extended configuration property
@Properties LocalData
IdcService=EC_GET_PROPERTY
dECPropType=account
dECPropKey=#abc
dECPropSubKey=URL
@end
```

12.2.12 EC_GET_PROPERTY_BY_KEY

Service that returns a specified extended configuration property in the ResultSet PROPERTY_LIST.

Location:
IdcHomeDir`/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm`

Additional Required Service Parameters

- dECPropType: The type of property.
- dECPropKey: A key (generally relating to dECPropType).

Example

The following example from `get_property_by_key.txt` illustrates the use of this service:

```
# Retrieve an extended configuration property by key - gets all
# accounts of key #abc (regardless of subkey)
@Properties LocalData
IdcService=EC_GET_PROPERTY_BY_KEY
dECPropType=account
dECPropKey=#abc
@end
```

12.2.13 SET_DEFAULT_ATTRIBUTES

Service that sets some default attributes that will always be applied to a user. The type is `defaults` and the key is `userattributes`, so these do not need to be provided.

Location:
IdcHomeDir`/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm`

Additional Required Service Parameters

- dDefAttribs: The default attributes in the form of a comma-separated entry consisting of three items.

Optional Service Parameters

- dECPropSubKey: A second key; the default attributes will load for all users whether or not this key is specified.
- IsSecureDefaultAttribute: When set to 1 and used with the databinder when using the default web services ( ) then it registers defaults that only apply to logged in users. (By default, default attributes apply to all users, including *anonymous*.)

Example

The following example from `set_defs.txt` illustrates the use of this service:

```
# Sets some default attributes for all users
@Properties LocalData
IdcService=SET_DEFAULT_ATTRIBUTES
dECPropSubKey=extprops
dDefAttribs=account,#abc,15
@end
```

12.2.14 DELETE_DEFAULT_ATTRIBUTES

Service that deletes some default attributes that will always be applied to a user.

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Optional Service Parameters

- dECPropSubKey: A second key. If this parameter is not specified, the service will only remove the default attributes specified without a subkey; it will not remove other attributes.
- IsSecureDefaultAttribute: If set to 1 and used with the databinder when using the default web services ( ) then it deletes defaults that only apply to logged in users. (By default, default attributes apply to all users, including *anonymous*.)

Example

The following example from `del_defs.txt` illustrates the use of this service:

```
# Deletes some default attributes for all users
@Properties LocalData
IdcService=DELETE_DEFAULT_ATTRIBUTES
dECPropSubKey=extprops
@end
```

12.2.15 GET_DEFAULT_ATTRIBUTES

This service returns the default attributes that are applied to all users in the ResultSet DEFAULT_ATTRIBUTES.

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Optional Service Parameters

- dECPropSubKey: If specified, it narrows the search for the default attributes to only the one matching that subkey.

- IsSecureDefaultAttribute: If set to 1 and used with the databinder when using the default web services ( ) then it retrieves defaults that only apply to logged in users. (By default, default attributes apply to all users, including *anonymous*.)

Example

The following example from `get_defs.txt` illustrates the use of this service:

```
# Returns all default attributes for all users
@Properties LocalData
IdcService=GET_DEFAULT_ATTRIBUTES
@end
```

12.2.16 SET_EXTENDED_ATTRIBUTE_MAPPINGS

Service that sets mappings from one user attribute to another user attribute. Only mappings from roles to other attributes is supported (mapping does not work when the source is an account).

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- ExtUserAttribInfo: A result set containing extended user attribute information.

Example

The following example from `set_attr_mapping.txt` illustrates the use of this service:

```
<?hda version="11gR1-dev" jcharset=Cp1252 encoding=iso-8859-1?>

# Add the 'contributor' role to all users with the
# 'guest' role, and then consequently give all users
# with the 'contributor' role read/write access to
# account 'abc'. (This does mean that users with
# the 'guest' role ultimately end up with access to 'abc').
@Properties LocalData
IdcService=SET_EXTENDED_ATTRIBUTE_MAPPINGS
@end
@ResultSet ExtUserAttribInfo
3
dAttributeName
dApplication
guest
appl
role,contributor,15
contributor
app2
account,abc,3
@end
```

12.2.17 DELETED_EXTENDED_ATTRIBUTE_MAPPINGS

Service that deletes mappings from one user attribute to another user attribute.

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Additional Required Service Parameters

- ExtUserAttribInfo: A result set containing extended user attribute information.

Optional Service Parameters

- allowMissingAttributes: If set to *true*, the service suppresses errors when trying to delete attributes that do not exist. The default is *false*.

Example

The following example from `del_attr_mapping.txt` illustrates the use of this service:

```
<?hda version="11gR1-dev" jcharset=Cp1252 encoding=iso-8859-1?>

# Delete two attribute mappings
@Properties LocalData
IdcService=DELETE_EXTENDED_ATTRIBUTE_MAPPINGS
@end
@ResultSet ExtUserAttribInfo
3
dAttributeName
dApplication
AttributeInfo
guest
appl
role,contributor,15
contributor
app2
account,abc,3
@end
```

12.2.18 ADD_USER

The service has been altered to add extended user attributes at the same time the user is added. For additional information see Section 4.14.4, "ADD_USER."

Location:
IdcHomeDir`/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm`

Optional Service Parameter

- ExtUserAttribInfo ResultSet: A result set containing extended user attribution information. If it is not included, no extended attributes are added.

Example

The following example from `add_user.txt` illustrates the use of this service:

```
# Add user with extended security data
@Properties LocalData
IdcService=ADD_USER
dName=jsmith
dUserAuthType=Local
dFullName=Jennifer Smith
dPassword=password
dEmail=email@example.com
@end
@ResultSet UserAttribInfo
2
dUserName
AttributeInfo
```

```
jsmith
role,guest,15
@end
@ResultSet ExtUserAttribInfo
3
dUserName
dApplication
AttributeInfo
jsmith
appl
account,abc,15
@end
```

12.2.19 EDIT_USER

This service has been altered to allow editing of extended user attributes at the same time. The old values are completely replaced by the new ones. For additional information, see Section 4.14.4, "ADD_USER."

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Optional Service Parameters

- ExtUserAttribInfo ResultSet: A result set containing extended user attribution information. If it is not included, the extended attributes are not replaced.

Example

The following example from `edit_user.txt` illustrates the use of this service:

```
# Edit user with extended security data
@Properties LocalData
IdcService=EDIT_USER
dName=jsmith
dUserAuthType=Local
dFullName=Justin Smith
dPassword=12345
dEmail=real@real.com
@end
@ResultSet UserAttribInfo
2
dUserName
AttributeInfo
jsmith
role,guest,15
@end
@ResultSet ExtUserAttribInfo
3
dUserName
dApplication
AttributeInfo
jsmith
appl
account,abc,7,account,xyz,15
@end
```

12.2.20 DELETE_USER

This service has been altered to delete a user's extended attributes if they exist. For additional information, see Section 4.14.10, "DELETE_USER."

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Example

The following example from `delete_user.txt` illustrates the use of this service:

```
# Delete user with extended security data
@Properties LocalData
IdcService=DELETE_USER
dName=jsmith
@end
```

12.2.21 QUERY_USER_ATTRIBUTES

This service acts as usual however the attributes it returns are contingent on the value of the getAllAttributes parameter. If the parameter is passed as *true*, this service returns a merger of both regular and extended user attributes. For additional information, see Section 4.14.25, "QUERY_USER_ATTRIBUTES."

Location:
IdcHomeDir/components/ExtendedUserAttributes/resources/extendeduserattributes_services.htm

Example

The following example from `query_user_attr.txt` illustrates the use of this service:

```
# The normal QUERY_USER_ATTRIBUTES, except it gathers all
# attributes now
@Properties LocalData
IdcService=QUERY_USER_ATTRIBUTES
dName=jsmith
getAllAttributes=true
@end
```

13

Folios Services

This chapter describes the Oracle WebCenter Content services available when using and customizing the Folios component.

This chapter covers the following topics:

- About Folios Services
- Folio Services

13.1 About Folios Services

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific Folios services are listed within each individual service.

For more information about Folios, see *Oracle Fusion Middleware Managing Oracle WebCenter Content*.

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

13.2 Folio Services

The following services are described in this section:

- LOAD_FOLIO_NODE
- UPDATE_FOLIO
- CHECKIN_NEW_FOLIO
- CREATE_FOLIO_SNAPSHOT
- LOCK_FOLIO
- UNLOCK_FOLIO
- CREATE_FOLIO_RENDITION
- GENERATE_GUIDS

13.2.1 LOAD_FOLIO_NODE

Service that displays the contents of a folio node by returning the nodes, slots and links that are the immediate children of the requested node. By default this service is not recursive; if called recursively, this service returns a complete folio.

Location: *IdcHomeDir*/components/ContentContentFolios/resources/cpd_service.htm

Additional Required Service Parameters

- dID: The content ID of the folio.
- RevisionSelectionMethod: Tells the Content Server instance whether to use retrieved information based on the latest revision, which could be in an unreleased state, or the latest released version. The value can be: *Specific*, *Latest* (which could be unreleased), or *Latest Released*.

Optional Service Parameter

- parentNodeId: Retrieves details, but not the properties, of the contents of the specified node within a folio. If this parameter is omitted, by default the service retrieves the root node and its properties.

Example 1

```
http://myserver/idc/idcplg?IdcService=LOAD_FOLIO_NODE&dID=68095
&RevisionSelectionMethod=Specific
```

Example 2

```
http://myserver/idc/idcplg?IdcService=LOAD_FOLIO_NODE&dID=68095
&RevisionSelectionMethod=Specific&parentNodeId=E176-3B00-1D2A-A592-E0CF
```

13.2.2 UPDATE_FOLIO

Service that adds one or more content items to a folio and modifies a folio. Multiple changes can be requested as part of a single service call.

Prerequisite: Before an item can be added to a folio, the item must be checked in to the Content Server instance using the standard CHECKIN_NEW or CHECKIN_UNIVERSAL services.

Location: *IdcHomeDir*/components/ContentContentFolios/resources/cpd_service.htm

Additional Required Service Parameters

- NumChanges: Specifies the number of changes or updates passed as part of the service call. For example, to pass two changes, `NumChanges=2`.
- change(*n*-1): Specifies the actual change to be made to the folio. The default is one change, which is indicated by change() or change0. If there are multiple changes to the folio, as indicated by NumChanges, then a parameter must be passed for each change. For example, if NumChanges=3, then change(), change1, and change2.

> **Note:** For each change(*n*-1) parameter, you must use a change_data(*n*-1) parameter. For example, if you specify a `change1` parameter you must also specify a `change_data1` parameter.

The value for the change(*n*-1) parameter must be in the following format: *dID*:*Operation*:*TargetGUID*:*ParentGUID*

- dID: The dID of the folio to be updated.
- Operation: One of the following choices:
  - removeNode: remove a node
  - addNode: add a node
  - addItem: add a slot
  - addContent: add an actual file
  - removeContent: remove a content item
  - modifyAttribute: modify a property
  - modifyTemplateAttribute: modify a template property
- TargetGUID: The GUID of the node, slot, or content item in the folio to be modified.
- ParentGUID: The parent GUID of the mode, slot, or content item to be modified. This is relevant for only some of the operations, such as addNode.

- change_data(*n*-1): Specifies the values for the operation specified in the change(*n*-1) parameter. The format for this parameter is *name:value* pairs separated by commas. A carat (^) can be used as a separator if a colon (:) is required in any of the names or values (*name:name^value*).

  When adding content these properties must be specified:

  - dDocAuthor
  - dDocName
  - dDocName_encoded
  - dDocTitle
  - dDocType
  - dFormat
  - dID
  - docURL
  - docURL_encoded
  - dOriginalName
  - dRevLabel

  Property information can be retrieved using a DOC_INFO service call against the item you want to add to a folio. You must provide the information for the dDocName_encoded, docURL, and docURL_encoded properties. You can specify custom properties that you add to the folio definition as XCST or XCSD data.

Example 1

To add a new content item to an existing content folio, you must define two changes. The first change operation is to specify `addItem`, which will add a slot. The second change operation is to specify `addContent`, which will insert the document into the slot. For both changes, the change_data parameter must specify the content properties. The xcsd^name and xcst^name designation for the properties also must be specified.

```
IdcService=UPDATE_FOLIO
NumChanges=2
dDocName=OCS_223948
RevisionSelectionMethod=Latest
change0=8611:addItem:4952-6EBE-1275-0A59-0D19:F5C4-A3D0-E671-D4CF-9C28
change_data0=xcsd^dFormat:application/vnd.ms-excel,id:4952-6EBE-1275-0A59-0D19,
xcsd^dID:18251,xcsd^dDocTitle:Next Year Goals,xcsd^dDocType:Document,xcsd^dRendition1:T,
xcsd^dRevLabel:3,xcsd^dOriginalName:Next Year Goals.xls,xcsd^dDocAuthor:weblogic,
xcsd^docURL:/cs/groups/Secure/documents/Document/OCS_158654.xls,xcsd^dDocName_encoded:OCS_158654,
xcsd^docURL_encoded:%2Fcs%2Fgroups%2FSecure%2Fdocuments%2FDocument%2FEOCS_158654.xls,
xcst^name:Next Year Goals,xcst^description:
change1=8611:addContent:4952-6EBE-1275-0A59-0D19
change_data1=xcsd^dFormat:application/vnd.ms-excel,id:4952-6EBE-1275-0A59-0D19,
xcsd^dDocName:OCS_158654,xcsd^dID:18251,xcsd^dDocTitle:Next Year Goals,xcsd^dDocType:Document,
xcsd^dRendition1:T,xcsd^dRevLabel:3,xcsd^dOriginalName:Next Year Goals.xls,
xcsd^dDocAuthor:weblogic,xcsd^docURL:/cs/groups/Secure/documents/Document/OCS_158654.xls,
xcsd^dDocName_encoded:OCS_158654,
xcsd^docURL_encoded:%2Fcs%2Fgroups%2FSecure%2Fdocuments%2FDocument%2FEOCS_158654.xls,
xcst^name:Next Year Goals,xcst^description:
```

Example 2

To add a new content item to an existing folio as part of RIDC code:

```
// Start a new client connection to UCM
IdcClientManager manager = new IdcClientManager();
// build a client that will communicate using the intradoc protocol
IdcClient idcClient = manager.createClient("idc://localhost:4444");

// create a trusted user connection to UCM
IdcContext userContext = new IdcContext("weblogic");

// create the binder for the set of GUIDs needed later on
DataBinder guidBinder = idcClient.createBinder();
// populate the binder with the parameters
guidBinder.putLocal("IdcService", "GENERATE_GUIDS");

// send the request and get the response back from local data and result set
ServiceResponse guidResponse = idcClient.sendRequest(userContext, guidBinder);
DataBinder folderData = guidResponse.getResponseAsBinder ();
DataResultSet guidResults = folderData.getResultSet ("GUID_SET");

// put the set of GUIDs in an array
String guids[] = new String[25];
int n = 0;
for (DataObject dataObject : guidResults.getRows ()) {
      guids[n] = dataObject.get("id");
}

// create the binder to get the content information of the document to add
DataBinder docInfoBinder = idcClient.createBinder();
// populate the binder with the parameters
docInfoBinder.putLocal("IdcService", "DOC_INFO_BY_NAME");
docInfoBinder.putLocal("dDocName", "OCS_158654");

// send the request and get the response back from local data and result set
ServiceResponse infoResponse = idcClient.sendRequest(userContext, docInfoBinder);
DataBinder infoData = infoResponse.getResponseAsBinder ();
DataResultSet docInfoResults = infoData.getResultSet ("DOC_INFO");

// create the binder to get the content information of the folio to update
```

```
DataBinder folioInfoBinder = idcClient.createBinder();
 // populate the binder with the parameters
folioInfoBinder.putLocal("IdcService", "DOC_INFO_BY_NAME");
folioInfoBinder.putLocal("dDocName", "OCS_223948);

// send the request and get the response back from local data and result set
ServiceResponse folioInfoResponse = idcClient.sendRequest(userContext, folioInfoBinder);
DataBinder folioInfoData = folioInfoResponse.getResponseAsBinder ();
DataResultSet folioInfoResults = folioInfoData.getResultSet ("DOC_INFO");
String foliodID = folioInfoData.getLocal("dID");

// build the change data string for the content
String newItemStr = "";
for (DataObject dataObject : docInfoResults.getRows ()) {
// build the path to the document
String urlStr = "/cs/groups/" + dataObject.get ("dSecurityGroup") + "/documents/" + dataObject.get
("dDocType") + "/" + dataObject.get ("dDocName") + "." + dataObject.get ("dWebExtension");
// assemble the string with the parameters needed for the content folio service change string
newItemStr = "xcsd^dFormat:" + dataObject.get ("dFormat") + ",";
newItemStr += "id:" + guids[0] + ",";
newItemStr += "xcsd^dDocName:" + dataObject.get ("dDocName") + ",";
newItemStr += "xcsd^dID:" + dataObject.get ("dID") + ",";
newItemStr += "xcsd^dDocTitle:" + dataObject.get ("dDocTitle") + ",";
newItemStr += "xcsd^dDocType:" + dataObject.get ("dDocType") + ",";
newItemStr += "xcsd^dRendition1:" + dataObject.get ("dRendition1") + ",";
newItemStr += "xcsd^dRevLabel:" + dataObject.get ("dRevLabel") + ",";
newItemStr += "xcsd^dOriginalName:" + dataObject.get ("dOriginalName") + ",";
newItemStr += "xcsd^dDocAuthor:" + dataObject.get ("dDocAuthor") + ",";
newItemStr += "xcsd^docURL:" + urlStr + ",";
newItemStr += "xcsd^dDocName_encoded:" + java.net.URLEncoder.encode(dataObject.get ("dDocName"),
"UTF-8") + ",";
newItemStr += "xcsd^docURL_encoded:" + java.net.URLEncoder.encode(urlStr, "UTF-8") + ",";
newItemStr += "xcst^name:" + dataObject.get ("dDocTitle") + ",";
newItemStr += "xcst^description:";
}

// create the binder to get the folio root GUID
DataBinder rootBinder = idcClient.createBinder();
// populate the binder with the parameters
rootBinder.putLocal("IdcService", "LOAD_FOLIO_NODE");
rootBinder.putLocal("dDocName", "folioName");
rootBinder.putLocal("RevisionSelectionMethod", "Latest");

// send the request and get the response back from local data
ServiceResponse rootGUIDResponse = idcClient.sendRequest(userContext, rootBinder);
DataBinder rootGUIDData = rootGUIDResponse.getResponseAsBinder ();
String rootGUID = rootGUIDData.getLocal("RootNode");

// assemble the first change parameter for the service.  This will tell the content folio service
// to add a slot
String change0 = foliodID + ":addItem:" + guids[0] + ":" + rootGUID;

// assemble the second change parameter for the service.  This will tell the content folio service
// to add a content item to the new slot
String change1 = foliodID + ":addContent:" + guids[0];

// create the binder to run the UPDATE_FOLIO service
DataBinder folioBinder = idcClient.createBinder();
// populate the binder with the parameters
folioBinder.putLocal("IdcService", "UPDATE_FOLIO");
```

```
folioBinder.putLocal("dDocName", folioName)
folioBinder.putLocal("RevisionSelectionMethod", "Latest");
folioBinder.putLocal("NumChanges", "2");
folioBinder.putLocal("change0", change0);
folioBinder.putLocal("change_data0", newItemStr);
folioBinder.putLocal("change1", change1);
folioBinder.putLocal("change_data1", newItemStr);

// send the request and get the response back from local data
ServiceResponse folioResponse = idcClient.sendRequest(userContext, folioBinder);
DataBinder folioData = folioResponse.getResponseAsBinder ();
```

13.2.3 CHECKIN_NEW_FOLIO

Service that creates and checks in a new folio with a root node.

Location: *IdcHomeDir*/components/ContentContentFolios/resources/cpd_service.htm

Optional Service Parameters

- This service can use standard metadata parameters, such as dDocName, dDocTitle, and so forth. Any other metadata required by parameters used in this service must also be specified.
- If auto-generation of content ID is set on the server and you do not specify the dDocName parameter, then a new content ID is automatically generated.
- The dID of the new folio is listed in the LocalData section of the service response.

Example

```
http://myserver/xpedio/icdplg?IdcServer=CHECKIN_NEW_FOLIO
&dDocName=myFolioTest&dDocTitle=MyFolioAPITest2&dDocType=TestData
&dDocAuthor=authorname&dSecurityGropu=Public&IsJava=1
```

13.2.4 CREATE_FOLIO_SNAPSHOT

Service that takes a version of a folio and creates a locked version and a new version.

Location: *IdcHomeDir*/components/ContentContentFolios/resources/cpd_service.htm

Additional Required Service Parameters

- dID: The content ID of the folio.
- RevisionSelectionMethod: Tells the Content Server instance whether to use retrieved information based on the latest revision, which could be in an unreleased state, or the latest released version. The value can be: Specific, Latest (which could be unreleased), or Latest Released.

Example

```
http://myserver/idc/idcplg?IdcService=CREATE_FOLIO_SNAPSHOT&dID=68095
&RevisionSelectionMethod=Specific
```

13.2.5 LOCK_FOLIO

Service that locks a folio so that it can not be edited, only viewed. No further modifications are allowed on this content item.

Location: *IdcHomeDir*/components/ContentContentFolios/resources/cpd_service.htm

Additional Required Service Parameters

- dID: The content ID of the folio.
- RevisionSelectionMethod: Tells the Content Server instance whether to use retrieve information based on the latest revision, which could be in an unreleased state, or the latest released version. The value can be: *Specific*, *Latest* (which could be unreleased), or *Latest Released*.

Example

```
http://myserver/idc/idcplg?IdcService=LOCK_FOLIO&dID=68095&RevisionSelectionMethod=Specific
```

13.2.6 UNLOCK_FOLIO

Service that creates a new version of a locked folio, where the new version of the folio can be edited.

Location: *IdcHomeDir*/components/ContentContentFolios/resources/cpd_service.htm

Additional Required Service Parameters

- dID: The content ID of the folio.
- RevisionSelectionMethod: Tells the Content Server instance whether to use retrieve information based on the latest revision, which could be in an unreleased state, or the latest released version. The value can be: *Specific*, *Latest* (which could be unreleased), or *Latest Released*.

Example

```
http://myserver/idc/idcplg?IdcService=UNLOCK_FOLIO&dID=68095&RevisionSelectionMethod=Specific
```

13.2.7 CREATE_FOLIO_RENDITION

Service that returns a specific rendition of a folio revision to a browser. A copy of the folio is retrieved without performing a checkout.

Location: *IdcHomeDir*/components/ContentContentFolios/resources/cpd_service.htm

Additional Required Service Parameters

- dID: The content ID of the folio.
- RenderID: The type of rendition: PDF, ZIP, XML. A custom rendition can be used.

Example

```
http://myserver/idc/idcplg?IdcService=CREATE_FOLIO_RENDITION&dID=68095&RenditionID=PDF
```

13.2.8 GENERATE_GUIDS

Service that generates and returns 25 group user IDs for use.

Location: *IdcHomeDir*/components/ContentContentFolios/resources/cpd_service.htm

Example

```
http://myserver/expedio/idcplg?IdcService=GENERATE_GUIDS&IsJava=1
```

14

Link Manager Services

This chapter describes the Oracle WebCenter Content services available when using and customizing the Link Manager component.

This chapter covers the following topics:

- About Link Manager Services
- Link Manager Services

14.1 About Link Manager Services

The Link Manager component evaluates, filters, and parses the URL links of indexed content items before extracting them for storage in a database table (ManagedLinks). After the ManagedLinks table is populated with the extracted URL links, the component references this table to generate link search results, lists of link references for the Content Information page, and the resource information for the Link Info page.

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

The locations for specific Link Manager services are listed within each individual service.

For more information about Link Manager, see *Oracle Fusion Middleware Managing Oracle WebCenter Content*.

14.2 Link Manager Services

The following services are used in Link Manager functions:

- ABORT_LINKS_ACTIVITY
- ADD_MANAGED_DOCLINKS
- DELETE_MANAGED_DOCLINKS
- GET_LINK_INFO
- GET_LINKS_ADMIN_PAGE
- LK_GET_SEARCH_RESULTS
- RECOMPUTE_MANAGED_LINKS
- REFRESH_MANAGED_DOCLINKS

- REFRESH_MANAGED_LINKS
- REFRESH_REFS_MANAGED_LINKS

14.2.1 ABORT_LINKS_ACTIVITY

This service is used to stop the current activity that Link Manager is performing.

You must be logged on to the Content Server instance with administrator credentials to execute this service. This service is also available from the Link Manager Admin page.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

```
IdcService=ABORT_LINKS_ACTIVITY
```

14.2.2 ADD_MANAGED_DOCLINKS

This service is used to extract HTML links out of the given input file. It is executed automatically when content items are indexed

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

```
IdcService=ADD_MANAGED_DOCLINKS
```

14.2.3 DELETE_MANAGED_DOCLINKS

This service is used to remove the managed links in the database for a given content item. It is executed automatically when content items are deleted.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

```
IdcService=DELETE_MANAGED_DOCLINKS
```

14.2.4 GET_LINK_INFO

This service is used to display link information for a given content item.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Additional Required Service Parameters

- dLkClassid: The unique class ID assigned to a link in the ManagedLinks table.

Results

- ResultSets:
  - DOC_INFO (standard metadata fields)
  - LinkInfo:
    * dLkGUID: Unique ID assigned to a row in the ManagedLinks table.

* dLkClassId: Unique ID assigned to the link in the ManagedLinks table.
* dDocName: Doc name where the link was discovered.
* dLkType: Type of link, which is one of the following: external, unknown, ss_link_rel, ss_link_page, ss_link_node, ss_link_page, ss_link_node, ss_link_internal, ss_link_page, ss_link_page, ss_link_node, ss_link_abs.
* dLkOriginalUrl: Original URL of the link.
* dLkContainerID: Site Studio siteId where the link was found.
* dLkResource: Value of one of the following: dID, dDocName, ssDocName, nodeId (depending on dLkResourceType value, this is null for external links).
* dLkResourceAlias: The docName or nodeLabel.
* dLdResourceType: One of the following: id, doc, node.
* dLkHasAlias: (Boolean) Indicates whether an alias exists.
* dLkState: Y or N. Y means a link references a checked in content item and is valid. N means a link references a deleted content item and is invalid.
* dLkCreateTs: Timestamp when the link was created.
* dLkUpdateTs: Timestamp when the link was last updated.
* dLkCycle: Flag used to tell LinkManager who is working on this link (A - Refresher, D - Decoder, null - none)

Example

```
IdcService=GET_LINK_INFO
dLkClassId=652
```

Sample return information. The beginning of the ResultSet shows how many fields exist (14 in this example) along with the names of the columns, the data types (string, date, integer, and so on), and the size of the fields in bytes. After these entries the values for these fields are listed in the next 14 rows. If there are multiple entries in a result set, another set of 14 rows is listed, and so on.

```
@ResultSet LinkInfo
14
dLkGUID 6 30
dLkClassId 3 38
dDocName 6 30
dLkType 6 20
dLkOriginalUrl 6 1024
dLkContainerId 6 50
dLkResource 6 50
dLkResourceAlias 6 50
dLkResourceType 6 50
dLkHasAlias 6 1
dLkState 6 1
dLkCreateTs 5 20
dLkUpdateTs 5 20
dLkCycle 6 1
```

14.2.5 GET_LINKS_ADMIN_PAGE

This service is used to obtain the status of the current Link Manager activity. Used by the Link Manager admin page to display status.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

```
IdcService=GET_LINKS_ADMIN_PAGE
```

14.2.6 LK_GET_SEARCH_RESULTS

This service is used to search for links to and from content items.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

This example opens a form that when submitted will execute the LK_GET_SEARCH_RESULTS service.

```
IdcService=LK_SEARCH_FORM
```

14.2.7 RECOMPUTE_MANAGED_LINKS

This service is used to extract URL links of indexed documents; evaluate, filter, and parse the URLs according to a pattern engine; and then store the results in a database table.

You must be logged on to the Content Server instance with administrator credentials to execute this service. This service is also available from the Link Manager Admin page.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

```
IdcService=RECOMPUTE_MANAGED_LINKS
```

14.2.8 REFRESH_MANAGED_DOCLINKS

This service is used to refresh the links for a specific document. A refresh does not involve extracting the links again. It iterates through the links as they are listed in the database and determines their current status.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

```
IdcService=REFRESH_MANAGED_DOCLINKS
```

14.2.9 REFRESH_MANAGED_LINKS

This service is used to refresh the managed links as they exist in the MANAGEDLINKS database table. The managed links are not re-extracted from the content items like in RECOMPUTE_MANAGED_LINKS.

You must be logged on to the Content Server instance with administrator credentials to execute this service. This service is also available from the Link Manager Admin page.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

```
IdcService=REFRESH_MANAGED_LINKS
```

14.2.10 REFRESH_REFS_MANAGED_LINKS

This service is used to perform a recount on the content references as they exist in the MANAGEDLINKS table. Each content item is referenced in the table and will be tallied and stored in the reference table.

You must be logged on to the Content Server instance with administrator credentials to execute this service. This service is also available from the Link Manager Admin page.

Location: *IdcHomeDir*/components/LinkManager/resources/linkmanager_service.htm

Example

```
IdcService=REFRESH_REFS_MANAGED_LINKS
```

15

Virtual Content Repository Services

This chapter describes the Oracle WebCenter Content services available when using and customizing Virtual Content Repository (VCR).

This chapter covers the following topics:

- About Virtual Content Repository Services
- Virtual Content Repository Services

15.1 About Virtual Content Repository Services

Services are used by applications that work with the VCR service provider interface, which is used by the Oracle WebLogic Portal as a way to expose content from different repositories. The Content Server application supports VCR functionality.

> **Note:** VCR services are supported only when used with Contribution Folders (provided by the Folders_g component). VCR services are **not** supported with Folders (provided by the FrameworkFolders component). For information about Contribution Folders services, see Chapter 7.

Information about what is a WebCenter Content service and how services can be used is provided in Section 2, "Using Services." Information about basic services structure, attributes, actions, and a service example is provided in Section 3, "Customizing Services." You should be familiar with this information before customizing current services or creating new services

15.2 Virtual Content Repository Services

The following VCR services are described in this section:

- VCR_FOLDER_INFO
- VCR_GET_CONTENT_TYPE
- VCR_GET_CONTENT_TYPES
- VCR_GET_DOCUMENT
- VCR_GET_DOCUMENT_BY_NAME

15.2.1 VCR_FOLDER_INFO

Service that is used to retrieve information about a Folders_g folder having the content and in the format required by the VCR service provider interface.

Service Class: DocService

Additional Required Service Parameters

One of the following parameters to identify the Folders_g folder:

- dCollectionID: The folder ID of the folder containing the target content.
- dCollectionPath: The path to the folder containing the target content.

Results

- ResultSets:
  - VcrNode
    - createdBy
    - createdDate
    - hasChildren
    - modifiedBy
    - name
    - objectClass
    - parentID
    - path
  - VcrPropertyValues
    - dChildManipulation
    - dCollectionCreator
    - dCollectionEnabled
    - dCollectionGUID
    - dCollectionID
    - dCollectionInherit
    - dCollectionMark
    - dCollectionModifier
    - dCollectionName
    - dCollectionOwner
    - dCollectionPath
    - dCollectionQueries
    - dCollectionType
    - dCreateDate
    - dDependent
    - dDocAccount
    - dDocAuthor

* dDocName
* dDocTitle
* dDocType
* dInDate
* dLastModifiedDate
* dOutDate
* dParentCollectionID
* dPromptForMetadata
* dReleaseDate
* dRevLabel
* dSecurityGroup
* xClbraAliasList
* xClbraRoleList
* xClbraUserList
* xCollectionID
* xComments
* xDontShowInListsForWebsites
* xEmailCC
* xEmailFrom
* xEmailSubject
* xEmailTo
* xForceFolderSecurity
* xHidden
* xIdcProfile
* xInhibitUpdate
* xProfileTrigger
* xPublicationDate
* xReadOnly
* xReceivedDate
* xRegionDefinition
* xt
* xTrashDeleteDate
* xTrashDeleteLoc
* xTrashDeleteName
* xTrashDeleter
* xtestdec2
* xtestdec3

        * xtTestField1
        * xtTestField2
        * xtestINTEGER
        * xWebsiteObjectType
        * xWebsites
        * xWebsiteSection

15.2.2 VCR_GET_CONTENT_TYPE

Service that requests detailed information about a particular VCR content type.

Service Class: Service

- vcrContentType: The ID of the content type; also referred to as the content type's name.

Results

- ResultSets:
  - VcrContentType
  - VcrProperties
  - VcrPropertyChoices (FIELDNAME)

15.2.3 VCR_GET_CONTENT_TYPES

Service that requests a list of all VCR content types defined by the system.

Service Class: Service

Results

- ResultSets: VcrContentTypes

15.2.4 VCR_GET_DOCUMENT

Service that retrieves content item information for a specific revision of a content item.

This service is almost identical to the DOC_INFO service, however, the data returned is modified to be easier to work with in a VCR context.

Service Class: Service

Additional Required Service Parameters

- dID: The generated content item revision ID.

Results

- ResultSets:
  - VcrPropertyValues
    * all standard DOC_INFO ResultSet columns returned by the DOC_INFO service
    * idcPrimaryFile
    * idcRenditions

- REVISION_HISTORY
  - * dDocName
  - * dFormat
  - * dID
  - * dInDate
  - * dOutDate
  - * dProcessingState
  - * dRevLabel
  - * dRevisionID for all non-deleted revisions
  - * dStatus

15.2.5 VCR_GET_DOCUMENT_BY_NAME

Service that retrieves information about the latest revision of a content item based on the content ID (the dDocName) as a parameter.

This service is almost identical to DOC_INFO_BY_NAME service, however, the data returned is modified to be easier to work with in a VCR context.

Service Class: Service

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*, a dID can be used instead of a dDocName to point to a specific revision.

Results

- ResultSets:
  - VcrPropertyValues
    - * all standard DOC_INFO ResultSet columns returned by the DOC_INFO service
    - * idcPrimaryFile
    - * idcRenditions
  - REVISION_HISTORY
    - * dDocName
    - * dFormat
    - * dID
    - * dInDate
    - * dOutDate
    - * dProcessingState
    - * dRevisionID for all non-deleted revisions

* dRevLabel
* dStatus

16

Security Services

This chapter describes the services available for securing the search-related services from SQL injections in Oracle WebCenter Content.

This chapter covers the following topics:

- About Security Services
- Security Services

16.1 About Security Services

This security component is enabled by default and can be invoked by an Admin user in Oracle WebCenter Content.

The locations for specific Security service are listed within each individual service.

16.2 Security Services

The following services can be used when the Security component is enabled in Oracle WebCenter Content:

- ASC_GET_SECURITY_CONFIGURATIONS
- ASC_UPDATE_SECURITY_CONFIGURATIONS

16.2.1 ASC_GET_SECURITY_CONFIGURATIONS

This service gets the current security configuration that is set in the WebCenter Content application.

Location:
IdcHomeDir/components/OracleAdvancedSecurityConfig/resources/securityconfig_service.idoc

Results

- ResultSets:
  - CoreQueryTextSecurityConfig: Information about the Core QueryText Security Configuration.
    - dName: Name of the field.
    - dValue: Value of the field.
  - FolderQueryTextSecurityconfig: Information about the FrameworkFolders QueryText Security Configuration.

        * dName: Name of the field.
        * dValue: Value of the field.

16.2.2 ASC_UPDATE_SECURITY_CONFIGURATIONS

This service allows the Admin user to update the security configuration.

Location:
IdcHomeDir/components/OracleAdvancedSecurityConfig/resources/securityconfig_service.idoc

Additional Required Service Parameters

- IsCoreQueryTextSecurityEdit: Set this flag to *true* in binder local data for each service request to update Core Security Configuration section. If not set, no update will be sent to this Core Security Configuration section.
- CORE_QUERYTEXT_SECURITY_ENABLED: To enable/disable validation of QueryText in GET_SEARCH_RESULTS service.
- CORE_CUSTOM_TABLES: Semicolon(;) separated list of tables whose columns will be allowed in QueryText.
- CORE_CUSTOM_FIELDS: Semicolon(;) separated list of field names to be allowed in QueryText.
- IsFfQueryTextSecurityEdit: Set this flag to *true* in binder local data for each service request to update FrameworkFolders Security Configuration Section. If not set, no update to this FrameworkFolders Security Configuration Section.
- FF_QUERYTEXT_SECURITY_ENABLED: To enable/disable validation of QueryText in FrameworkFolders.
- FF_CUSTOM_TABLES: Semicolon(;) separated list of tables whose columns will be allowed in QueryText.
- FF_CUSTOM_FIELDS: Semicolon(;) separated list of field names to be allowed in QueryText.

Results

- ResultSets:
  - CoreQueryTextSecurityConfig: Information about the Core QueryText Security Configuration.
    * dName: Name of the field.
    * dValue: Value of the field.
  - FolderQueryTextSecurityconfig: Information about the FrameworkFolders QueryText Security Configuration.
    * dName: Name of the field.
    * dValue: Value of the field.

A

Actions

This appendix lists alphabetically and describes the actions used by Oracle WebCenter Content 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
- A
- B
- C
- D
- E
- F
- G
- H
- I
- L
- M
- N
- P
- Q
- R
- S
- T
- U
- V

A.1 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.
- For details about actions, their parameters, and control masks, see Section 3.1.3, "Actions."

A.2 A

addAliases

Adds the aliases and passes *IworkflowAlias* as a parameter. Called as a Java method by ADD_WORKFLOWALIASES.

addCollaboration

Called as a Java method by ADD_COLLABORATION.

addContext

Called as a Java method by ADD_WEB_APP.

addFiles:

Checks in the content item by name. Called as a Java method by these services:

- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB
- INSERT_NATIVE
- INSERT_NEW
- REPLACE_METAFILE_SUB
- UPDATE_BYREV
- WORKFLOW_CHECKIN_SUB

addOrEditDocMetaData

Called as a Java method by ADD_METADEF and EDIT_METADEF.

addOrEditProvider

Adds the new provider. Called as a Java method by ADD_EDIT_PROVIDER

addOrEditSchemaTable

Called as a Java method by ADDOREDIT_SCHEMA_TABLE.

addProblemReport

Adds the problem report to the database. Called as a Java method by ADD_PROBLEMREPORT.

addRegisteredUserAttribute

Retrieves the user database profile information. Called as a Java method by REGISTER_USER.

addSchemaRelation

Called as a Java method by ADD_SCHEMA_RELATION.

addSchemaView

Called as a Java method by ADD_SCHEMA_VIEW.

addSubscription

Adds the subscription and passes *Isubscription* as a parameter. Called as a Java method by SUBSCRIBE.

addTemplate

Called as a Java method by ADD_WF_TEMPLATE.

addUserAttributes

Adds/updates the user security attributes and passes *IuserSecurityAttribute* as a parameter. Called as a Java method by ADD_USER and EDIT_USER.

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.

addWorkflow

Adds the workflow. Called as a Java method by ADD_WORKFLOW.

addWorkflowScript

Called as a Java method by ADD_WORKFLOW_SCRIPT.

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.

addWorkflowStepScript

Called as a Java method by ADD_WORKFLOWSTEP.

addWorkflowToken

Called as a Java method by ADD_WORKFLOW_TOKEN.

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.

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.

allowProblemReportAction

Allows the problem report action to execute and passes *update* as a parameter. Called as a Java method by DELETE_PROBLEMREPORT and UPDATE_PROBLEMREPORT.

appendCommonSystemInfo

Called as a Java method by GET_SYSTEM_AUDIT_INFO.

appendDatabaseAuditMessage

Called as a Java method by APPEND_DATABASE_AUDIT_INFO.

appendFileCachingMessage

Called as a Java method by APPEND_FILE_CACHING_INFO.

approveDo

Approves the content item for the workflow. Called as a Java method by WORKFLOW_APPROVE.

A.3 B

buildAllWebStringFiles

Called as a Java method by LM_BUILD_WEB_STRING_FILES.

buildExpiredContentQuery

Called as a Java method by GET_EXPIRED.

buildPreviewList

Called as a Java method by these services:

- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN

buildSearchIndex

Called as a Java method by START_SEARCH_INDEX.

buildSourceInfo

Builds the source information. Called as a Java method by PROBLEMREPORT_INFO.

A.4 C

cacheCheckin

Called as a Java method by these services:

- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN

cacheTemplates

Caches templates. Called as a Java method by these services:

- ADD_WF_TEMPLATE
- DELETE_WF_TEMPLATE
- EDIT_WF_TEMPLATE

cancelComponentInstall

Cancels the installation of a component. Called as a Java method by CANCEL_COMPONENT_INSTALL.

cancelCriteriaWorkflow

Cancels the criteria workflow and passes *WfDocuments* and *Qdocuments* as parameters. Called as a Java method by these services:

- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- DELETE_WORKFLOWCRITERIA

cancelSearchIndex

Cancels the search indexing session. Called as a Java method by CANCEL_SEARCH_INDEX.

cancelWorkflow

Cancels the workflow and passes *WfDocuments* and *Qdocuments* as parameters. Called as a Java method by DELETE_WORKFLOW and WORKFLOW_CANCEL.

canDeleteStep

Evaluates whether a workflow step can be deleted. Called as a Java method by DELETE_WORKFLOWSTEP.

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
- CHECKOUT_SUB
- UNDO_CHECKOUT
- UNDO_CHECKOUT_BY_NAME

checkCollaborationAccess

Called as a Java method by these services:

- ADD_COLLABORATION

- ADD_COLLABORATION_FORM
- DELETE_COLLABORATION
- EDIT_CLBRA_ACCESS_LIST
- EDIT_CLBRA_ACCESS_LIST_FORM
- EDIT_COLLABORATION
- EDIT_COLLABORATION_FORM
- GET_CLBRA_DOCUMENTS
- GET_CLBRA_INFO

checkConversionCache

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB.

checkCounters

Called as a Java method by INSERT_NATIVE.

checkCriteriaWorkflow

Evaluates the criteria workflow. Called as a Java method by these services:

- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB

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
- CHECKIN_NEW_SUB
- CHECKIN_SEL_FORM
- CHECKOUT_BY_NAME
- CHECKOUT_SUB
- GET_UPDATE_FORM
- UNDO_CHECKOUT
- UNDO_CHECKOUT_BY_NAME
- UPDATE_DOCINFO_SUB

checkDocState

Called as a Java method by ADD_WORKFLOWDOCUMENT_SUB.

checkForceLogin

Forces a log in. Called as a Java method by these services:

- GET_CLBRA_DOCUMENTS
- LOGIN
- PING_SERVER

- UPDATE_SUBSCRIPTION_NOTIFY
- UPDATE_SUBSCRIPTION_USED

checkForPublish
Called as a Java method by UPDATE_BYREV.

checkForRefreshingCachedResources
Called as a Java method by LOAD_RESOURCE_FILE.

checkInByID
Called as a Java method by CACHE_CHECKIN_SEL and CHECKIN_SEL_SUB.

checkInByName
Checks in the content item by name. Called as a Java method by CHECKIN_BYNAME.

checkIsLatestRev
Evaluates for the latest revision. Called as a Java method by CHECKOUT_BY_NAME and RESUBMIT_FOR_CONVERSION.

checkIsSelf
Checks whether the user is editing self. Called as a Java method by EDIT_USER_PROFILE.

checkParametersAgainstResultSet
Called as a Java method by DELETE_DOC.

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
- GET_UPDATE_PROBLEMREPORT_FORM
- PROBLEMREPORT_INFO
- RESEND_PROBLEMREPORT
- UPDATE_PROBLEMREPORT

checkRefreshUsers
Updates the user information. Called as a Java method by these services:

- ADD_USER
- CHANGE_USER_AUTH_TYPE
- DELETE_USER
- EDIT_USER
- EDIT_USER_PROFILE
- REGISTER_USER

checkRevisionProperties

Retrieves the latest revision information and passes *Qrevisions* as a parameter. Called as a Java method by these services:

- CHECKIN_SEL_FORM
- CHECKOUT_OK
- CHECKOUT_SUB
- GET_UPDATE_FORM

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
- ADD_WORKFLOW
- ADD_WORKFLOWALIASES
- ADD_WORKFLOWDOCUMENT
- ADD_WORKFLOWDOCUMENTS
- APPLET_DOCINFO
- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_BYNAME
- CHECKIN_NEW
- CHECKIN_SEL
- CONTINUE_SUBMIT_HTML_FORM
- CONTINUE_CHECKIN
- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_ENABLE
- DELETE_CHECKIN_CACHE
- DELETE_DOC
- DELETE_REV
- DELETE_REV_EX
- DELETE_WFCONTRIBUTORS
- DELETE_WORKFLOW
- DELETE_WORKFLOWCRITERIA
- DELETE_WORKFLOWDOCUMENTS
- DOC_INFO
- EDIT_WORKFLOW

- EDIT_WORKFLOWCRITERIA
- FORM_PROCESS
- GET_ARCHIVED_FILE
- GET_DOCUMENT_PROBLEMREPORTS
- GET_DYNAMIC_CONVERSION
- GET_EXTERNAL_DOC_INFO
- GET_EXTERNAL_HIGHLIGHT_INFO
- GET_EXTERNAL_XML_HIGHLIGHT_INFO
- GET_FILE
- GET_HIGHLIGHT_INFO
- GET_UPDATE_FORM
- GET_WF_COMPANION_INFO
- GET_WORKFLOW_INFO
- GET_WORKFLOW_INFO_BYNAME
- GET_WORKFLOWDOCREVISIONS
- GET_WORKFLOWDOCUMENTS
- GET_XML_HIGHLIGHT_INFO
- LOAD_RESOURCE_FILE
- RESUBMIT_FOR_CONVERSION
- REV_HISTORY
- REVIEW_WORKFLOW_DOC
- SELECTDOC
- SUBMIT_HTML_FORM
- SUBSCRIBE
- UNSUBSCRIBE
- UPDATE_DOCINFO_SUB
- VALIDATE_DOCINFO
- VIEW_DOC
- WORKFLOW_APPROVE
- WORKFLOW_CANCEL
- WORKFLOW_CHECKIN
- WORKFLOW_REJECT
- WORKFLOW_REJECT_FORM
- WORKFLOW_START

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
- ADD_WORKFLOW
- ADD_WORKFLOW_SCRIPT
- ADD_WORKFLOW_TOKEN
- ADD_WORKFLOWALIASES
- ADD_WORKFLOWDOCUMENT
- ADD_WORKFLOWDOCUMENTS
- ADD_WORKFLOWSTEP
- CHANGE_USER_AUTH_TYPE
- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_ENABLE
- DELETE_USER
- DELETE_WFCONTRIBUTORS
- DELETE_WORKFLOW
- DELETE_WORKFLOW_SCRIPT
- DELETE_WORKFLOW_TOKEN
- DELETE_WORKFLOWCRITERIA
- DELETE_WORKFLOWDOCUMENTS
- DELETE_WORKFLOWSTEP
- EDIT_USER
- EDIT_WORKFLOW
- EDIT_WORKFLOW_SCRIPT
- EDIT_WORKFLOW_TOKEN
- EDIT_WORKFLOWCRITERIA
- EDIT_WORKFLOWSTEP
- GET_ADMIN_PAGE
- GET_CRITERIA_WORKFLOWS_FOR_GROUP
- GET_USERS
- GET_WF_COMPANION_INFO
- GET_WORKFLOW_SCRIPT
- GET_WORKFLOWS_FOR_ALL
- PAGE_HANDLER
- QUERY_USER_ATTRIBUTES
- UPDATE_USEROPTION_LIST
- WORKFLOW_CANCEL
- WORKFLOW_START

checkUserAuthType

Called as a Java method by CHANGE_USER_AUTH_TYPE.

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
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_BYNAME
- CHECKIN_SEL_SUB
- CHECKOUT_SUB
- DELETE_DOC
- DELETE_REV
- DELETE_REV_EX
- WORKFLOW_CHECKIN_SUB
- WORKFLOW_START

checkWorkflowAdminAccess

Called as a Java method by GET_WORKFLOWS.

computeCompanionDirectory

Called as a Java method by ADD_WORKFLOWDOCUMENT_SUB.

computeDeleteSecurity

Called as a Java method by DELETE_REV.

computeDocID

Computes the generated content item revision ID. Called as a Java method by these services:

- GET_ARCHIVED_FILE
- GET_DYNAMIC_CONVERSION
- GET_FILE
- LOAD_RESOURCE_FILE

computeDocInfoInHtmlPage

Evaluates the content information for the HTML page. Parameters vary by service. Called as a Java method by these services:

- CHECKIN_LIST
- CONTINUE_CHECKIN
- DELETE_CHECKIN_CACHE
- DOC_SUBS_LIST
- GET_EXPIRED
- SUBSCRIPTION_LIST

- UNSUBSCRIBE_FROM_LIST
- WORK_IN_PROGRESS

computeDocSubscribers

Evaluates the content item for subscribers. Called as a Java method by GET_DOC_SUBSCRIBERS.

computeRejectTargetStep

Computes the reject target step. Called as a Java method by WORKFLOW_REJECT.

computeRemainingUsers

Called as a Java method by these services:

- GET_WORKFLOW_INFO
- GET_WORKFLOW_INFO_BYNAME
- WORKFLOW_REJECT

computeURLs

Evaluates URLS and passes *REVISIONS* as a parameter. Called as a Java method by REV_HISTORY.

computeWfDocumentsInfo

Evaluates the workflow content item information and passes *WfDocuments* as a parameter. Called as a Java method by GET_WORKFLOWDOCUMENTS.

conditionalDetermineCheckin

Prepares the form for check in. Called as a Java method by FORM_PROCESS.

continueCheckin

Called as a Java method by CONTINUE_CHECKIN.

controlIndexing

Executes the search indexing control. Called as a Java method by CONTROL_SEARCH_INDEX.

createAddEditProviderForm

Creates an Add/Edit provider form. Called as a Java method by GET_ADD_EDIT_PROVIDER_FORM.

createArchiveFileName

Creates the archive file name. Called as a Java method by GET_ARCHIVED_FILE.

createFileName

Creates a new file name. Called as a Java method by GET_FILE and LOAD_RESOURCE_FILE.

createNewRev

Called as a Java method by ADD_WORKFLOWDOCUMENT_SUB.

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
- CONTINUE_CHECKIN
- DELETE_CHECKIN_CACHE
- GET_ACTIVE_WORKFLOWS
- DOC_SUBS_LIST
- GET_CLBRA_DOCUMENTS
- GET_DATARESULTSET
- GET_EXPIRED
- GET_WORKFLOWS
- SUBSCRIBE_EX
- UNSUBSCRIBE_FROM_LIST_EX
- WORK_IN_PROGRESS

createSubscriptionType

Creates the subscription type. Called as a Java method by CREATE_SUBSCRIPTION_TYPE.

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.

createWorkflowID

Creates the workflow label. Called as a Java method by ADD_WORKFLOW.

A.5 D

Dalias

Deletes the alias. Called as an Execute Query action by DELETE_ALIAS.

DaliasUsers

Deletes the alias user. Called as an Execute Query action by DELETE_ALIAS and EDIT_ALIAS.

Dcollaboration

Called as an Execute Query action by DELETE_COLLABORATION.

DdocAccount

Deletes the content item account. Called as an Execute Query action by DELETE_DOC_ACCOUNT.

DdocFormat

Deletes the content item format. Called as an Execute Query action by DELETE_DOCFORMAT.

DdocType

Deletes the content item type. Called as an Execute Query action by DELETE_DOCTYPE.

Ddocument

Called as an Execute Query action by these services:

- REMOVE_METAFILE_SUB
- UPDATE_BYREV
- WORKFLOW_CHECKIN_SUB

decodeTopicValues

Decodes the topic values. Called as a Java method by PNE_SAVE_QUERY.

deleteAliases

Called as a Java method by DELETE_WFCONTRIBUTORS.

deleteCollaboration

Called as a Java method by DELETE_COLLABORATION.

deleteDoc

Retrieves the data assigned to the parameter *REVISIONS* and deletes the content item. Called as a Java method by DELETE_BYCLASS and DELETE_BYNAME.

deleteDocumentSubscription

Deletes the content item subscription and passes *deleteRev* as a parameter. Called as a Java method by these services:

- DELETE_BYCLASS
- DELETE_BYNAME
- DELETE_BYREV
- DELETE_DOC
- DELETE_REV_EX

deleteProblemReport

Deletes the problem report information. Called as a Java method by DELETE_PROBLEMREPORT.

deleteProvider

Deletes the provider. Called as a Java method by DELETE_PROVIDER.

deleteResultTemplate

Deletes the result template. Called as a Java method by DELETE_RESULT_TEMPLATE.

deleteRev

Deletes a previous revision of a content item and passes *DOC_INFO* as a parameter. Called as a Java method by DELETE_REV and DELETE_REV_EX.

deleteRevFiles

Called as a Java method by these services:

- REMOVE_METAFILE_SUB
- UPDATE_BYREV
- WORKFLOW_CHECKIN_SUB

deleteSchemaRelation

Called as a Java method by DELETE_SCHEMA_RELATION.

deleteSchemaTable

Called as a Java method by DELETE_SCHEMA_TABLE.

deleteSchemaView

Called as a Java method by DELETE_SCHEMA_VIEW.

deleteSubscriptionType

Action that deletes the subscription type. Called as a Java method by DELETE_SUBSCRIPTION_TYPE.

deleteTemplate

Deletes the workflow template. Called as a Java method by ADD_WF_TEMPLATE.

deleteWfDesign

Called as a Java method by DELETE_WORKFLOW and DELETE_WORKFLOWCRITERIA.

deleteWfDocuments

Deletes workflow content items. Called as a Java method by DELETE_WORKFLOWDOCUMENTS.

deleteWorkflowScript

Deletes a workflow script. Called as a Java method by DELETE_WORKFLOW_SCRIPT.

deleteWorkflowStepScript

Deletes the script associated with the workflow step. Called as a Java method by DELETE_WORKFLOWSTEP.

deleteWorkflowToken

Deletes the workflow token. Called as a Java method by DELETE_WORKFLOW_TOKEN.

determineCheckin

Prepares the form for check in. Called as a Java method by these services:

- CHECKIN_UNIVERSAL

- CONTINUE_SUBMIT_HTML_FORM
- SUBMIT_HTML_FORM

DextensionMap

Deletes the extension map. Called as an Execute Query action by DELETE_DOCEXTENSION.

DgroupRole

Deletes the role. Called as an Execute Query action by DELETE_GROUP.

disableSendFile

Called as a Java method by GET_DYNAMIC_CONVERSION.

Dmetadef

Deletes the meta definition information. Called as an Execute Query action by DEL_METADEF.

doCachedCheckinCleanup

Called as a Java method by CONTINUE_CHECKIN and DELETE_CHECKIN_CACHE.

doCachedCleanup

Called as a Java method by CONTINUE_SUBMIT_HTML_FORM.

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.

docHistoryInfo

Evaluates content item history information and passes *Checkout* and *IdocHistory* as parameters. Called as a Java method by these services:

- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB
- CHECKOUT_SUB
- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- CRITERIAWORKFLOW_ENABLE
- DELETE_BYCLASS
- DELETE_BYNAME
- DELETE_BYREV
- DELETE_DOC
- DELETE_REV
- DELETE_REV_EX

- INSERT_NATIVE
- INSERT_NEW
- UNDO_CHECKOUT
- UNDO_CHECKOUT_BY_NAME
- UPDATE_BYREV
- UPDATE_DOCINFO_BYREV
- UPDATE_DOCINFO_SUB
- WORKFLOW_APPROVE
- WORKFLOW_CANCEL
- WORKFLOW_CHECKIN_SUB
- WORKFLOW_REJECT
- WORKFLOW_START

docRefinery

Initiates the refinery check-in process. Called as a Java method by these services:

- CHECKIN_BYNAME
- CHECKIN_NEW
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB
- INSERT_NATIVE
- INSERT_NEW
- REPLACE_METAFILE_SUB
- UPDATE_BYREV
- UPDATE_DOCINFO_SUB
- WORKFLOW_CHECKIN_SUB

doSubService

Executes the specified SubService. Called as a Java method by these services:

- ADD_WORKFLOWDOCUMENT
- CHECKIN_NEW
- CHECKIN_SEL
- CHECKOUT
- CHECKOUT_BY_NAME
- GET_DOC_PAGE
- GET_DYNAMIC_CONVERSION
- GET_SECURE_PAGE
- GET_SYSTEM_AUDIT_INFO
- LM_LOAD_LAYOUTS
- UPDATE_DOCINFO

- UPDATE_DOCINFO_BYFORM
- WORKFLOW_CHECKIN

doSubserviceIfMetafile
Called as a Java method byUPDATE_DOCINFO_SUB.

doUpload
Called as a Java method by CHUNKED_UPLOAD.

downloadComponent
Called as a Java method by DOWNLOAD_COMPONENT.

downloadItems
Called as a Java method by DOWNLOAD_LISTBOX_ITEMS.

doWorkflowAction
Performs the workflow action. Called as a Java method by these services:

- DELETE_BYCLASS
- DELETE_BYNAME
- DELETE_BYREV
- DELETE_DOC
- DELETE_REV
- DELETE_REV_EX

DproblemReport
Deletes the problem report information. Called as an Execute Query action by DELETE_PROBLEMREPORT.

Drole
Deletes the role. Called as an Execute Query action by DELETE_ROLE.

DsecurityGroup
Deletes the security group. Called as an Execute Query action by DELETE_GROUP.

DsubscriptionType
Deletes the subscription type. Called as an Execute Query action by DELETE_SUBSCRIPTION_TYPE.

Duser
Called as an Execute Query action by DELETE_USER.

DuserSecurityAttributes
Deletes the user security attributes. Called as an Execute Query action by these services:

- ADD_USER
- DELETE_ROLE
- EDIT_USER

DuserSubscription

Called as an Execute Query action by DELETE_USER.

Dworkflow

Deletes the workflow. Called as an Execute Query method by DELETE_WORKFLOW and DELETE_WORKFLOWCRITERIA.

DworkflowAliasAll

Deletes the workflow aliases. Called as an Execute Query action by DELETE_WORKFLOW and DELETE_WORKFLOWCRITERIA.

DworkflowCriteria

Deletes the workflow criteria. Called as an Execute Query action by DELETE_WORKFLOWCRITERIA.

DworkflowDocState

Deletes the workflow content item state. Called as an Execute Query action by WORKFLOW_REJECT.

DworkflowDocumentAll

Deletes the workflow content items. Called as an Execute Query by these services:

- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- DELETE_WORKFLOW
- DELETE_WORKFLOWCRITERIA

DworkflowStateAll

Deletes the workflow states. Called as an Execute Query action by these services:

- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- DELETE_WORKFLOW
- DELETE_WORKFLOWCRITERIA
- WORKFLOW_CANCEL

DworkflowStep

Deletes the workflow steps. Called as an Execute Query action by DELETE_WORKFLOWSTEP.

DworkflowStepAliases

Deletes the workflow step aliases. Called as an Execute Query action by DELETE_WORKFLOWSTEP.

DworkflowStepsAll

Deletes the workflow steps. Called as an Execute Query action by DELETE_WORKFLOW and DELETE_WORKFLOWCRITERIA.

DworkflowUserAttributeAll

Deletes the workflow user attributes. Called as an Execute Query action by these services:

- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- DELETE_WORKFLOWCRITERIA

A.6 E

editCollaboration

Called as a Java method by EDIT_CLBRA_ACCESS_LIST and EDIT_COLLABORATION.

editCriteria

Prepares the criteria for edit. Called as a Java method by EDIT_WORKFLOWCRITERIA.

editRole

Edits role information. Called as a Java method by EDIT_ROLE.

editSchemaRelation

Called as a Java method by EDIT_SCHEMA_RELATION.

editSchemaView

Called as a Java method by EDIT_SCHEMA_VIEW.

editSchemaViewValues

Called as a Java method by EDIT_SCHEMA_VIEW_VALUES.

editTemplate

Edits the workflow template. Called as a Java method by EDIT_WF_TEMPLATE.

editWorkflowScript

Called as a Java method by EDIT_WORKFLOW_SCRIPT.

editWorkflowStep

Edits the workflow step and passes *IworkflowStep* as a parameter. Called as a Java method by EDIT_WORKFLOWSTEP.

editWorkflowStepScript

Edits the script associated with the workflow step. Called as a Java method by EDIT_WORKFLOWSTEP.

editWorkflowToken

Called as a Java method by EDIT_WORKFLOW_TOKEN.

enableDisableProvider

Changes the provider enable state. Called as a Java method by ENABLE_DISABLE_PROVIDER.

executeArchiveMethod

Performs a specified action on an archive collection: add, delete, and so on. Called as a Java method by these services:

- ADD_ARCHIVE
- ADD_COLLECTION
- ADD_PROXIEDCOLLECTION
- CANCEL_ARCHIVE
- COPY_ARCHIVE
- DELETE_ARCHIVE
- DELETE_BATCH_FILE
- DELETE_BATCH_FILE_DOCUMENTS
- EDIT_ARCHIVE
- EDIT_ARCHIVEDATA
- EDIT_EXPORTERS
- EDIT_TRANSFEROPTIONS
- EXPORT_ARCHIVE
- GET_ARCHIVES
- GET_BATCH_SCHEMA
- GET_BATCH_VALUES
- GET_BATCHFILES
- GET_BATCH_FILE_DOCUMENTS
- GET_REPLICATION_DATA
- GET_TARGET_INFO
- GET_TARGET_TRANSFER_STATUS
- GET_TRANSFER_SOURCE_INFO
- IMPORT_ARCHIVE
- IMPORT_DOCUMENT
- INSERT_NATIVE
- REMOVE_COLLECTION
- REMOVE_EXPORTER
- REMOVE_IMPORTER
- REMOVE_PROXIEDTRANSFER
- REMOVE_PROXIEDTRANSFER
- REQUEST_TRANSFER
- TRANSFER_ARCHIVE
- UPDATE_TARGET_TOTALS
- UPDATE_TRANSFER_STATUS
- UPLOAD_ARCHIVE_TRANSFER

executeCommands

Called as a Java method by EXECUTE_BATCH.

executeManifest

Executes the manifest. Called as a Java method by UPLOAD_NEW_COMPONENT.

executePageService

Executes the page service. Called as a Java method by PAGE_HANDLER and SAVE_GLOBALINCLUDES.

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.

A.7 F

filterUpdateData

Filters the update information. Called as a Java method by EDIT_USER_PROFILE.

A.8 G

getCollaborationData

Called as a Java method by these services:

- EDIT_CLBRA_ACCESS_LIST_FORM
- EDIT_COLLABORATION_FORM
- GET_CLBRA_INFO

getCollaborationDocs

Called as a Java method by GET_CLBRA_DOCUMENTS.

getCollaborations

Called as a Java method by GET_COLLABORATION_LIST.

getCompanionHistory

Called as a Java method by GET_WORKFLOW_INFO and GET_WORKFLOW_INFO_BYNAME.

getComponentConfig

Called as a Java method by GET_COMPONENT_CONFIG.

getDefaultDocFormats

Sets the default content item format. Called as a Java method by DOC_FORMATS_WIZARD.

getDocFormats

Retrieves the file formats for the content item. Called as a Java method by these services:

- DOC_INFO
- REVIEW_WORKFLOW_DOC
- WORKFLOW_REJECT_FORM

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
- REVIEW_WORKFLOW_DOC
- SUBSCRIBE_DOC_USER
- SUBSCRIBE_FORM
- UNSUBSCRIBE_FORM

getDocumentList

Retrieves the content item list. Called as a Java method by DOC_SUBS_LIST.

getDynamicPage

Retrieves the dynamic page. Called as a Java method by GET_DYNAMIC_PAGE.

getExternalDocInfo

Retrieves the external content item information. Called as a Java method by GET_EXTERNAL_DOC_INFO.

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
- GET_EXTERNAL_HIGHLIGHT_INFO
- GET_EXTERNAL_XML_HIGHLIGHT_INFO
- VIEW_DOC

getFilesInAppDir

Retrieves the files listed in the application directory. Called as a Java method by GET_FILELIST.

getHighlightInfo

Presents the content item highlight information. Called as a Java method by these services:

- GET_EXTERNAL_HIGHLIGHT_INFO
- GET_EXTERNAL_XML_HIGHLIGHT_INFO
- GET_HIGHLIGHT_INFO
- GET_XML_HIGHLIGHT_INFO

getLatestID

Passes *QlatestID* as a parameter. Called as a Java method by DELETE_REV.

getLatestIDRevInfo

Retrieves the latest revision information and passes *Qrevisions* as a parameter. Called as a Java method by these services:

- CHECKIN_SEL_FORM
- CHECKOUT
- CHECKOUT_BY_NAME
- UNDO_CHECKOUT
- UNDO_CHECKOUT_BY_NAME

getOptionList

Retrieves the latest revision information and passes *Qrevisions* as a parameter. Called as a Java method by GET_OPTION_LIST and UPDATE_OPTION_LIST.

getOutgoingProviders

Called as a Java method by GET_PROXIEDSERVERS.

getProxiedArchiveCollections

Called as a Java method by GET_PROXIED_ARCHIVECOLLECTIONS.

getSchemaRelations

Called as a Java method by GET_SCHEMA_RELATIONS.

getSchemaTableInfo

Called as a Java method by GET_SCHEMA_TABLE_INFO and GET_SCHEMA_VIEW_INFO.

getSchemaTables

Called as a Java method by GET_SCHEMA_TABLES.

getSchemaViewInfo

Called as a Java method by GET_SCHEMA_VIEW_INFO.

getSchemaViews

Called as a Java method by GET_SCHEMA_VIEWS.

getSearchResults

Retrieves the search results. Called as a Java method by GET_SEARCH_RESULTS and PNE_GET_SEARCH_RESULTS.

getSubscriptionList

Retrieves the subscription list and passes *QdocNameSubscription QnotDocNameSubscriptions* as parameters. Called as a Java method by SUBSCRIPTION_LIST and UNSUBSCRIBE_FROM_LIST.

getTable

Called as a Java method by GET_TABLE.

getTemplate

Retrieves template data. Called as a Java method by GET_WF_TEMPLATE.

getTemplateConversions

Called as a Java method by GET_TEMPLATE_CONVERSIONS.

getTemplates

Retrieves the workflow template data. Called as a Java method by GET_WF_TEMPLATES.

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 and REVIEW_WORKFLOW_DOC.

getUserAttributes

Retrieves user attributes. Called as a Java method by QUERY_USER_ATTRIBUTES.

getUserCollaborationList

Called as a Java method by GET_USER_CLBRA_LIST.

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
- DOC_INFO_LATESTRELEASE
- DOC_INFO_SIMPLE
- DOC_INFO_SIMPLE_BYREV
- GET_WORKFLOW_INFO
- GET_WORKFLOW_INFO_BYNAME
- PROBLEMREPORT_INFO
- REVIEW_WORKFLOW_DOC

getUserProfile

Retrieves user profile information. Called as a Java method by GET_USER_INFO.

getUsers

Retrieves user list. Called as a Java method by GET_USERS.

getUserUnique

Retrieves the unique user name. Called as a Java method by REGISTER_USER.

getViewEditInfo

Called as a Java method by GET_SCHEMA_VIEW_EDIT_INFO.

getViewValues

Called as a Java method by GET_SCHEMA_VIEW_VALUES.

getWebAppStatus

Called as a Java method by GET_WEB_APP_STATUS.

getWfDocuments

Retrieves workflow content items and passes *WfDocuments* as a parameter. Called as a Java method by these services:

- DELETE_WORKFLOW
- DELETE_WORKFLOWCRITERIA
- WORKFLOW_CANCEL
- WORKFLOW_START

getWorkflowDesignInfo

Called as a Java method by GET_WORKFLOW.

getWorkflowDocumentInfo

Called as a Java method by GET_WORKFLOW.

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 and REVIEW_WORKFLOW_DOC.

getWorkflowScript

Called as a Java method by GET_WORKFLOW_SCRIPT.

getWorkflowStepAliasesInfo

Retrieves the workflow step alias information. Called as a Java method by GET_WORKFLOW.

A.9 H

hidePassword

Hides the password and passes *USER_INFO* as a parameter. Called as a Java method by GET_USER_INFO.

A.10 I

Ialias

Inserts the alias user information. Called as an Execute Query action by ADD_ALIAS.

Icollaboration

Called as an Execute Query action by ADD_COLLABORATION.

IdocAccount

Inserts a new account. Called as an Execute Query action by ADD_DOC_ACCOUNT.

IdocFormat

Inserts the content item format. Called as an Execute Query action by ADD_DOCFORMAT.

IdocType

Inserts the content item type. Called as an Execute Query action by ADD_DOCTYPE.

IextensionMap

Inserts the extension map. Called as an Execute Query action by ADD_DOCEXTENSION.

Imeta

Inserts the meta data information. Called as an Execute Query action by these services:

- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB
- INSERT_NATIVE
- INSERT_NEW

Imetadef

Inserts the meta data definition information. Called as an Execute Query action by ADD_METADEF.

insertAliasUsers

Adds the alias information to the database. Called as a Java method by ADD_ALIAS and EDIT_ALIAS.

insertGroupRow

Called as a Java method by ADD_GROUP.

IproblemReport

Updates the problem report. Called as an Execute Query action by ADD_PROBLEMREPORT.

Irevision

Updates the revision ID. Called as an Execute Query action by these services:

- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB
- INSERT_NATIVE
- INSERT_NEW

Irole

Inserts the role. Called as an Execute Query by ADD_ROLE.

isDocCheckedOut

Called as a Java method by these services:

- CACHE_WORKFLOW_CHECKIN
- WORKFLOW_APPROVE
- WORKFLOW_CHECKIN_SUB

lsecurityGroup

Inserts the security group. Called as an Execute Query action by ADD_GROUP.

lsubscription

Inserts the subscription. Called as an Execute Query by SUBSCRIBE_EX.

lworkflow

Inserts the workflow information in the database. Called as an Execute Query action by ADD_WORKFLOW.

lworkflowDocument

Called as an Execute Query action by ADD_WORKFLOWDOCUMENT_SUB.

lworkflowState

Provides an internal status table that stores information about content items in active workflows. Called as an Execute Query action by WORKFLOW_APPROVE.

A.11 L

loadActiveComponentData

Loads configuration information and passes *ACTIVE_COMPONENTS* as a parameter. Called as a Java method by CONFIG_INFO.

loadAndValidateValues

Evaluates the specified values. Called as a Java method by these services:

- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- CRITERIAWORKFLOW_ENABLE
- WORKFLOW_CANCEL
- WORKFLOW_START

loadCollaborationList

Called as a Java method by GET_ACTIVE_WORKFLOWS.

loadComponentInstallInfo

Called as a Java method by GET_COMPONENT_INSTALL_FORM.

loadComponentInstallSettings

Called as a Java method by GET_COMPONENT_INSTALL_SETTINGS'.

loadConfigurationInfo

Loads the content item specific configuration information. Called as a Java method by GET_DOC_CONFIG_INFO.

loadDefaultInfo

Loads the default configuration information. Called as a Java method by these services:

- ASSIGN_DOCINFO_FORM
- CHECKIN_NEW_FORM
- CHECKIN_SEL_FORM
- CHECKIN_SIMILAR_FORM
- GET_DOC_CONFIG_INFO
- GET_PORTAL_PAGE
- GET_UPDATE_FORM
- LOAD_DOC_ENVIRONMENT

loadDocConfig

Loads the content item configuration information. Called as a Java method by DOC_FORMATS_WIZARD and EDIT_DOC_FORMATS.

loadDocDefaults

Called as a Java method by these services:

- ASSIGN_DOCINFO_FORM
- CHECKIN_NEW_FORM
- CHECKIN_SIMILAR_FORM

loadFilterConfig

Called as a Java method by these services:

- ASSIGN_DOCINFO_FORM
- CHECKIN_NEW_FORM
- CHECKIN_SIMILAR_FORM

loadGlobalIncludes

Loads the global includes. Called as a Java method by LOAD_GLOBALINCLUDES.

loadMetaDefaults

Loads the default configuration information. Called as a Java method these services:

- ASSIGN_DOCINFO_FORM
- CHECKIN_NEW_FORM
- CHECKIN_SIMILAR_FORM
- GET_UPDATE_FORM

loadMetaOptionsLists

Loads the meta data options list. Called as a Java method by these services:

- ASSIGN_DOCINFO_FORM
- CHECKIN_NEW_FORM
- CHECKIN_SEL_FORM
- CHECKIN_SIMILAR_FORM
- GET_DOC_CONFIG_INFO
- GET_PORTAL_PAGE
- GET_UPDATE_FORM
- LOAD_DOC_ENVIRONMENT

loadPRStateLists

Loads the problem report state lists. Called as a Java method by GET_PROBLEMREPORTS_SEARCH_FORM and GET_UPDATE_PROBLEMREPORT_FORM.

loadRegisteredProjects

Retrieves problem report search information. Called as a Java method by GET_PROBLEMREPORTS_SEARCH_FORM.

loadServerOutput

Loads the server output. Called as a Java method by CLEAR_SERVER_OUTPUT and GET_SERVER_OUTPUT.

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
- DELETE_SUBSCRIPTION_TYPE
- GET_ARCHIVECOLLECTIONS
- GET_DOC_CONFIG_INFO
- GET_PACKAGE_ENVIRONMENT_PAGE
- GET_PORTAL_PAGE
- GET_RESULT_OPTIONS
- GET_SYSTEM_AUDIT_INFO
- GET_USER_INFO
- LOAD_DOC_ENVIRONMENT
- UPDATE_SUBSCRIPTION_TYPE

loadTopic

Called as a Java method by these services:

- LOAD_PNE_PORTAL
- LOAD_USER_TOPIC
- LOAD_WORKFLOW_QUEUE

loadTraceFlags

Called as a Java method by GET_SYSTEM_AUDIT_INFO.

loadUserAndCheckEditAllowed

Loads the user information and evaluates the allowed edit privilege for the user. Called as a Java method by these services:

- ADD_USER
- CHANGE_USER_AUTH_TYPE
- DELETE_USER
- EDIT_USER
- QUERY_USER_ATTRIBUTES

loadUserMetaData

Loads the user meta data information. Called as a Java method by these services:

- GET_FILTER_ADMIN_PAGE
- GET_SELF_REGISTER_PAGE
- GET_USER_INFO

loadWfCompanionInfo

Retrieves information for companion workflows. Called as a Java method by GET_WF_COMPANION_INFO.

A.12 M

makeNewRevClass

Called as a Java method by CACHE_CHECKIN_NEW and CHECKIN_NEW_SUB.

mapDocNamedResultSetValuesCheckMetaChange

Called as a Java method by CACHE_WORKFLOW_CHECKIN.

mapDocResultSetCheckMetaChange

Called as a Java method by CACHE_CHECKIN_SEL and CHECKIN_SEL_SUB.

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
- DOC_INFO
- RESUBMIT_FOR_CONVERSION
- REVIEW_WORKFLOW_DOC

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
- APPLET_DOCINFO
- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_FORM
- CHECKIN_SEL_SUB
- DOC_INFO_LATESTRELEASE
- DOC_INFO_SIMPLE_BYREV
- INSERT_NEW
- RESUBMIT_FOR_CONVERSION
- UNDO_CHECKOUT
- UNDO_CHECKOUT_BY_NAME
- SUBSCRIBE
- WORKFLOW_APPROVE
- WORKFLOW_CHECKIN_SUB
- WORKFLOW_REJECT

markDocDeleted

Sets the status message and passes *delete_doc* as a parameter. Called as a Java method by these services:

- DELETE_BYCLASS
- DELETE_BYNAME
- DELETE_DOC

markRevDeleted

Updates the revision status as deleted. Called as a Java method by these services:

- DELETE_BYREV
- DELETE_REV
- DELETE_REV_EX

mergeTable

Merges the named table. Called as a Java method by MERGE_TABLE.

A.13 N

notifyChange

Notifies the provider of changes to the content item. Called as a Java method by NOTIFY_CHANGE.

notifyCollaborationUsers

Called as a Java method by ADD_COLLABORATION and EDIT_CLBRA_ACCESS_LIST.

notifyContributor

Called as a Java method by ADD_PROBLEMREPORT and RESEND_PROBLEMREPORT.

A.14 P

packageEnvironment

Called as a Java method by PACKAGE_ENVIRONMENT.

packageLocalization

Called as a Java method by LOAD_USER_LOCALIZATION.

pageRequest

Executes an HTML page request. Called as a Java method by these services:

- GET_ADMIN_PAGE
- GET_DOC_PAGE
- GET_PORTAL_PAGE
- GET_SECURE_PAGE
- LOGIN
- PNE_SAVE_QUERY
- PNE_UPDATE_PERSONAL_URLS
- PNE_UPDATE_PORTAL_INFO

postCheckinFilter

Processes the post checkin filter. Called as a Java method by FORM_PROCESS.

postHtmlFormCheckin

Submits the HTML form. Called as a Java method by CONTINUE_SUBMIT_HTML_FORM and SUBMIT_HTML_FORM.

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
- CHECKIN_SEL

- UPDATE_DOCINFO_SUB

prepareDocInfoValidate
Called as a Java method by VALIDATE_DOCINFO.

prepareFormContinue
Called as a Java method by CONTINUE_SUBMIT_HTML_FORM.

prepareForPreview
Called as a Java method by CACHE_SUBMIT_HTML_FORM.

prepareInsertNew
Called as a Java method by INSERT_NEW.

prepareMailForStepUsers
Sends email to the workflow step users. Called as a Java method by WORKFLOW_REJECT and WORKFLOW_START.

prepareRedirect
Prepares the redirect template for the specified service. Called as a Java method by these services:

- ADD_COLLABORATION
- ADD_EDIT_PROVIDER
- ADD_PROBLEMREPORT
- CHECKIN_NEW
- CHECKIN_SEL
- CHECKOUT
- CLEAR_SERVER_OUTPUT
- CONTINUE_SUBMIT_HTML_FORM
- DELETE_COLLABORATION
- DELETE_PROBLEMREPORT
- DELETE_PROVIDER
- DELETE_REV
- EDIT_CLBRA_ACCESS_LIST
- EDIT_COLLABORATION
- EDIT_DOC_FORMATS
- EDIT_TRACE_OPTIONS
- EDIT_USER_PROFILE
- ENABLE_DISABLE_PROVIDER
- RESUBMIT_FOR_CONVERSION
- SAVE_TEMPLATE_CONVERSIONS
- SAVE_USER_TOPICS

- SUBMIT_HTML_FORM
- SUBSCRIBE
- TEST_PROVIDER
- UNDO_CHECKOUT
- UNSUBSCRIBE
- UPDATE_DOCINFO_BYFORM
- UPDATE_FILTER_INFO
- UPDATE_PROBLEMREPORT
- UPLOAD_NEW_COMPONENT
- WORKFLOW_APPROVE
- WORKFLOW_CHECKIN
- WORKFLOW_REJECT

prepareTopicEdits

Prepares the topic edits. Called as a Java method by these services:

- EDIT_USER_PROFILE
- GET_CLBRA_DOCUMENTS
- PNE_GET_SEARCH_RESULTS
- PNE_SAVE_QUERY
- PNE_UPDATE_PERSONAL_URLS
- PNE_UPDATE_PORTAL_INFO
- SAVE_USER_TOPICS

prepareWebViewableDelivery

Called as a Java method by GET_DYNAMIC_CONVERSION.

prepSubscription

Prepares the subscription. Called as a Java method by DOC_SUBS_LIST.

prepSubscriptionDateUpdate

Prepares the subscription date for update and passes the specified parameter. Called as a Java method by UPDATE_SUBSCRIPTION_NOTIFY and UPDATE_SUBSCRIPTION_USED.

processCheckinArchive

Checks the content item into the archive. Called as a Java method by these services:

- CHECKIN_ARCHIVE
- CHECKIN_ARCHIVE_NO_NOTIFY

processForm

Processes the form submission and passes *checkSecurity* as a parameter. Called as a Java method by FORM_PROCESS.

A.15 Q

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 and EDIT_ALIAS.

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.

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.

Qcache

Called as a Select Cache Query action by CONTINUE_SUBMIT_HTML_FORM.

QcheckinCache

Called as a Select Cache Query by these services:

- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN
- CONTINUE_CHECKIN
- DELETE_CHECKIN_CACHE
- GET_CACHED_CHECKIN_INFO

QcheckinCachesForUser

Called as a Select Cache Query action by these services:

- CHECKIN_LIST
- CONTINUE_CHECKIN
- "DELETE_BYREV" on page 43

Qcollaboration

Called as a Select Query action by ADD_COLLABORATION.

Called as a Select Cache Query action by these services:

- DELETE_COLLABORATION
- EDIT_CLBRA_ACCESS_LIST
- EDIT_CLBRA_ACCESS_LIST_FORM
- EDIT_COLLABORATION
- EDIT_COLLABORATION_FORM

- GET_CLBRA_DOCUMENTS
- GET_CLBRA_INFO

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.

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.

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
- APPLET_DOCINFO
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_SEL_FORM
- CHECKOUT
- CHECKOUT_BY_NAME
- CHECKOUT_OK
- DELETE_DOC
- DELETE_REV
- DELETE_REV_EX
- DOC_INFO
- DOC_INFO_LATESTRELEASE
- DOC_INFO_SIMPLE
- DOC_INFO_SIMPLE_BYREV
- GET_DOC_CONFIG_INFO
- GET_DYNAMIC_CONVERSION
- GET_FILE
- GET_UPDATE_FORM
- GET_WORKFLOW_INFO
- NOTIFY_CONTRIBUTOR
- RESUBMIT_FOR_CONVERSION
- REV_HISTORY
- REVIEW_WORKFLOW_DOC
- SELECTDOC
- SUBSCRIBE

- SUBSCRIBE_DOC_USER
- SUBSCRIBE_FORM
- UNSUBSCRIBE_FORM
- UPDATE_DOCINFO_BYREV
- UPDATE_DOCINFO_SUB
- WORKFLOW_CHECKIN

QdocInfoSimilarCheckin

Called as a Select Cache Query by CHECKIN_SEL_SUB.

QdocName

Called as a Select Cache Query by these services:

- DELETE_PROBLEMREPORT
- DOC_INFO_SIMPLE
- GET_UPDATE_PROBLEMREPORT_FORM
- GET_WF_COMPANION_INFO
- PROBLEMREPORT_INFO
- RESEND_PROBLEMREPORT
- UPDATE_PROBLEMREPORT

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
- GET_DOCUMENT_PROBLEMREPORTS
- GET_WORKFLOW_INFO_BYNAME

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.

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.

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.

QdocumentProblemReports

Called as a Select Cache Query action by GET_DOCUMENT_PROBLEMREPORTS.

Qdocuments

Called as a Select Cache Query action by UPDATE_DOCINFO_BYREV and UPDATE_DOCINFO_SUB.

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.

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.

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.

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.

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.

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.

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.

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.

QisDocWFLocked

Called as a Select Query action by ADD_WORKFLOWDOCUMENT_SUB.

QisWfTokenUsed

Called as a Select Query action by DELETE_WORKFLOW_TOKEN.

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 and UNDO_CHECKOUT_BY_NAME.

Qmetadef

Called as a Select Query action by these services:

- ADD_METADEF
- EDIT_METADEF
- DEL_METADEF

QmetaFieldInfo

Retrieves meta data field information and passes *MetaFieldInfo* as a parameter. Called as a Select Cache Query action by GET_METADEFS and UPDATE_META_TABLE.

QODMAdocInfo

Called as a Select Cache Query action by ODMA_DOC_INFO_SIMPLE.

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
- DELETE_PROBLEMREPORT
- GET_UPDATE_PROBLEMREPORT_FORM
- PROBLEMREPORT_INFO
- RESEND_PROBLEMREPORT
- UPDATE_PROBLEMREPORT

QprojectDocument

Called as a Select Cache Query by these services:

- ADD_PROBLEMREPORT
- GET_DOCUMENT_PROBLEMREPORTS
- PROBLEMREPORT_INFO
- RESEND_PROBLEMREPORT

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
- GET_DOCUMENT_PROBLEMREPORTS
- PROBLEMREPORT_INFO
- RESEND_PROBLEMREPORT

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 and GET_XML_HIGHLIGHT_INFO.

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 and REVIEW_WORKFLOW_DOC.

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.

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 and DELETE_DOC.

QrevisionsByName

Called as a Select Cache Query action by DELETE_BYNAME.

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.

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.

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.

Called as a Select Cache Query action by QUERY_GROUP.

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.

QuserMetaFieldInfo

Called as a Select Cache Query action by GET_USER_METADEFS and UPDATE_USER_META_TABLE.

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.

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.

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 and CRITERIAWORKFLOW_DISABLE_SUB.

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.

QwfDocName

Called as a Select Cache Query action by ADD_WORKFLOWDOCUMENT_SUB.

QwfDocState

Called as a Select Cache Query action by GET_WORKFLOW_INFO and GET_WORKFLOW_INFO_BYNAME.

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.

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.

Called as a Select Cache Query action by these services:

- ADD_WORKFLOWALIASES
- ADD_WORKFLOWDOCUMENT
- ADD_WORKFLOWDOCUMENTS
- ADD_WORKFLOWSTEP
- CRITERIAWORKFLOW_ENABLE
- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- DELETE_WFCONTRIBUTORS
- DELETE_WORKFLOW
- DELETE_WORKFLOWCRITERIA
- DELETE_WORKFLOWSTEP
- DELETE_WORKFLOWDOCUMENTS

- DELETE_WORKFLOWSTEP
- EDIT_WORKFLOW
- EDIT_WORKFLOWCRITERIA
- EDIT_WORKFLOWSTEP
- GET_WORKFLOW
- GET_WORKFLOWDOCUMENTS
- GET_WORKFLOWDOCREVISIONS
- WORKFLOW_START
- WORKFLOW_CANCEL

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.

QworkflowAliases

Retrieves workflow alias information. Called as a Select Cache Query action by DELETE_WFCONTRIBUTORS

QworkflowCriteriaForGroup

Called as a Select Cache Query action by GET_CRITERIA_WORKFLOWS_FOR_GROUP.

QworkflowCriteriaStepsForGroup

Called as a Select Cache Query action by GET_CRITERIA_WORKFLOWS_FOR_GROUP.

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
- GET_WORKFLOW_INFO
- GET_WORKFLOW_INFO_BYNAME

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 and GET_WORKFLOWDOCUMENTS.

QworkflowForID

Called as a Select Cache Query action by GET_WORKFLOW_INFO and GET_WORKFLOW_INFO_BYNAME.

Qworkflows

Called as a Select Cache Query by GET_WORKFLOWS_FOR_ALL.

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. Called as a Select Cache Query action by EDIT_WORKFLOWSTEP.

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
- GET_WORKFLOW_INFO
- GET_WORKFLOW_INFO_BYNAME
- WORKFLOW_START

QworkflowStepsAll

Called as a Select Cache Query action by GET_WORKFLOWS_FOR_ALL.

A.16 R

refreshCache

Refreshes the cached specified information. The specified information varies by service. Called as a Java method by these services:

- ADD_ALIAS
- ADD_DOC_ACCOUNT
- ADD_DOCEXTENSION
- ADD_DOCFORMAT
- ADD_DOCTYPE
- ADD_METADEF
- ADD_WORKFLOW
- ADD_WORKFLOWALIASES
- ADD_WORKFLOWDOCUMENT
- ADD_WORKFLOWDOCUMENTS
- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_ENABLE
- DEL_METADEF
- DELETE_ALIAS
- DELETE_DOC_ACCOUNT
- DELETE_DOCEXTENSION
- DELETE_DOCFORMAT
- DELETE_DOCTYPE
- DELETE_WFCONTRIBUTORS

- DELETE_WORKFLOW
- DELETE_WORKFLOWCRITERIA
- DELETE_WORKFLOWDOCUMENTS
- EDIT_ALIAS
- EDIT_DOCEXTENSION
- EDIT_DOCFORMAT
- EDIT_DOC_FORMATS
- EDIT_DOCTYPE
- EDIT_METADEF
- EDIT_WORKFLOW
- EDIT_WORKFLOWCRITERIA
- UPDATE_META_TABLE
- UPDATE_USER_META
- UPDATE_USER_META_TABLE
- WORKFLOW_CANCEL
- WORKFLOW_START

refreshLayoutLists
Called as a Java method by LM_LOAD_LAYOUTS_SUB.

refreshRoles
Refreshes the user roles. Called as a Java method by these services:

- ADD_GROUP
- ADD_ROLE
- DELETE_GROUP
- DELETE_ROLE
- EDIT_ROLE

rejectDoc
Rejects the content item and passes *UrevisionStatus* as a parameter. Called as a Java method by WORKFLOW_REJECT.

remoteCredentialsCheck
Called as a Java method by CHECK_USER_CREDENTIALS.

removeCachedUser
Called as a Java method by these services:

- ADD_USER
- DELETE_USER
- EDIT_USER
- REGISTER_USER

removeContext

Called as a Java method by REMOVE_WEB_APP.

removeSubscription

Deletes the subscription and passes *Dsubscription* as a parameter. Called as a Java method by these services:

- UNSUBSCRIBE
- UNSUBSCRIBE_FROM_LIST
- UNSUBSCRIBE_FROM_LIST_EX

renameValues

Renames the associated values and passes specified parameters. Called as a Java method by these services:

- ADD_PROBLEMREPORT
- CACHE_CHECKIN_SEL
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_SEL_SUB
- CHECKIN_SIMILAR_FORM
- DELETE_PROBLEMREPORT
- DELETE_REV
- DELETE_ROLE
- DELETE_USER
- GET_UPDATE_FORM
- GET_WORKFLOW_INFO
- GET_WORKFLOW_INFO_BYNAME
- NOTIFY_CONTRIBUTOR
- SUBSCRIBE
- WORKFLOW_APPROVE
- WORKFLOW_CHECKIN_SUB
- WORKFLOW_REJECT
- WORKFLOW_START

requestSecurityInfo

Requests the security information. Called as a Java method by REQUEST_SECURITYINFO.

resubmitDocToConversion

Resubmits the content item for conversion. Called as a Java method by RESUBMIT_FOR_CONVERSION.

retrieveAllProviderInfo

Retrieves the provider list. Called as a Java method by GET_ALL_PROVIDERS

retrieveCachedInfo
Called as a Java method by these services:

- CONTINUE_CHECKIN
- CONTINUE_SUBMIT_HTML_FORM
- GET_CACHED_CHECKIN_INFO

retrieveProblemReportInfo
Retrieves the problem report information. Called as a Java method by these services:

- GET_UPDATE_PROBLEMREPORT_FORM
- PROBLEMREPORT_INFO
- RESEND_PROBLEMREPORT

retrieveProviderInfo
Retrieves registered provider information. Called as a Java method by GET_PROVIDER_INFO.

retrieveUserDatabaseProfileData
Retrieves the user database profile information. Called as a Java method by REGISTER_USER.

runHtmlConversion
Called as a Java method by GET_DYNAMIC_CONVERSION_SUB.

A.17 S

saveGlobalIncludes
Saves the global includes. Called as a Java method by SAVE_GLOBALINCLUDES.

saveTemplateConversions
Called as a Java method by SAVE_TEMPLATE_CONVERSIONS

saveUserOptionList
Saves the user option list. Called as a Java method by UPDATE_USEROPTION_LIST.

searchCacheReport
Called as a Java method by APPEND_SEARCH_AUDIT_INFO.

searchProblemReports
Retrieves problem report search information. Called as a Java method by GET_PROBLEMREPORTS_SEARCH_RESULTS.

sendMailTo
Sends an email to the problem report contributor. Called as a Java method by NOTIFY_CONTRIBUTOR.

setConditionVars
Called as a Java method by these services:

- ADD_COLLABORATION
- ADD_COLLABORATION_FORM
- ADD_WORKFLOW
- ADD_WORKFLOWALIASES
- ADD_WORKFLOWDOCUMENT
- ADD_WORKFLOWDOCUMENTS
- ASSIGN_DOCINFO_FORM
- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN
- CRITERIAWORKFLOW_DISABLE
- CRITERIAWORKFLOW_DISABLE_SUB
- CRITERIAWORKFLOW_ENABLE
- DELETE_COLLABORATION
- DELETE_WFCONTRIBUTORS
- DELETE_WORKFLOW
- DELETE_WORKFLOWCRITERIA
- DELETE_WORKFLOWDOCUMENTS
- EDIT_COLLABORATION
- EDIT_COLLABORATION_FORM
- EDIT_WORKFLOW
- EDIT_WORKFLOWCRITERIA
- GET_CLBRA_INFO
- GET_WORKFLOWDOCREVISIONS
- GET_WORKFLOWDOCUMENTS
- WORKFLOW_APPROVE
- WORKFLOW_CANCEL
- WORKFLOW_REJECT
- WORKFLOW_REJECT_FORM
- WORKFLOW_START

setDateToPresent

Sets the current date and passes the specified parameter. Called as a Java method by these services:

- SUBSCRIBE_EX
- UPDATE_SUBSCRIPTION_NOTIFY
- UPDATE_SUBSCRIPTION_USED

setDeleteRevReleaseState

Updates the revision status as deleted. Called as a Java method by these services:

- DELETE_BYREV
- DELETE_REV
- DELETE_REV_EX

setExternalDocInfoFields

Sets the external content item information fields. Called as a Java method by GET_EXTERNAL_DOC_INFO.

setFileConversionInfo

Called as a Java method by GET_DYNAMIC_CONVERSION.

setInputConversionInfo

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB.

setLocalValues

Sets the associated local values and passes the specified parameters. Called as a Java method by these services:

- ADD_GROUP
- ADD_PROBLEMREPORT
- ADD_PROXIEDCOLLECTION
- ADD_ROLE
- ADD_USER
- ASSIGN_DOCINFO_FORM
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_ARCHIVE_NO_NOTIFY
- CHECKIN_LIST
- CHECKIN_NEW_FORM
- CHECKIN_SIMILAR_FORM
- CLEAR_SERVER_OUTPUT
- CONTINUE_CHECKIN
- CONTINUE_SUBMIT_HTML_FORM
- DELETE_CHECKIN_CACHE
- DELETE_ROLE
- EDIT_CLBRA_ACCESS_LIST_FORM
- EDIT_USER
- EDIT_USER_PROFILE
- GET_ACTIVE_WORKFLOWS
- GET_CLBRA_DOCUMENTS
- GET_DOC_CONFIG_INFO

- GET_EXPIRED
- GET_SYSTEM_AUDIT_INFO
- GET_WORKFLOWS
- LOAD_PNE_PORTAL
- LOAD_WORKFLOW_QUEUE
- REGISTER_USER
- REMOVE_PROXIEDTRANSFER
- RESUBMIT_FOR_CONVERSION
- REVIEW_WORKFLOW_DOC
- UPDATE_BYREV
- UPDATE_DOCINFO_BYREV
- UPDATE_DOCINFO_SUB
- VALIDATE_DOCINFO
- WORK_IN_PROGRESS
- WORKFLOW_APPROVE
- WORKFLOW_CHECKIN_SUB
- WORKFLOW_REJECT

setOutputConversionInfo

Called as a Java method by GET_DYNAMIC_CONVERSION_SUB.

setStatusMessage

Sets the status message and passes the specified parameter. Called as a Java method by these services:

- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB
- DELETE_BYCLASS
- DELETE_BYNAME
- DELETE_BYREV
- DELETE_DOC
- DELETE_REV
- DELETE_REV_EX
- INSERT_NATIVE
- INSERT_NEW
- REPLACE_METAFILE_SUB

- UPDATE_BYREV
- WORKFLOW_CHECKIN_SUB

setTemplateConversionInfo
Called as a Java method by GET_DYNAMIC_CONVERSION_SUB.

setTraceFlags
Called as a Java method by EDIT_TRACE_OPTIONS.

startCriteriaWorkflow
Enables the criteria workflow. Called as a Java method by CRITERIAWORKFLOW_ENABLE.

startWorkflow
Starts the workflow process and passes *WfDocuments* as a parameter. Called as a Java method by WORKFLOW_START.

storeUserDatabaseProfileData
Stores the user profile information in the database. Called as a Java method by these services:

- ADD_USER
- EDIT_USER
- EDIT_USER_PROFILE
- REGISTER_USER

submitForm
Submits the form. Called as a Java method by FORM_SUBMIT.

submitHtmlForm
Submits the form. Called as a Java method by these services:

- CACHE_SUBMIT_HTML_FORM
- CONTINUE_SUBMIT_HTML_FORM
- SUBMIT_HTML_FORM

A.18 T

testProvider
Tests the provider. Called as a Java method by TEST_PROVIDER.

testWorkflowScript
Called as a Java method by TEST_WORKFLOW_SCRIPT.

A.19 U

Ualias
Updates an alias. Called as an Execute Query action by EDIT_ALIAS.

UcheckoutRevision

Updates the revision as checked out. Called as an Execute Query action by CHECKOUT_SUB.

Ucollaboration

Called as an Execute Query action by EDIT_CLBRA_ACCESS_LIST and EDIT_COLLABORATION.

UdeleteRevision

Updates the revision status as deleted. Called as an Execute Query action by these services:

- DELETE_BYREV
- DELETE_REV
- DELETE_REV_EX

UdocFormat

Updates the content item format information. Called as an Execute Query action by EDIT_DOCFORMAT.

UdocType

Updates the content item type. Called as an Execute Query action by EDIT_DOCTYPE.

UextensionMap

Updates the extension map. Called as an Execute Query action by EDIT_DOCEXTENSION.

Umeta

Updates the metadata information. Called as an Execute Query action by these services:

- UPDATE_BYREV
- UPDATE_DOCINFO_BYREV
- UPDATE_DOCINFO_SUB
- UPDATE_METADATA
- WORKFLOW_CHECKIN_SUB

Umetadef

Updates the meta data definition information. Called as an Execute Query action by EDIT_METADEF.

UnextCounter

Updates the counter. Called as an Execute Query action by ADD_PROBLEMREPORT.

UnextRevID

Updates the revision ID. Called as an Execute Query action by these services:

- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB

- INSERT_NEW

uninstallComponent
Called as a Java method by UNINSTALL_COMPONENT.

updateAffectedWorkflows
Called as a Java method by EDIT_COLLABORATION.

updateCache
Called as a Java method by UPDATE_FILTER_INFO.

updateCollaborationCache
Called as a Java method by these services:

- ADD_COLLABORATION
- DELETE_COLLABORATION
- EDIT_CLBRA_ACCESS_LIST
- EDIT_COLLABORATION

updateComponentConfig
Called as a Java method by UPDATE_COMPONENT_CONFIG.

updateConversionCache
Called as a Java method by GET_DYNAMIC_CONVERSION_SUB.

updateDocFormatsConfig
Updates content item format configuration. Called as a Java method by EDIT_DOC_FORMATS.

updateDocInfo
Called as a Java method by UPDATE_DOCINFO_BYREV and UPDATE_DOCINFO_SUB.

updateFilterConfig
Called as a Java method by UPDATE_FILTER_INFO.

updateLicense
Called as a Java method by UPDATE_LICENSE.

updateMetaTable
Updates the meta data information table. Called as a Java method by UPDATE_META_TABLE.

updateOptionList
Updates the option list table. Called as a Java method by UPDATE_OPTION_LIST.

updateProblemReport
Updates the problem report information in the database. Called as a Java method by UPDATE_PROBLEMREPORT.

updateResultTemplate

Updates the result template. Called as a Java method by UPDATE_RESULT_TEMPLATE.

updateRevisionIDAndLabel

Updates the revision ID and revision label. Called as a Java method by CHECKIN_SEL_FORM.

updateSubscriptionType

Updates the subscription type. Called as a Java method by UPDATE_SUBSCRIPTION_TYPE.

updateUserLocale

Called as a Java method by EDIT_USER_PROFILE.

updateUserMeta

Called as a Java method by UPDATE_USER_META.

updateUserMetaTable

Called as a Java method by UPDATE_USER_META_TABLE.

updateWorkflowAndDocState

Updates the workflow and content item state information. Called as a Java method by WORKFLOW_APPROVE and WORKFLOW_REJECT.

updateWorkflowState

Updates the workflow state. Called as a Java method by DELETE_WORKFLOWDOCUMENTS.

updateWorkflowStateAfterCheckin

Called as a Java method by WORKFLOW_CHECKIN_SUB.

UproblemReport

Updates the problem report information in the database. Called as an Execute Query action by UPDATE_PROBLEMREPORT.

Urevision

Called as an Execute Query action by WORKFLOW_CHECKIN_SUB.

Urevision2

Called as an Execute Query action by these services:

- UPDATE_BYREV
- UPDATE_DOCINFO_BYREV
- UPDATE_DOCINFO_SUB

UrevisionStatus

Updates the revision status. Called as an Execute Query action by WORKFLOW_REJECT.

UroleDefinition

Updates the role definition. Called as an Execute Query action by ADD_GROUP and ADD_ROLE.

UsecurityGroup

Updates the security group. Called as an Execute Query action by EDIT_GROUP.

UsubscriptionNotification

Updates the subscription notification data. Called as an Execute Query action by UPDATE_SUBSCRIPTION_NOTIFY.

UsubscriptionUse

Updates the user subscription. Called as an Execute Query action by UPDATE_SUBSCRIPTION_USED.

Uuncheckout

Updates the checkout information. Called as an Execute Query action by these services:

- UNDO_CHECKOUT
- UNDO_CHECKOUT_BY_NAME
- UPDATE_BYREV
- WORKFLOW_CHECKIN_SUB

UuncheckoutPrevID

Updates the revision ID. Called as an Execute Query action by CHECKIN_BYNAME and CHECKIN_SEL_SUB.

UuserAuthType

Updates the user authorization type. Called as an Execute Query action by CHANGE_USER_AUTH_TYPE.

Uworkflow

Updates the workflow. Called as an Execute Query action by EDIT_WORKFLOW.

UworkflowCriteria

Updates the workflow criteria. Called as an Execute Query action by EDIT_WORKFLOWCRITERIA.

UworkflowDocStep

Updates the content item workflow step. Called as an Execute Query action by WORKFLOW_REJECT.

UworkflowWithProject

Updates workflow project information. Called as an Execute Query action by EDIT_WORKFLOWCRITERIA.

A.20 V

validateCheckinData

Validates the check in data and passes the specified parameter. Called as a Java method by these services:

- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN
- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB
- INSERT_NATIVE
- INSERT_NEW
- UPDATE_BYREV
- UPDATE_DOCINFO_BYREV
- UPDATE_DOCINFO_SUB
- VALIDATE_DOCINFO
- WORKFLOW_CHECKIN_SUB

validateCollaboration

Called as a Java method by these services:

- ADD_COLLABORATION
- EDIT_CLBRA_ACCESS_LIST
- EDIT_COLLABORATION

validateDelete

Validates the delete request. Called as a Java method by DELETE_USER

validateMetaData

Called as a Java method by UPDATE_METADATA.

validateProblemReport

Validates the problem report. Called as a Java method by ADD_PROBLEMREPORT and UPDATE_PROBLEMREPORT.

validateStandard

Validates the standard data. Called as a Java method by these services:

- CACHE_CHECKIN_NEW
- CACHE_CHECKIN_SEL
- CACHE_SUBMIT_HTML_FORM
- CACHE_WORKFLOW_CHECKIN

- CHECKIN_BYNAME
- CHECKIN_NEW_SUB
- CHECKIN_SEL_SUB
- INSERT_NATIVE
- INSERT_NEW
- UPDATE_BYREV
- UPDATE_DOCINFO_BYREV
- UPDATE_DOCINFO_SUB
- VALIDATE_DOCINFO
- WORKFLOW_CHECKIN_SUB

validateSteps

Validates the steps for *WfSteps* and *QworkflowStepAliases*. Called as a Java method by CRITERIAWORKFLOW_DISABLE and WORKFLOW_START.

validateUserNameAndType

Called as a Java method by ADD_USER and CHANGE_USER_AUTH_TYPE.

viewDoc

Displays the content item and passes *SearchCollectionDocInfo* as a parameter. Called as a Java method by VIEW_DOC.